```
</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 Scripte 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 DirecX-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 Laufzeitsystemes 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 Laufzeitsystemes 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 Rechnersystemes nötig werden kann. DA-Aktionen sind also **sparsam** zu verwenden ! Direct Animation stammt - trotz der noch immer ansehlichen 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 windowseigenen Lautstärkeregelung die vom User als dauerhaft eingestellten Reglerstellungen zum Ärgernis des Users temporär oder sogar permanent verändert werden. Eine Synchronisierung mit Laustärken 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 kodierten, 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 Linzensierung 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 verusacht 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 kostenpflichte 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 fehlerhatem 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:
Scripfehleranzeige 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 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.visivility="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 kannauch 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/enus/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 wird nun durch das Windows Media Player ActiveX Steuerelement abgedeckt. Das Active Movie Runtime Steuerelement wird daher nicht mehr unterstützt, von einer Freigabe wirdabgeraten.

Media Player / ActiveMovie-Steuerelement{05589FA1-C356-11CE-BF01-00AA0055595A} Die Funktionalität dieses Steuerelements wird 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 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 Schaltfächen auf Webseiten. Das Steuerelement wird nicht mehr unterstützt und dürfte nur noch vereinzelt im Einsatz sein. Von derFreigabe 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-Bilbiothek 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-Bilbiothek 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 geparst 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 instanziert 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_Diviend, 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_Usprung_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_Usprung_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(AchtEcktKoordinatenFeld);

// ----- 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 mir 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 ezeugt eine Berg- und Tal-Ansicht, die auch als Phasenmodell bezeichnet wird.

Ohne die zeitliche Streckung gilt folgender Ansatz:

Kreis:	Kreisumfang	ist 2 * Pi * 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 * AnzahlSchwingungen mit 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 Usprunges auf der Y-Achse im Rahmen des zeitlichen Intervalls.
Ohne zeitliches Intervall ist die Elongation die Position auf dem Kreisumfang, also Sinus(2 * Pi * 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-RotartionsAchsen ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
        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_Diviend, DA_Divisor);}

// ##### DA-Routine zur Ermittlung der Elongation mit zeitlicher Steckung und periodischem Wert
function DA_ElongationErmitteln()          // 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 * AnzahlSchwingunen   /  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 / Zeitehineit)
        //                                                                                      * 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-Bilbiothek 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_ElongationErmitteln();

// ##### 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_AbweichungDer2DAnimationsachseVomZentrumDerGesamten2DAnimation =
                                                  ScriptWertWert_Nach_DA_Wert_Konvertieren(
                                  ichungDer2DAnimationsachseVomZentrumDerGesamten2DAnimation
                                                                                          );
// - - - Schritt 3
DA_2DAnimationsAchse  = DA_Bibliothek.Translate2Anim(
                                  DA_AbweichungDer2DAnimationsachseVomZentrumDerGesamten2DAnimation,
                                  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 ensprechenden 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:\\dxm\\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 automatischer 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 automatischer 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_Animier
                                                                                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 =
```


```
                                                DA_AlleFontZeichenAlsGesamtBild_NichtAnimiert.Transform(DA_Punk
                                        t);

                        // ##### 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 = 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:500;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
                    //                  äßeres 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 ahand 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 positioneren
                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 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>
```

5.2.2.3.8. 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
                //              äßeres 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 Eventverabeitung 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_Verandern_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:300;height:300"
        >
        </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 Frewall müsste Zugriffe beanstanden (Afrage 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 Virenscanner 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 Virenscanner kann Scriptanweisungen mit Bezug auf das FileSystemObject als virenverseuchtes Script erkennen und die Scriptaktivierung verhindern. Falls der Virenscanner 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 Virenscanner 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