img Objekt des Internet Explorer

Erzeugung unter HTML:

Beispiele:

Bild nicht im Formular:

```
<IMG
        SRC="url_des_bildes_in_voller_auflösung"
        NAME="logischer_bild_name"
        LOWSRC="url_des_bildes_in_geringer_auflösung"
        ALT="alternativer_text"
        ALIGN=ausrichtung
        HEIGHT=hoehe
        WIDTH=breite
        BORDER=rahmenbreite
        HSPACE=abstand_links_und_rechts_zur_umgebung
        VSPACE=abstand_oben_und_unten_zur_umgebung
        ISMAP
        USEMAP=         "map_url#image_map"
                oder    "map_name"
        onAbort="eventhandler1"
        onerror="eventhandler2"
        onLoad="eventhandler3"
>
```

Bild im Formular:

```
<INPUT  TYPE=image
        SCR=url_des_bildes_in_voller_auflösung"...
>
```

Beispiel für Linie mit variabler Dimension:

Es wird ein Bild aus 1x1 neu dimensioniert, das eine kleine Dateigröße hat und nur 1 mal geladen werden muss. Durch die Angaben von Breite und Höhe kann eine vertikale oder horizontale Linie erzeugt werden. Die Linienpositionierungerfolgt per STYLE-Attribut.

horizontale Linie mit 10 Pixel Dicke:

```
<IMG SRC="1x1bild.gif" WIDTH="300" HEIGHT="10" BORDER="0" ALT="" STYLE=" ....">
```

vertikale Linie mit 10 Pixel Dicke:

```
<IMG SRC="1x1bild.gif" WIDTH="10" HEIGHT="300" BORDER="0" ALT="" STYLE=" ....">
```

Erzeugung in Script:

```
var Bild =  new Image([breite, hoehe]);
```

erzeugt Instanz und lädt zugleich das Bild
zeigt das Bild NICHT an
Verwendung z.B. bei animiertem Bild, wobei die Animation geladene Bilder voraussetzt

Zugriff:

Bild nicht im Formular:

```
document.ID_Img.eigenschaft
document.images[index].eigenschaft
```

index: ab 0
muss in **[ ]** kodiert sein

Bild im Formular:

```
document.ID_Img.eigenschaft
document.images[index].eigenschaft
document.ID_Formular.elements[index].eigenschaft
```

index: ab 0
muss in **[ ]** kodiert sein

Bild in Script: per Variable aus new

Beispiel für Belegen des Attributes SRC am Beispiel des Vorladens eines Bildes:

```
var Bild_Hoehe=45; var Bild_Breite=60; var Bild_Url="test.gif";
var BildObjekt=new Image(Bild_Breite,Bild_Hoehe);
BildObjekt.src=Bild_Url;                        // ohne "" kodieren da Zeiger
document.write(         '<IMG NAME="IMG_Bild"'
                        +   ' SRC="'      + BildObjekt.src + ""
                        +   ' HEIGHT=' + Bild_Hoehe
                        +   ' WIDTH='  + Bild_Breite
```

```
                    + '>'
                );
```

Eigenschaften (ausgewählte):

.border	entspricht BORDER nur lesen
.complete	wenn mit **true** belegt, so Bild komplett geladen wenn mit **false** belegt, so Bild noch nicht komplett geladen nur lesen
.height	entspricht HEIGHT nur lesen
.hspace	entspricht HSPACE nur lesen
.lowsrc	entspricht LOWSRC
.name	entspricht NAME nur lesen
.src	entspricht SRC
.vspace	entspricht VSPACE; Standard ist 0 nur lesen
.width	entspricht WIDTH nur lesen

Beispiel für Belegen des Attributes SRC am Beispiel des Vorladens eines Bildes:

```
var Bild_Hoehe=45; var Bild_Breite=60; var Bild_Url="test.gif";
var BildObjekt=new Image(Bild_Breite,Bild_Hoehe);
BildObjekt.src=Bild_Url;                    // ohne "" kodieren da Zeiger
document.write(         '<IMG NAME="IMG_Bild"'
                    +   ' SRC="'      + BildObjekt.src + ""
                    +   ' HEIGHT=' + Bild_Hoehe
                    +   ' WIDTH='  + Bild_Breite
                    + '>'
                );
```

Events bei sich überlagernden Objekten:

Beispiel: Sollte ein Image mit seiner Erzeugung ein anderes Image überlagern, so ist die Eventübergabe von und an das untere Image unterbrochen: Ereignisse onXXX kommen nicht mehr durch, solange das untere Bild nicht den **Fokus** erhält. Alternativ ist die Style-Eigenschaft z-index zu kodieren und dann der z-index auf den obersten zu setzten.

img Objekt des Internet Explorer:

HTML-Container für ein Bild, Videoclip, VRML-Datei
Scriptsteuerung erst ab IE 4.x
Anwendung auch im Formular per input image Objekt
Die Collection images wird unter document.images beschrieben !

ab IE 5.x werden folgende Image-Dateiarten unterstützt:

*.avi	Audio-Visual Interleaved (AVI)
*.bmp	Windows Bitmap (BMP)
*.emf	Windows Enhanced Metafile (EMF)
*.gif	Graphics Interchange Format (GIF)
*.jpg, *.jpeg	Joint Photographic Experts Group (JPEG)
*.mov	Apple QuickTime Movie (MOV)
*.mpg, *.mpeg	Motion Picture Experts Group (MPEG)
*.png	Portable Network Graphics (PNG)
*.wmf	Windows Metafile (WMF)
*.xbm	X Bitmap (XBM)

Hinweis: Ereignis onfocus nicht ausgelöst bei einem MAP-Image
Kodierung der Datei-Url:
für statisches Bild das Attribut SRC verwenden
für Videoclip oder VRML-Datei (virtual reality modeling language-Datei) das Attribut DYNSRC verwenden

Beispiel

```
<IMG SRC=mygraphic.bmp>
```

Beispiel für Aus-und Einblende eines Bildes:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
    var Sichtbar=true;      // DIV ist natürlich nach dem Dokument-Laden sichtbar
    var StandardFarbe="green";

    function Blenden()
    {
```

```
                        //  Fade zurücksetzen
                        DIV_ID.filters[0].Apply();

                        if (Sichtbar)
                        {
                                Sichtbar=false;
                                DIV_ID.style.visibility="hidden";
                        }
                        else
                        {
                                Sichtbar=true;
                                DIV_ID.style.visibility="visible";
                                DIV_ID.style.backgroundColor=StandardFarbe;
                        }

                        //  Fade starten
                        DIV_ID.filters[0].Play();
                }
// -->
</SCRIPT>
</HEAD>
<BODY>
<BUTTON onclick="Blenden()">
        Aus- und Einblenden
</BUTTON>
<BR>
<BR>
<DIV    ID="DIV_ID"
        STYLE="height:250px;width:250px;background-color:red;
                filter:progid:DXImageTransform.Microsoft.Fade(duration=1);"
                //  alles EINE Zeile !!
>
        <IMG SRC="Bild.gif" ....HEIGHT=250 WIDTH=250>
                //  Bild-Dimension muss in die DIV-Dimension reinpassen !
</DIV>
</BODY>
</HTML>
```

Beispiel für Aus- und Einblende von Bildern mit Bildwechsel:

Variante 1

Es werden zwei DIV an absoluter Position überlagert. Jedes DIV hat einen eigenen Inhalt, hier im Beispiel sein eigenes Bild. Durch das Ein- und Ausblenden wird ein Slide-Show-Effekt der beiden Bilder erreicht.
Pro Bild wird ein eigenes DIV erzeugt.

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
        var DIV_ID1_Sichtbar=true;      //  DIV_ID1 ist nach dem Dokument-Laden sichtbar
                                        //          DIV_ID2 aber nicht

        function BildWechsel()
        {
                //  Fade zurücksetzen
                DIV_ID1.filters[0].Apply;
                DIV_ID2.filters[0].Apply;

                if (DIV_ID1_Sichtbar)
                {
                        DIV_ID1_Sichtbar = false;
                        DIV_ID1.style.visibility="hidden";
                        DIV_ID2.style.visibility="visible";
                }
                else
                {
                        DIV_ID1_Sichtbar = true;
                        DIV_ID1.style.visibility="visible";
                        DIV_ID2.style.visibility="hidden";
                }
                //  Fade starten
                DIV_ID1.filters[0].Play();
                DIV_ID2.filters[0].Play();
        }
```

```
// -->
</SCRIPT>
</HEAD>
<BODY>
	<BUTTON onclick="BildWechsel()">
		Bild wechseln mit Aus- und Einblenden
	</BUTTON>
	<BR>
	<BR>
	<DIV	ID="DIV_ID1"
		STYLE="height:250px;position:absolute;top:50;width:250px;
			background-color:red; visibility:visible
			filter:progid:DXImageTransform.Microsoft.Fade(duration=1);"
		 //  alles EINE Zeile
	>
		<IMG SRC="DIV_ID1_Bild.gif" ....HEIGHT=250 WIDTH=250>
		//  Bild-Dimension muss in die DIV-Dimension reinpassen !
	</DIV>

	<DIV	ID="DIV_ID2"
		STYLE="height:250px;position:absolute;top:50;width:250px;
			background-color:red; visibility:hidden
			filter:progid:DXImageTransform.Microsoft.Fade(duration=1);"
		 //  alles EINE Zeile
	>
		<IMG SRC="DIV_ID2_Bild.gif" ....HEIGHT=250 WIDTH=250>
		//  Bild-Dimension muss in die DIV-Dimension reinpassen !
	</DIV>
</BODY>
</HTML>
```

Variante 2

Es wird der innere DIV-Bereich zwischen <DIV> und </DIV> ersetzt und damit je ein Bild dargestellt. Durch das Ein- und Ausblenden wird ein Slide-Show-Effekt der Bilder erreicht.

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
	var BildUrlFeld=Array
	(
		//  hier beliebig viele Bilder eintragen
		"test1.jpg",
		"test2.jpg",
		"test3.jpg"
	);

	var BildUrlFeld_Laenge= BildUrlFeld.length;

	//		fuer alle Bilder die identischen Dimensionen !!!
	var BildHoehe=444;
	var BildBreite=640;

	//  Zeigerfeld zum Vorladen der Grafiken
	var BildZeigerFeld = Array();

	var Index=1;		//  Starten mit test2.jpg, da test1.jpg bereits angezeigt mit Laden des DIV

	function BildWechsel()
	{
		//  Fade zurücksetzen
		DIV_ID.filters[0].Apply();

		// Bild im DIV anzeigen
		document.AktuellesBild.src= BildZeigerFeld[Index].src;

		//  Fade starten
		DIV_ID.filters[0].Play();

		// nächstes Bild einstellen
		Index++;

		//  Index korrigieren
		if (Index >= BildUrlFeld_Laenge)
```


```
                {Index=0;}
        }

        //  nachfolgender Code wird mit dem Laden des HEAD-Teiles abgearbeitet

        // Bildobjekte vorladen: allokieren und Zeiger merken per Prototyping
        for (i=0; i<= BildUrlFeld_Laenge; i++)
        {
                //  Image-Zeiger bilden als Feldeintrag
                BildZeigerFeld[i] = new Image();

                //  Url dem Zeiger zuweisen und Bild somit vorladen
                BildZeigerFeld[i].src= BildUrlFeld[i];
        }

        //  DIV erzeugen mit  test1.gif als Startbild
        //                          in Dimension aller Bilder
        document.write(         '<DIV '
                        +       'ID="DIV_ID" '
                        +       'STYLE="height:' + BildHoehe + 'px;width:' + BildBreite + 'px;'
                        +               'filter:progid:DXImageTransform.Microsoft.Fade(duration=1);'
                        +               ""
                        +   '>\n'
                );

        document.write(                 '<IMG  NAME="AktuellesBild"'
                        +               'SRC="'   + BildZeigerFeld[0].src + '" '
                        +               'WIDTH='  + BildBreite + ' '
                        +               'HEIGHT=' + BildHoehe
                        +       '>\n'
                );

        document.write(         '</DIV>\n');

        //  Button zum Bildwechsel anzeigen
        document.write(          '<BUTTON onclick="BildWechsel()">'
                        +               'Naechstes Bild'
                        +  '</BUTTON>\n'
                );
// -->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>
```

Beispiel für mehrere Bilder in den RAM laden und mit Wartezeit preloaden:

```
<HTML>
<HEAD>
        <SCRIPT LANGUAGE="JavaScript1.2" TYPE="text/javascript">
        <!--
                var bild_feld =  [      "b1.gif",
                                        // hier alle Bilder eintragen.
                                ];

                var wartezeit = 500; // Zeit in ms zwischen zwei Ladevorgaengen
                var feld_index = 0;

                function preloaden()
                {
                        var bild = new Image();
                        bild.src = bild_feld[feld_index];

                        feld_index ++;

                        if(feld_index < bild_feld.length) // Länge ab 1, Index ab 0
                        {setTimeout('preloaden()', load_next)}
                }

                function start()
                {setTimeout('preloaden()', wartezeit)}
        // -->
        </SCRIPT>
```

```
</HEAD>
<BODY ... onLoad="start()">
</BODY>
</HTML>
```

Beispiele für Bildfolge:

Beispiel 1:

10 Bilder mit Namen b1.gif bis b10.gif. Wichtig ist der numerische Namenteil 1 bis 10, da als Zähler verwendet

```
<HTML>
<HEAD>
        <SCRIPT  LANGUAGE="JavaScript">
        <!--

                verzoegerung = 120; // in Millisekunden
                bildNummer = 2;

                bilder= new Array(); //nimmt die Bilder der Animation auf

                // hier werden die Bilder im Hintergrund geladen
                for (i = 1; i <= 10; i++)
                {
                        bilder[i] = new Image();
                        bilder[i].src = "b" + i + ".gif";
                }

                function naechstesBild()
                {
                        document.animation.src = bilder[bildNummer].src;
                        bildNummer++;
                        if (bildNummer > 10) bildNummer = 1;
                }
        // -->
        </SCRIPT>
</HEAD>

<BODY>
        <IMG    SRC="dp1.gif"
                NAME="animation"
                WIDTH="165"
                HEIGHT="185"
                onLoad="setTimeout('naechstesBild()', verzoegerung)">
</BODY>
</HTML>
```

Beispiel 2:

Es können **geladene Bilder** bild1.gif bis bild5.gif an fester Position im Wechsel ausgegeben werden.
geladenen Bilder erzeugen per Image-Objekt
erstes Bild wird angezeigt per <IMG>
alle Bilder müssen gemeinsame Dimension haben, sonst wird gestreckt oder gestaucht
Durch Aktualisierung vom Attribut SRC aus <IMG> mit der jeweiligen Url des Bildes wird der sichtbare Bildwechsel vollzogen.

```
<HEAD>
<SCRIPT ...>
<!--    //  alle Bilder vorladen

        // globale Größen festlegen

        var bild_anzahl=5;              // ab 1

        var feld=new Array(bild_anzahl);        // Inhalt des Feldes ist hier noch unbekannt

        feld[1]=new Image();            // Feldelement 1 ist Image-Objekt
                                        //      Bild 1 wird vorgeladen und nicht nicht agenzeigt,
                                                da der Javascript-Code VOR dem IMG-Tag
                                                aus <BODY> .. </BODY> abgearbeitet wird !
        feld[1].url="bild1.gif";        // Prototyping:  Eigenschaft url  hinzufügen für Index 1
        .....

        feld[5]=new Image();            // Feldelement 5 ist ein Image-Objekt
        feld[5].url="bild5.gif";        // Prototyping:  Eigenschaft url hinzufügen für Index 5
```


```
		var index=bild_anzahl;		//  auch möglich		var index= feld.length;

		function bild_wechsel()
		{
			// Nummer des aktuellen Bild ermitteln
			if (index == bild_anzahl)
			{index=0;}

			index+=1;

			// Kopieren der Url nach <IMG>-Attribut SRC
			//		also gleichzeitig anzeigen
			self.document.logischer_img_name.url=feld[index].url;
		}
// -->
</SCRIPT>
</HEAD>
<BODY>
		<IMG	NAME="logischer_img_name"
			SRC="bild1.gif"			// vorgeladenes Bild anzeigen
		>
		<SCRIPT>
		<!--
			setInterval(bild_wechsel,2000);  // periodische Abarbeitung alle 2 Sekunden
		// -->
		</SCRIPT>
</BODY>
```

Beispiele für festes, nicht mitscrollendes Bild:

Beispiel 1:

feste Grafik in einem Fenster für Internet Explorer oder Netscape

```
<HTML>
<HEAD>
<TITLE> .... </TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!--
		posX=20;			// horizontal
		posY=20;			// vertikal
		idName="grafik_name";

		function merkeYPosition()
		{	if (document.all)		// IE
			{merkeY=document.body.scrollTop;}
			else				// Netscape
			{
				if (document.layers)
				{merkeY=pageYOffset;}
			}
		}

		funkction setzeYPosition()
		{	merkeYPosition();
			if (document.all)
			{document.all[idName].style.top=merkeY+posY;}
			else
			{	if (document.layers)
				{document.layers[idName].top=merkeY+posY;}
			}
		}

		function merkeXPosition()
		{	if (document.all)		// IE
			{merkeX=document.body.scrollLeft;}
			else				// Netscape
			{
				if (document.layers)
				{merkeX=pageXOffset;}
			}
		}

		funkction setzeXPosition()
```

```
		{		merkeXPosition();
				if (document.all)
				{document.all[idName].style.left=merkeX+posX;}
				else
				{
					if (document.layers)
					{document.layers[idName].left=merkeX+posX;}
				}

		}

		function korrigiereScrollen()
		{
				setzeYPosition();
				setzeXPosition();
				setTimeout("korrigiereScrollen()",1000); // alle Sekunde korrigieren
		}
//-->
</SCRIPT>
</HEAD>

<BODY onLoad="korrigiereScrollen()">
		<DIV	ID="grafik_name"   // identisch mit  idName
				STYLE="position:absolute; top:20; left:20; width:32px;">
								// top identisch mit Anfangswert von posY
								// left identísch mit Anganfswert von posX
				<IMG	SRC=grafik.gif
					WIDTH="32"	// identisch mit STYLE
					HIGHT="32"
					BORDER="0"
					ALT="grafik.gif"
				 >
		</DIV>

		// nachfolgender Code erzeugt einen Scrollfenster als Tabelle

		<TABLE  WIDT="800"
					HEIGHT="800"
					BGCOLOR="yellow"
		>
				<TR>
					<TD>  </TD>
				</TR>
		<TABLE>
</BODY>
</HTML>
```

Beispiel 2

```
<HTML>
<HEAD>
<SCRIPT TYPE="text/javascript" LANGUAGE="JavaScript1.2">
<!--
//			feste Grafik, die nicht mitscrollt
//			Grafik steht immer rechts unten im Fenster
//			Die Grafik-Positionskorrektur-Geschwindigkeit ist leider abhängig von der
//				Scrollgeschwindigkeit,
//				so dass ein Springen der Grafik unvermeidlich ist,
//				wenn die Scrollgeschwindigkeit zu gross ist.

//*************************************************************************************************
//
//			nachfolgende Variablen müssen vom Programmierer gesetzt werden
//
//*************************************************************************************************
var GrafikBreiteLautIMGinBODY=30;		// identisch mit IMG-Werten laut BODY
var GrafikHoeheLautIMGinBODY =49;		// identisch mit IMG-Werten laut BODY
var GrafikAbstandZurFensterEckeRechtsUntenX=0;	// in Pixel
var GrafikAbstandZurFensterEckeRechtsUntenY=0;	// in Pixel
var GrafikPositionsKorrekturGeschwindigkeit=1;	// >=1  wobei 1 = schnellste Positionskorrektur

//*************************************************************************************************
//
//			nachfolgenden Code nicht verändern
```

```
//
//*****************************************************************************************************
//		Variablen hier ohne extra Funktion intitialisieren
//			werden automatisch beim Dokumentladen belegt
//
//		BrowserTyp-Ermittlung
//			Annahme Netscape läuft
var BrowserTyp=true;
if (document.all)
{
	// IE läuft
	BrowserTyp=false;
}

// Browser-Fenster-Variablen
var BrowserFenster_AktuelleBreite	=0;
var BrowserFenster_AktuelleHoehe	=0;
var BrowserFenster_AktuelleScrollPositionX =0;
var BrowserFenster_AktuelleScrollPositionY =0;

// Grafik-Positions-Korrektur-Variablen
var GrafikPositionsKorrekturX=0;
var GrafikPositionsKorrekturY=0;

function GrafikPositionKorrigieren()
{
	// aktuelle Browserfenster-Daten ermitteln
	if (BrowserTyp)
	{
		// Netscape läuft
		BrowserFenster_AktuelleBreite		=window.innerWidth;
		BrowserFenster_AktuelleHoehe		=window.innerHeight;
		BrowserFenster_AktuelleScrollPositionX	=window.pageXOffset;
		BrowserFenster_AktuelleScrollPositionY	=window.pageYOffset;
	}
	else
	{
		//IE läuft
		BrowserFenster_AktuelleBreite		=document.body.clientWidth;
		BrowserFenster_AktuelleHoehe		=document.body.clientHeight;
		BrowserFenster_AktuelleScrollPositionX	=document.body.scrollLeft;
		BrowserFenster_AktuelleScrollPositionY	=document.body.scrollTop;
	}

	// Korrekturpositionen für Grafik ermitteln
	//			Achtung: Die Klammerung ist hier WICHTIG !!!
	GrafikPositionsKorrekturX=	BrowserFenster_AktuelleBreite
					- (GrafikBreiteLautIMGinBODY+GrafikAbstandZurFensterEckeRechtsUntenX)
					+ BrowserFenster_AktuelleScrollPositionX;

	GrafikPositionsKorrekturY=	BrowserFenster_AktuelleHoehe
					- (GrafikHoeheLautIMGinBODY+GrafikAbstandZurFensterEckeRechtsUntenY)
					+ BrowserFenster_AktuelleScrollPositionY;

	//  Grafikposition korrigieren
	// eval () ermittelt String und liefert dessen Inhalt anstelle eval()
	// da der String ein StyleSheed-Kommando ist, wird das somit ausgeführt
	if (BrowserTyp)
	{
		// Netscape läuft
		eval("document.StyleSheedID.left=" + GrafikPositionsKorrekturX);
		eval("document.StyleSheedID.top="  + GrafikPositionsKorrekturY);
	}
	else
	{
		// IE läuft
		eval("StyleSheedID.style.pixelLeft=" + GrafikPositionsKorrekturX);
		eval("StyleSheedID.style.pixelTop="  + GrafikPositionsKorrekturY);
	}

	setTimeout("GrafikPositionKorrigieren()",GrafikPositionsKorrekturGeschwindigkeit);
}
//-->
</SCRIPT>
```

```
</HEAD>
<BODY onload="GrafikPositionKorrigieren();">
<DIV ID="StyleSheedID" STYLE="position:absolute; visibility:show; left:0px; top:0px; z-index:2">
        <CENTER>
                <IMG SRC="picture.gif" WIDTH="30" HEIGHT="49" BORDER="0">
        </CENTER>
</DIV>
Zeile<BR>
Zeile<BR>
Zeile<BR>
Zeile<BR>
Zeile<BR>
....
</BODY>
</HTML>
```

Beispiele für Bild verschwinden lassen:

Variante1: Bild ausserhalb des sichtbaren Bereiches positionieren

```
if (document.layers)
{
        document.layers[DIV_ID].left    = ( -1 * 40);
        document.layers[DIV_ID.top      = ( -1 * 20);
}
else
{
        if (document.all)
        {
                document.all.DIV_ID.style.left    = ( -1 * 40);
                document.all.DIV_ID_.style.top  = ( -1 * 20);
        }
}

<DIV    ID="DIV_ID"
        STYLE="position:absolute;left=0px;top=0px;"
>
        <IMG    NAME="IMG_ID"
                SCR="alt.jpg"
                HEIGHT=20
                WIDTH=40
        STYLE="...."
>
</DIV>
```

Variante 2: Bild ersetzen durch treansparentes 1x1 Pixel

```
document.IMG_ID.src="neu.gif";          //  neu.gif ist transparent und 1x1 Pixel

<DIV    ID="DIV_ID"
        STYLE="position:absolute;left=0px;top=0px;"
>
        <IMG    NAME="IMG_ID"
                SCR="alt.jpg"
                HEIGHT=20
                WIDTH=40
        STYLE="...."
>
</DIV>
```

Vairante 3: Sichtbarkeit verändern per style.visibility

Achtung: style.visibility=hidden lässt den Platz des Bildes im Layout **nicht** bestehen, was dann wichtig ist, wenn Umgebung des Bildes **nicht** per STYLE="position:absolute ..." erzeugt wurde.

Beispiel 1:

```
<HEAD>
<STYLE>
        .vis1 { visibility:visible }
        .vis2 { visibility:hidden }
</STYLE>
</HEAD>
<BODY>
        <IMG ID="ID_IMG" SRC="test.jpg">
```


```
                <P      onmouseover="ID_IMG.className='vis1'"
                        onmouseout="ID_IMG.className='vis2'"
                >
                        Testtext
                </P>
        </BODY>
```

Beispiel 2:

```
<SCRIPT>
        function Verstecken()
        {ID_IMG.style.visibility="hidden"; }

        function Anzeigen()
        {ID_IMG.style.visibility="visible"; }
</SCRIPT>
<IMG    ID="ID_IMG"
        SRC="test.jpeg">
<SPAN   onmouseover="Verstecken()"
        onmouseout="Anzeigen()"
>
        Testtext
</SPAN>
```

Vairante 4: Sichtbarkeit verändern per style.display

Achtung: style.visibility=none lässt den Platz des Bildes im Layout bestehen, was dann wichtig ist, wenn Umgebung des Bildes **nicht** per STYLE="position:absolute ..." erzeugt wurde.

Beispiel 1:

```
<HEAD>
<STYLE>
        .vis1 { display:inline }
        .vis2 { display:none }
</STYLE>
</HEAD>
<BODY>
        <IMG ID="ID_IMG" SRC="test.jpg">
        <P      onmouseover="ID_IMG.className='vis1'"
                onmouseout="ID_IMG.className='vis2'"
        >
                Testtext
        </P>
</BODY>
```

Beispiel 2:

```
<SCRIPT>
        function Verstecken()
        {ID_IMG.style.display="none"; }

        function Anzeigen()
        {ID_IMG.style.display="inline"; }
</SCRIPT>
<IMG    ID="ID_IMG"
        SRC="test.jpeg">
<SPAN   onmouseover="Verstecken()"
        onmouseout="Anzeigen()"
>
        Testtext
</SPAN>
```

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
.align	Ausrichtung
.alt	Alternativer Text als Tooltip
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.border	Rahmendicke in Pixel

.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.complete	Zustand des Ladens des Objektes
.dataFld	Datenquelle-Name vergeben (ID)
.dataSrc	Datenquelle als Anker festlegen
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.dynsrc	Adresse von Videoclip oder VRML
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.fileCreatedDate	Datum der Dokumenterstellung
.fileModifiedDate	Datum der letzten Dokumentveränderung
.fileSize	Größe des Dokumentes
.fileUpdatedDate	Datum des letzten Datei-Updates
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.galleryImg	Toobar "My Pictures Photo Support image toolbar" ein/ausschalten für aktuelles Image
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.height	Höhe des Objektes in Pixel
.hideFocus	Focussierbarkeit
.hspace	horizontaler Abstand in Pixel zum Elternobjekt
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMap	server-seitige Image-Map
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.longDesc	Uniform Resource Identifier (URI) zur einer "long description" umwandeln
.loop	Anzahl der Wiederholungen nur Objekt BGSOUND bzw. IMG mit Sound bzw. INPUT-Element mit Sound
.lowsrc	Url des Images in geringerer Auflösung
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nameProp	Dateiname-Teil einer Url ohne Path und ohne Protokoll liefern
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeigt, .offserLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offestParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offestParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound

.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.ownerDocument	Referenz auf das document Objekt zu dem der Koten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.protocol	Protokoll-Teil einer Url inklusive http und ftp liefern
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
.src	Url der Daten z.B. vom Image in normaler Auflösung
.start	Start eines Videoclips oder VRML-Datei
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.syncMaster	Synchronisierung der Animation des im Container liegenden Elementes auf Timeline (Container ist Master in der Synchronisierung), wobei ein Master **nur** genau ein zu synchronisierendes Element haben darf (Eineindeutigkeit). nur sinvoll bei Zwangssynchronisierung (siehe .syncBehavior) ersetzt die Eigenschaft clockSource, da diese deprecated ist und nicht mehr verwendet werden darf ! siehe Objekt currTimeState und Behavior .style.time2 siehe .syncTolerance und .syncBehavior
.systemBitrate	wird hier nicht erklärt
.systemCaptions	wird hier nicht erklärt
.systemLanguage	Sprache festlegen für das Objekt
.systemOverdubOrSubtitle	wird hier nicht erklärt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
.uniqueID	durch den Browser automatisch-genriertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiede ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.useMap	Url oder Anker für client-seitige Image-Map
.vspace	vertikaler Abstand in Pixel zum Elternobjekt
.width	Breite des Objektes in Pixel

Methoden:

.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparst wurde

Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im
im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !

.attachEvent() Einschalten des Registrieren eines Events durch Eventhandler
Hinweis: Abschalten mit Methode .detachEvent()
Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider
NICHT verkettet sondern in **Zufallsfolge**, es sei denn
die Handler prüfen ihre Aufruffolge (muss programmiert werden)

.blur() Element den Focus wegnehmen und Event onblur auslösen
Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt !
vor IE 5.0 TABINDEX-Attribut muss kodiert sein
ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein

.clearAttributes() alle HTML-Attribute eines Objektes entfernen
außer ID, STYLE und per Script definierte Attribute
Script-erzeugte Attribute nicht entfernbar
DOM wird geändert

.click() simuliert einen Klick auf das Element und löst onclick-Event aus
manipuliert nicht den Focus

.cloneNode() Objekt klonen und Referenz des erzeugten Klone liefern
DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)

.componentFromPoint() Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt
auch für CSS-Layout
onmouseover-Event hat **nicht die Pixelgenauigkeit** wie die Angaben der Methode .componentFromPoint()
also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben
Overbereich der Maus ist mehr als 1 Pixel gross
beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.

.contains() prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist
DOM nicht geändert

.detachEvent() Abschalten des Registrieren eines Events durch Eventhandler
wobei Registrierung mit Methode .attachEvent() aktiviert wurde
Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das **nicht** behandelt werden soll, falls es für das Window-Objekt auftritt
(also Standardbehandlung aktiv)

.dragDrop() prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element

.fireEvent() ein Event auslösen

.focus() Focus setzen und Focus-Event auslösen
nur nach dem kompletten Laden des Dokumentes
vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen

.getAdjacentText() Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann
Text kann HTML-Tags enthalten, muss aber nicht
DOM nicht geändert

.getAttribute() Wert eines per HTML erzeugten Attributes liefern
DOM nicht geändert

.getAttributeNode() Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft.
Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht
Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt
Wert des Attributes wird somit über die Referenz laut DOM erreichbar
DOM nicht geändert

.getBoundingClientRect() Referenz auf TextRectangle-Objekt im Element holen

.getClientRects() Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster
Feld mit Index als Integer ab 0
pro Eintrag ein Rectangle

.getElementsByTagName() Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc.
Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden
(beachte dabei document.all Collection)
Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst !
Für Verwaltung per ID (analog zum ID-Attribut):
siehe Methode getElementsById()
Für Verwaltung per NAME (analog zum NAME-Attribut):
siehe Methode .getElementsByName()
DOM nicht geändert

.getExpression() Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern
Style-Eigenschaft ist per Methoden
expression() oder setExpression()
zu definieren
DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout
(nach dem eventuellen expliziten Dokument-Refresh)

.hasChildNodes() prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt
DOM nicht geändert

.insertAdjacentElement() Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann

wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM
nur nach dem kompletten Laden des Dokumentes möglich
DOM wird geändert

.insertAdjacentHTML() HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann
nur nach dem kompletten Laden des Dokumentes möglich
HTML- und Script-Code müssen syntaktisch korrekt sein
wenn nicht, so wird das Einfügen **nicht** ausgeführt
eingefügter Code wird **nur** dann sofort geparst und ausgeführt, wenn syntaktisch korrekt ist
bei Script-Code: <SCRIPT DEFER .....> muss kodiert werden
DOM wird geändert

.insertAdjacentText() Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann
nur nach dem kompletten Laden des Dokumentes
DOM wird geändert

.mergeAttributes() alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen
Attribute sind: HTML
Events
Styles
ab IE 5.01 auch ID, NAME
Achtung: Diese Methode ist mir Vorsicht zu geniessen !!
DOM wird geändert

.normalize() Normalisierung des DOM zur Erreichung einer konsistenten Struktur
Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen

.releaseCapture() Maus-Überwachung ausschalten für ein Objekt
Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick,
onmouseover und onmouseout.
Hinweis: einschalten per Methode .setCapture()

.removeAttribute() entfernen eines per HTML erzeugten Attributes
Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute!
per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
DOM wird geändert

.removeAttributeNode() entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das
entfernte Attribut liefern
DOM wird geändert

.removeBehavior() per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem
Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie)
DOM wird geändert

.removeExpression() Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der
Form objekt.style.eigenschaft. dient.
Ausdruck muss mit der Methode .setExpression() gesetzt worden sein
DOM wird nicht geändert

.replaceAdjacentText() Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu
ersetzenden Text liefern
DOM wird nicht geändert

.scrollIntoView() Objekt derart scrollen, dass es im Fenster für User sichtbar wird
Objekt muss an sich schon renderbar sein

.setActive() Objekt für die Eventdurchreichung aktivieren
aber ohne es zu fokussieren
und ohne es scrollbar zu machen

.setAttribute() Wert von vorhandenem Attribut setzen
wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
DOM wird nur bei Erzeugung geändert

.setAttributeNode() Attribut einem Knoten zuweisen und Referenz liefern
DOM wird geändert

.setCapture() Maus-Überwachung einschalten für ein Objekt
Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick,
onmouseover und onmouseout.
ab IE 5.5
Hinweis: ausschalten per Methode .releaseCapture()

.setExpression() Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-
Eigenschaft als Objektreferenz der Form
objekt.style.eigenschaft.
dient
Ausdruck nur als Script kodierbar
DOM wird nicht geändert

.swapNode() Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
nur sichtbar wenn Endetag geparst
DOM wird geändert