

```

        Index = 0;
    }

    // Rechteck der aktuellen Zeile ermitteln unter Beachtung eines eventuellen Scrollens
    var PosRechts    = ZeigerAufTextRectangleCollection [Index].right + ID_Body.scrollLeft;
    var PosLinks    = ZeigerAufTextRectangleCollection [Index].left  + ID_Body.scrollLeft;
    var PosOben1    = ZeigerAufTextRectangleCollection [Index].top   + ID_Body.scrollTop;

    // und Div1 auf die Zeile positionieren, also Zeile einfärben
    ID_Div1.style.top    = PosOben1;
    ID_Div1.style.width = (PosRechts - PosLinks) - 5;
    ID_Div1.style.display = 'inline';

    // aktuelle Position des Rechteckes von DIV0 ermitteln unter Beachtung eines eventuellen
    // Scrollens
    PosRechts    = Zeiger.getBoundingClientRect().right + ID_Body.scrollLeft;
    PosLinks    = Zeiger.getBoundingClientRect().left  + ID_Body.scrollLeft;
    var PosOben2    = Zeiger.getBoundingClientRect().top   + ID_Body.scrollTop;

    // und Div2 überlagern positionieren
    ID_Div2.style.top    = PosOben2;
    ID_Div2.style.width = (PosRechts - PosLinks) - 5;
    ID_Div2.style.height = PosOben1 - PosOben2;

    // aber nur einfärben, wenn mindestens 1 Zeile bereits eingefärbt wurde
    if (Index > 0) { ID_Div2.style.display = 'inline'; }

    // Rectangle der nächsten Zeile einstellen
    Index++;
}
</SCRIPT>
</HEAD>
<BODY ID="ID_Body">
    <DIV ID="ID_Div0"
        onclick=" Anzeigen(this)"
    >
        klicke
    <BR>
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
    1234567890123456789012345678901234567890
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
    1234567890123456789012345678901234567890
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
    1234567890123456789012345678901234567890
    ABCDEFGHIJKLMNOPQRSTUVWXYZ
    1234567890123456789012345678901234567890
    </DIV>
    <DIV ID="ID_Div1"
        STYLE="position:absolute; left:5; top:400; z-index:-1; background-color:yellow; display:none"
    >
    </DIV>
    <DIV ID="ID_Div2"
        STYLE="position:absolute; left:5; top:400; z-index:-1; background-color:beige; display:none"
    >
    </DIV>
</BODY>

```

Zugriff:

Zeiger.eigenschaft

Eigenschaften:

sind les- und schreibbar

.bottom	untere Pixelposition des Rechteckes um ein Objekt
.left	linke Pixelposition des Rechteckes um ein Objekt
.right	rechte Pixelposition des Rechteckes um ein Objekt
.top	obere Pixelposition des Rechteckes um ein Objekt

Methoden:

keine

4.3.2.2.4.2.2. Verwaltung mehrerer HTML-Elemente gleicher oder verschiedener Arten im HTML-Dokument**4.3.2.2.4.2.2.1. allgemeine HTML-Element bezogene Verwaltung**

Nachfolgend werden Collectionen beschrieben, die der allgemeinen HTML-Element-bezogenen Verwaltung dienen.



4.3.2.2.4.2.2.1.1. *attribute Objekt des Internet Explorer (Attribute-Verwaltung für ein HTML-Element)*

Pseudo-Objekt als Prototyp eines Attributes (einer Eigenschaft) eines HTML-Objektes

Zugriff:**nur über** die Collection attributes als Feld aller Zeiger aller Attribute des Objektes

Beispiel 1:

```

<SCRIPT>
    function ShowAttribs(ZeigerAufObjekt)
    {
        var ZeigerAufFeld = ZeigerAufObjekt.attributes;

        for (var Index = 0; Index < ZeigerAufFeld.length; Index ++)
        {
            var ZeigerAufFeldElement = ZeigerAufFeld [Index];

            alert(
                ZeigerAufFeldElement.nodeName
                + '='
                + ZeigerAufFeldElement.nodeValue
                + '('
                + ZeigerAufFeldElement.specified
                + ')';
            );
        }
    }
</SCRIPT>

```

Beispiel 2:

```

<SCRIPT>
    function Anzeigen()
    {
        for(var Index=0; Index < ID_Liste.attributes.length; Index ++)
        {
            if(ID_Liste.attributes[Index].specified)
            {
                alert(
                    ID_Liste.attributes[Index].nodeName
                    + " = "
                    + ID_Liste.attributes[Index].nodeValue
                );
            }
        }
    }
</SCRIPT>
<UL onclick="Anzeigen()">
    <LI ID = "ID_Liste" ACCESSKEY = "L">Listen-Element
</UL>

```

Beispiel 3:

```

<HTML>
<HEAD>
<SCRIPT>
    document.expando = false; // für gesamtes Dokument abschalten
</SCRIPT>
</HEAD>
<BODY>
<DIV>
    <SPAN ID="ID_Span" UNSELECTABLE="on">
        Dieser Text ist <B>selektierbar !!<B>
    </SPAN>
</DIV>
</BODY>
</HTML>

```

Beispiel für option objekt eines select objektes:

```

<SCRIPT>
    function Anzeigen()
    {
        var Kette = "Auswahl = " + ID_select.options(ID_select.selectedIndex).value;
        alert(Kette);
    }
</SCRIPT>

```



```

<SELECT ID="ID_select" onchange = "Anzeigen()">
  <OPTION VALUE="1">Auswahl 1 </OPTION>
  <OPTION VALUE="2">Auswahl 2 </OPTION>
  <OPTION VALUE="3">Auswahl 3 </OPTION>
</SELECT>

```

Eigenschaften:

siehe Collection attributes

Methoden:

siehe Collection attributes

4.3.2.2.4.2.2.1.2. attributes Collection des Internet Explorer

Diese Collection sammelt die Zeiger aller Attribute eines HTML-Elementes

Feld der Objekt-Attribute-Referenzen

aber nicht Style-Attribute-Werte des Objektes, da diese in der Eigenschaft .cssText abgeleigt sind (ab IE 5.x)

Feld muss mit den Methoden extra gepflegt werden, da dies nicht automatisch mit der Modifikation von Attributen vollzogen wird.

Feldelement: ist attribute Objekt (siehe dort)

Syntax:

```

[ var FeldZeiger = ] object.attributes
[ var FeldElementZeiger = ] object.attributes[Index]

```

Index: Integer und ab 0
muss in [] kodiert sein

Beispiel 1:

```

<SCRIPT>
function ShowAttribs(ZeigerAufObjekt)
{
    var ZeigerAufFeld = ZeigerAufObjekt.attributes;

    for (var Index = 0; Index < ZeigerAufFeld.length; Index ++)
    {
        var ZeigerAufFeldElement = ZeigerAufFeld [Index];

        alert(
            ZeigerAufFeldElement.nodeName
            + '='
            + ZeigerAufFeldElement.nodeValue
            + '('
            + ZeigerAufFeldElement.specified
            + ')';
        );
    }
}
</SCRIPT>

```

Beispiel 2:

```

<SCRIPT>
function Anzeigen()
{
    for(var Index=0; Index < ID_Liste.attributes.length; Index ++)
    {
        if(ID_Liste.attributes[Index].specified)
        {
            alert(
                ID_Liste.attributes[Index].nodeName
                + " = "
                + ID_Liste.attributes[Index].nodeValue
            );
        }
    }
}
</SCRIPT>
<UL onclick="Anzeigen()">
  <LI ID = "ID_Liste" ACCESSKEY = "L">Listen-Element
</UL>

```

Beispiel 3:

```

<HTML>
<HEAD>
<SCRIPT>
    document.expando = false; // für gesamtes Dokument abschalten
</SCRIPT>
</HEAD>
<BODY>
<DIV>

```



```

<SPAN ID="ID_Span" UNSELECTABLE="on">
  Dieser Text ist <B>selektierbar !!<B>
</SPAN>
</DIV>
</BODY>
</HTML>

```

Beispiel für option objekt eines select objektes:

```

<SCRIPT>
  function Anzeigen()
  {
    var Kette = "Auswahl = " + ID_select.options(ID_select.selectedIndex).value;
    alert(Kette);
  }
</SCRIPT>
<SELECT ID="ID_select" onchange = "Anzeigen()">
  <OPTION VALUE="1">Auswahl 1 </OPTION>
  <OPTION VALUE="2">Auswahl 2 </OPTION>
  <OPTION VALUE="3">Auswahl 3 </OPTION>
</SELECT>

```

Eigenschaften:

.expando Wirksamkeit von Attributen ein/aus
 true ein, Default
 false aus

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
  document.expando = false; // für gesamtes Dokument abschalten
</SCRIPT>
</HEAD>
<BODY>
<DIV>
  <SPAN ID="ID_Span" UNSELECTABLE="on">
    Dieser Text ist <B>selektierbar !!<B>
  </SPAN>
</DIV>
</BODY>
</HTML>

```

.length Anzahl der Feldelemente also Feldlänge
 .specified prüfen ob Objekt Attribute hat
 true, so Attribute ist vorhanden
 false, so keine Attribute vorhanden

Beispiel:

```

<SCRIPT>
  function Anzeigen()
  {
    var FeldAllerAttribute = ID_List.attributes;
    alert(FeldAllerAttribute(0).nodeName);    // Knotenname

    for( var i=0; i< FeldAllerAttribute.length; i++)
    {
      // neues LI-Element als Knoten der Liste anhängen
      var NeuesListenElement =document.createElement("LI");
      ID_List.appendChild(NeuesListenElement);

      // Wert des neuen LI-Elementes ist Text, also Textknoten
      var WertNeuesListenElement = document.createTextNode(
        i
        + " "
        + FeldAllerAttribute (i).nodeName
        + " = "
        + FeldAllerAttribute (i).nodeValue
      );
      NeuesListenElement.appendChild(WertNeuesListenElement);

      // auf specified prüfen
      if(FeldAllerAttribute (i).nodeValue != null )
      {
        alert(    FeldAllerAttribute(i).nodeName

```



```

+ " specified: "
+ FeldAlleAttribute(i).specified // boolean
);
}
}

```

```

</SCRIPT>
<UL ID="ID_List" onclick = " Anzeigen()">
  <LI>Klick mich
</UL>

```

.value Wert eines Objekt-Attributes
Hinweis: Wert eines Elementes ist z.T. über das VALUE-Attribut definierbar

Beispiel

```

<SCRIPT>
function Anzeigen()
{
    var Kette = "Auswahl = " + ID_select.options(ID_select.selectedIndex).value;
    alert(Kette);
}
</SCRIPT>
<SELECT ID="ID_select" onchange = "Anzeigen()">
  <OPTION VALUE="1">Auswahl 1 </OPTION>
  <OPTION VALUE="2">Auswahl 2 </OPTION>
  <OPTION VALUE="3">Auswahl 3 </OPTION>
</SELECT>

```

Methoden:

.getNamedItem() Zeiger auf Attribut liefern anhand des Attributnamen (analog zu ID oder NAME-Attribut)
ab IE 6.x

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
function Init()
{
    var ZeigerAufFeld = ID_P.attributes;
    var ZeigerAufFeldElement = ZeigerAufFeld.getNamedItem("align");
    alert("ALIGN Attribut Wert = " + ZeigerAufFeldElement.value);
}
</SCRIPT>
</HEAD>
<BODY ONLOAD="Init()">
  <P ID="ID_P" ALIGN="center">Test</P>
</BODY>
</HTML>

```

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

Beispiel 1:

```

<SCRIPT LANGUAGE="JScript">
var ZeigerAufCollectionDocumentAll = document.all;

if (ZeigerAufObjekt!=null)
{
    for (i=0; i< ZeigerAufCollectionDocumentAll.length; i++)
    {alert(ZeigerAufCollectionDocumentAll.item(i).tagName);}
}
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
function Init()
{
    var ZeigerAufFeld = ID_P.attributes;
    for (Index = 0; Index < ZeigerAufFeld.length; Index ++)
    {
        var ZeigerAufFeldElement = ZeigerAufFeld.item(Index);
        // hier numerischer Index
        var AttributWertSpezifiziert = ZeigerAufFeldElement.specified;
    }
}

```



```

// true oder false
var KnotenName = ZeigerAufFeldElement.nodeName;
// String
var KnotenWert = ZeigerAufFeldElement.nodeValue;
alert(    "Knotenname = "
        + KnotenName
        + "   mit spezifiziert = "
        + AttributWertSpezifiziert
        + "   und Wert = "
        + KnotenWert
        );
    }
}
</SCRIPT>
</HEAD>
<BODY ONLOAD="Init()">
  <P ID="ID_P">Test</P>
</BODY>
</HTML>

```

`.removeNamedItem()` Attribut entfernen anhand Attributname (analog zu ID oder NAME-Attribut), wobei danach der Standard-Attributwert automatisch weiterverwendet wird (falls Standard vorhanden ist), und Zeiger auf gelöscht Attribut liefern
ab IE 6

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
  function Entfernen()
  {
    var ZeigerAufFeld = ID_Div.attributes;
    ZeigerAufFeld.removeNamedItem("TITLE");
  }
</SCRIPT>
</HEAD>
<BODY>
  <DIV onclick="Entfernen();" ID="ID_Div" TITLE="Tooltip-Text ">
    Klick um den Tooltip-Text zu entfernen
  </DIV>
</BODY>
</HTML>

```

`.setNamedItem()` Attribut hinzufügen anhand Zeiger auf Attribut wenn noch nicht im Feld vorhanden, so Anhängen an das Feldende wenn schon im Feld vorhanden, so überschreiben und Referenz auf das überschriebene Attribut (vor dem Überschreiben) liefern
Bsp: Attribut erzeugen `document.createAttribute("title");`
Hinweis: in HTML können Attributnamen groß oder klein geschrieben werden
ab IE 6

Beispiel:

```

<HTML>
<HEAD>
<SCRIPT>
  function Hinzufuegen()
  {
    // Attribut mit Wert erzeugen
    var ZeigerAufAttribut = document.createAttribute("title");
    ZeigerAufAttribut.value = "Tooltip-Text ";

    // Attribut als Feldelement anhängen
    var ZeigerAufFeld = ID_Div.attributes;
    ZeigerAufFeld.setNamedItem(ZeigerAufAttribut);
  }
</SCRIPT>
</HEAD>
<BODY onload="Hinzufuegen();">
  <DIV ID="ID_Div">Es wird ein Tooltip-Text hinzugefuegt</DIV>
</BODY>
</HTML>

```

4.3.2.2.4.2.2.1.3. *childNodes* Collection des Internet Explorer

Feld der Zeiger aller Kinder-Knoten eines Objektes, also Feld der Zeiger aller HTML-Knoten-Kinder **und** Textknoten-Kinder des Objektes
Collection dient zur Ermittlung **nur von Knoteneigenschaften laut DOM**, falls diese im Element implementiert sind:



Elementfolge NICHT laut HTML-Coding sondern laut DOM.

Element (Knoten) kann per HTML oder Methode `.createElement()` erzeugt worden sein (falls Methode erlaubt ist).
Diese Collection sammelt die Zeiger aller Kinderknoten eines HTML-Elementes.

Syntax:

```
[ var ZeigerAufFeld = ] object.childNodes
[ var ZeigerAufFeldElement = ] object.childNodes[Index]

object                Zeiger auf Elternobjekt

Index                Integer und ab 0
                    muss in [ ] kodiert sein

ZeigerAufFeldElement ist null, wenn Feldelement nicht vorhanden
```

Zugriff auf Element:

Je nach Art des Kind-Elementes stehen dem Kind Eigenschaften und Methoden zur Verfügung (siehe Objektbeschreibungen):

```
object.childNodes[Index].eigenschaft_des_kind
object.childNodes[Index].methode_des_kind
```

```
object    Zeiger auf Elternobjekt

Index    Integer und ab 0
        muss in [ ] kodiert sein
```

Beispiel 1:

```
<SCRIPT>
    var ZeigerAufFeld = ID_Body.childNodes;
</SCRIPT>
<BODY ID="ID_Body">
    <SPAN >Test </SPAN>
</BODY>
```

Beispiel 2:

```
// DIV erzeugen
var ZeigerAufDivKnoten = document.createElement("DIV");

// B-Tag im DIV erzeugen
var ZeigerAufBKnoten = document.createElement("B");
ZeigerAufDivKnoten.insertBefore(ZeigerAufBKnoten);

// erst jetzt das DIV in den BODY einfügen, also DIV sichtbar machen
document.body.insertBefore(ZeigerAufDivKnoten);

// Collection referenzieren
var ZeigerAufFeld = ZeigerAufDivKnoten.childNodes;
```

Beispiel 3:

```
<SCRIPT LANGUAGE="JScript">
    var ZeigerAufFeldAllerPtag = document.all.tags("P");
    if (ZeigerAufFeldAllerPtag!=null)
    {
        for (i=0; i< ZeigerAufFeldAllerPtag.length; i++)
        { ZeigerAufFeldAllerPtag [i].style.textDecoration="underline"; } // auch .item(i) kodierbar
    }
</SCRIPT>
```

Beispiel 4:

```
<SCRIPT>
    var ErstesKind_Index = 0; // Index ab 0
    var ErstesKind_Name = Liste.childNodes(ErstesKind_Index).nodeName;
    alert(ErstesKind_Name); // liefert den Tagnamen 'LI' von Listenelement 1
                          // Hinweis: Listenelement 1 ist der Wert des Kindes
</SCRIPT>
<BODY>
    <UL ID = "Liste">
        <LI> Listenelement 1
        <LI> Listenelement 2
        <LI> Listenelement 3
    </UL>
</BODY>
```



Beispiel 5:

```

<SCRIPT>
function KnotenWertAendern( ZeigerAufListe,
                           IndexVonListenelement, // immer ab 0
                           Zeichenkette           // Listenelement muss Text sein
                           )
{
    var ReturnWert=false; // Annahme: Änderung schlägt fehl

    // prüfen auf UL-Tag
    if (ZeigerAufListe.nodeName == "UL")
    {
        // Anzahl der Listenelemente holen
        var AnzahlListenelemente= ZeigerAufListe.childNodes.length;
                                                // immer ab 1

        // und Anzahl und Index prüfen
        if ( (AnzahlListenelemente > 0) // immer ab 1
            && (IndexVonListenelement >= 0) // immer ab 0
            && (IndexVonListenelement < AnzahlListenelemente)
                                                // Index ist zulässig zur Anzahl
            )
        {
            // Zeiger auf das Listenelement laut Index holen
            var ZeigerAufListenelement =
                ZeigerAufListe.childNodes[IndexVonListenelement];

            // existiert das Listenelement ?
            if (ZeigerAufListenelement)
            {
                // ZeigerAufListenelement ist nicht null

                // Listenelement ist Textelement ?
                if (ZeigerAufListenelement.nodeType == 3)
                {
                    ZeigerAufListenelement.nodeValue =
                        Zeichenkette;
                    ReturnWert =true;
                }
            }
        }
    }

    return ReturnWert;
}
</SCRIPT>
<UL ID="Liste" onclick=" KnotenWertAendern(this, 0, 'Listenelement Neu')">
    <LI>Listenelement alt
</UL>

```

Beispiel 6:

```

<SCRIPT>
function TextElementAendern()
{
    var ZeigerAufTextElement = document.createTextNode("Neuer Text");
    var ZeigerAufSpanInhalt = ID_Span.childNodes(0);
    ZeigerAufSpanInhalt.replaceNode(ZeigerAufTextElement);
}
</SCRIPT>
<SPAN ID = "ID_Span" onclick=" TextElementAendern()">
    Original Text
</SPAN>

```

Beispiel 7:

```

<SCRIPT>
function Anzeige()
{
    var ZeigerAufOnClickEventQuelle=event.srcElement;
    var Feld =
        ZeigerAufOnClickEventQuelle.parentElement.getElementsByTagName("LI");
    alert( "Anzahl LI : "
        + Feld.length
    );
}

```



```

        + "\nErster Eintrag: "
        + Feld [0].childNodes[0].nodeValue
    );
}
</SCRIPT>
<UL onclick="Anzeige()">
  <LI>Menuepunkt 1
  <UL>
    <LI> Menuepunkt 1.1
    <OL>
      <LI> Menuepunkt 1 1.1
      <LI> Menuepunkt 1 1.2
    </OL>
    <LI> Menuepunkt 1.2
    <LI> Menuepunkt 1.3
  </UL>
<LI> Menuepunkt 2
<UL>
  <LI> Menuepunkt 2.1
  <LI> Menuepunkt 2.3
</UL>
<LI> Menuepunkt 3
</UL>

```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge

Methoden:

.item()
Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!

Beispiel 1:

```

<SCRIPT LANGUAGE="JScript">
  var ZeigerAufCollectionDocumentAll = document.all;

  if (ZeigerAufObjekt!=null)
  {
    for (i=0; i< ZeigerAufCollectionDocumentAll.length; i++)
    {alert(ZeigerAufCollectionDocumentAll.item(i).tagName);}
  }
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
  function Init()
  {
    var ZeigerAufFeld = ID_P.attributes;
    for (Index = 0; Index < ZeigerAufFeld.length; Index ++)
    {
      var ZeigerAufFeldElement = ZeigerAufFeld.item(Index);
      // hier numerischer Index
      var AttributWertSpezifiziert = ZeigerAufFeldElement.specified;
      // true oder false
      var KnotenName = ZeigerAufFeldElement.nodeName;
      // String
      var KnotenWert = ZeigerAufFeldElement.nodeValue;
      alert(
        "Knotenname = "
        + KnotenName
        + " mit spezifiziert = "
        + AttributWertSpezifiziert
        + " und Wert = "
        + KnotenWert
      );
    }
  }
</SCRIPT>
</HEAD>
<BODY ONLOAD="Init()">
  <P ID="ID_P">Test</P>
</BODY>

```



```

    </HTML>
.urns()      Referenz auf Feld aller Elemente mit gemeinsamer URN liefern
  Beispiel:
  <SCRIPT LANGUAGE="JScript">
    var ZeigerAufFeldAllerURN1 coll = document.all.urns("URN1");
    var Text = "";

    if (ZeigerAufFeldAllerURN1 != null)
    {
      for (i=0; i< ZeigerAufFeldAllerURN1.length; i++)
      {Text += ZeigerAufFeldAllerURN1.item(i).id + ', ';}
      alert (Text);
    }
  </SCRIPT>

```

4.3.2.2.4.2.1.4. children Collection des Internet Explorer

Feld der Zeiger aller **HTML-Elemente-Kinder** eines Objektes
 Collection dient **auch** zur Ermittlung von Knoteneigenschaften, wenn diese im Element implementiert sind.
Elementefolge laut HTML-Coding und nicht laut DOM
 Element kann per HTML oder Methode .createElement() erzeugt worden sein (falls Methode erlaubt ist).
 Für das Objekt form.input image **muß** die children Collection verwendet werden.

Syntax:

```

[ var ZeigerAufFeld = ] object.children
[ var ZeigerAufFeldElement = ] object.children[Index [, SubIndex] ]

```

object		Zeiger auf Elternobjekt
Index	oder	Integer ab 0 String Name oder ID des Elementes laut ID-Attribut bzw. NAME-Attribut muß in [] kodiert sein
SubIndex		optional Integer Unterindex also Unterelement eines Elementes nur kodieren wenn Index ein String ist
ZeigerAufFeldElement		ist null, wenn Feldelement nicht vorhanden

Beispiel 1:

```

<SCRIPT>
  var ZeigerAufFeld = ID_Body.children;
</SCRIPT>
<BODY ID="ID_Body">
  <SPAN >Test </SPAN>
</BODY>

```

Beispiel 2:

```

<SCRIPT>
  function Loeschen()
  {
    ID_Span.children[0].clearAttributes();
  }
</SCRIPT>
<SPAN ID="ID_Span">
  <DIV ID="ID_Div"
  ATTRIBUTE1="true"
  ATTRIBUTE2="true"
  onclick="alert('click');"
  onmouseover="this.style.color='#0000FF';"
  onmouseout="this.style.color='#000000';"
  >
    Test eines<B>Div</B>Elementes.
  </DIV>
</SPAN>
<INPUT TYPE="button" VALUE=" Loeschen" onclick="Loeschen()">

```

Beispiel 3:

```

<SCRIPT>
  function Hinzufuegen()
  {
    var ZeigerAufLI = document.createElement("LI");
    ID_Liste.children(0).insertAdjacentElement("beforeBegin", ZeigerAufLI);
  }

```



```

        ZeigerAufLI.innerText = "Listeneintrag 0";
    }
</SCRIPT>
<BODY>
    <OL ID = "ID_Liste ">
        <LI>Listeneintrag 1</LI>
        <LI>Listeneintrag 2</LI>
        <LI>Listeneintrag 3</LI>
    </OL>
    <INPUT TYPE = "button" VALUE = " Hinzufuegen" onclick=" Hinzufuegen()">
</BODY>

```

Beispiel 4:

```

<SCRIPT>
    function Tauschen()
    {Liste.children(0).swapNode(Liste.children(1)); }
</SCRIPT>
<UL ID = Liste>
    <LI>Listeneintrag 1
    <LI>Listeneintrag 2
    <LI>Listeneintrag 3
    <LI>Listeneintrag 4
</UL>
<INPUT TYPE = button VALUE = "Tauschen" onclick = "Tauschen()">

```

Beispiel 5:

```

<HEAD>
<SCRIPT>
    function Entfernen()
    {
        // versuche den Text zu entfernen
        try
        {
            var KindZeigerAufTextImDiv = ID_Div.children(0);
            ID_Div.removeChild(KindZeigerAufTextImDiv);
            // Achtung: Der Text ist noch sichtbar !!!!
        }
        // oder fange das Ereignis des bereits entfernten Textes ein
        // und behandle das Ereignis
        catch(x)
        {
            alert(
                "Text wurde entfernt !\n"
                + "Das Dokument muss neu geladen werden !
            );
            document.location.reload();
        }
    }
</SCRIPT>
</HEAD>
<BODY>
    <DIV ID="ID_Div" onclick=" Entfernen()">
        Klick, um diesen Text zu entfernen !
    </DIV>
</BODY>

```

Beispiel 6:

```

<HEAD>
<SCRIPT>
    function Ersetze()
    {
        var KindZeigerAufDivText = ID_Div.children(0);
        var RetteInnerHTML = KindZeigerAufDivText.innerHTML;

        // prüfen auf Tag im Div-Text
        if (KindZeigerAufDivText.tagName=="B")
        {
            // Bold-Tag <B>gefunden, also I-Tag erzeugen
            var ZeigerAufNeuenSchriftStilTag =document.createElement("I");

            // komplettes ersetzen von Div-Text,
            ID_Div.replaceChild(ZeigerAufNeuenSchriftStilTag, KindZeigerAufDivText);
        }
    }

```



```

        ZeigerAufNeuenSchriftStilTag.innerHTML= RetteInnerHTML;
    }
    else
    {
        // keinen Bold-Tag <B>gefunden
        var ZeigerAufNeuenSchriftStilTag =document.createElement("B");

        // komplettes ersetzen von Div-Text,
        ID_Div.replaceChild(ZeigerAufNeuenSchriftStilTag, KindZeigerAufDivText);
        ZeigerAufNeuenSchriftStilTag.innerHTML= RetteInnerHTML;
    }
}
</SCRIPT>
</HEAD>
<BODY>
    <DIV ID="ID_Div" onclick="Ersetze()">
        Klicke für den Wechseln des <B>Schriftstils<B>
    </DIV>
</BODY>

```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!

Beispiel 1:

```

<SCRIPT LANGUAGE="JScript">
    var ZeigerAufCollectionDocumentAll = document.all;

    if (ZeigerAufObjekt!=null)
    {
        for (i=0; i< ZeigerAufCollectionDocumentAll.length; i++)
        {alert(ZeigerAufCollectionDocumentAll.item(i).tagName);}
    }
</SCRIPT>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
    function Init()
    {
        var ZeigerAufFeld = ID_P.attributes;
        for (Index = 0; Index < ZeigerAufFeld.length; Index ++)
        {
            var ZeigerAufFeldElement = ZeigerAufFeld.item(Index);
            // hier numerischer Index
            var AttributWertSpezifiziert = ZeigerAufFeldElement.specified;
            // true oder false
            var KnotenName = ZeigerAufFeldElement.nodeName;
            // String
            var KnotenWert = ZeigerAufFeldElement.nodeValue;
            alert(
                "Knotenname = "
                + KnotenName
                + " mit spezifiziert = "
                + AttributWertSpezifiziert
                + " und Wert = "
                + KnotenWert
            );
        }
    }
</SCRIPT>
</HEAD>
<BODY ONLOAD="Init()">
    <P ID="ID_P">Test</P>
</BODY>
</HTML>

```

.tags() Referenz auf Feld aller HTML-Elemente mit gemeinsamen Tag-Namen liefern siehe tags Collection des DOM

Beispiel:



```

<SCRIPT LANGUAGE="JScript">
  var ZeigerAufFeldAllerPTag = document.all.tags("P");
  if (ZeigerAufFeldAllerPTag!=null)
  {
    for (i=0; i< ZeigerAufFeldAllerPTag.length; i++)
    { ZeigerAufFeldAllerPTag [i].style.textDecoration="underline";}
  }
</SCRIPT>

```

.urns() Referenz auf Feld aller Elemente mit gemeinsamer URN liefern

Beispiel:

```

<SCRIPT LANGUAGE="JScript">
  var ZeigerAufFeldAllerURN1 coll = document.all.urns("URN1");
  var Text = "";

  if (ZeigerAufFeldAllerURN1 != null)
  {
    for (i=0; i< ZeigerAufFeldAllerURN1.length; i++)
    {Text += ZeigerAufFeldAllerURN1.item(i).id + ', ';}
    alert (Text);
  }
</SCRIPT>

```

4.3.2.2.4.2.2.1.5. tags Collection des Internet Explorer

Diese Collection sammelt die Zeiger aller HTML-Elemente, die gemeinsamen HTML-Tag besitzen. wird nur von Collectionen und Objekten instanziiert, die die Methode .tags() besitzen

- z.B. childNodes Collection des DOM
- children Collection des DOM
- document.all Collection des DOM

Zugriff auf das Feld nur über die Methode .tags()

Syntax:

```
[ var Zeiger = ] zeiger_auf_collection_oder_objekt.tags(Kette)
```

Kette String mit HTML-Tag-Bezeichner

fZeiger weist auf ein leeres Feld, wenn keine HTML-Elemente mit dem Tag gefunden wurden

Beispiel:

```

<SCRIPT LANGUAGE="JScript">
  var ZeigerAufFeldAllerPTag = document.all.tags("P");
  if (ZeigerAufFeldAllerPTag!=null)
  {
    for (i=0; i< ZeigerAufFeldAllerPTag.length; i++)
    { ZeigerAufFeldAllerPTag [i].style.textDecoration="underline";}
  }
</SCRIPT>

```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge

Methoden:

keine

4.3.2.2.4.2.2.2. spezielle HTML-Element bezogene Verwaltung

4.3.2.2.4.2.2.2.1. Erzeugung ausgewählter eines HTML-Elemente durch Makro (command Objekt des Internet Explorer)

Objekt zum Verwalten von externen vordefinierten Kommandos des IE. Diese Kommandos sind eine andere Variante von Objekterzeugungen (z.B. von HTML-Elementen) als Analogon zu Makros nutzen das DOM

Das Objekt besitzt nur Methoden, die an andere Objekte vererbt werden.

Die meisten Methoden sind erst nach dem kompletten Laden des Dokumentes anwendbar.

Falls die Methoden Werte liefern, so sind die Werttypen kommandospezifisch.

Beispiel 1:

```

<HTML>
<BODY>
  <H1 UNSELECTABLE="on">Demo</H1>
  <SCRIPT>
    function AddLink()
    {
      var SelektierterText = document.selection.createRange();

      if (!SelektierterText == "")
      {

```



```

// Link erzeugen
document.execCommand("CreateLink");

if (SelektierterText.parentElement().tagName == "A")
{
    // markierten Text mit Eltern-Url ersetzen
    SelektierterText.parentElement().innerText=
        SelektierterText.parentElement().href;

    // Vordergrundfarbe setzen im Dokument
    document.execCommand(
        "ForeColor", "false", "#FF0033");
}
}
else
{alert("Bitte im blauen Text selektieren !");}
}
</SCRIPT>
<P UNSELECTABLE="on">
    Selektiere (markiere) im nachfolgenden blauen Text die Stelle mit
    dem Text MARKIERE_MICH.<BR>
    Danach auf das Button klicken.<BR>
    Anstelle von MARKIERE_MICH im blauen Text erscheint dort
    nun eine Url.
</P>
<P STYLE="color=#3366CC">
    Meine beliebteste Webseite MARKIERE_MICH bitte besuchen !
</P>
<BUTTON onclick="AddLink()" UNSELECTABLE="on">Klick</BUTTON>
</BODY>
</HTML>

```

Beispiel 2:

```

<HTML>
<HEAD>
<SCRIPT>
function HandlerFuerOnMoveStart()
{
    // anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
    var ZeigerAufObjektMitEvent = event.srcElement;

    ZeigerAufObjektMitEvent.style.backgroundColor = "green";
    ZeigerAufObjektMitEvent.innerText = "DIV wird bewegt ";
}

function HandlerFuerOnMove ()
{
    ID_Span1.innerHTML = event.srcElement.offsetLeft;
    ID_Span2.innerHTML = event.srcElement.offsetTop;
}

function HandlerFuerOnMoveEnd()
{
    // anstelle des ID vom DIV falls mehrere bewegbare Objekte vorhanden sind
    var ZeigerAufObjektMitEvent = event.srcElement;

    ZeigerAufObjektMitEvent.style.backgroundColor = "red";
    ZeigerAufObjektMitEvent.innerText = "DIV wird nicht mehr bewegt";
}

// 2-D Positionierung einschalten
document.execCommand("2D-position", false, true);
</SCRIPT>
</HEAD>
<BODY onmovestart="HandlerFuerOnMoveStart();"
onmove="HandlerFuerOnMove();"
onmoveend="HandlerFuerOnMoveEnd();"
>
    offsetLeft = <SPAN ID="ID_Span1"></SPAN>
    <BR>

```



```

offserTop = <SPAN ID="ID_Span2"></SPAN>
<BR>
<DIV CONTENTEDITABLE="true">
  <DIV STYLE=
    "position:absolute;width:300px;height:100px; background-color:red;"
  >
    bewegbarer DIV
  </DIV>
</DIV>
</BODY>
</HTML>

```

Eigenschaften:

keine

Methoden:

.execCommand()

Kommando ausführen z.B. im aktuellen Dokument
 in aktueller Selektion
 im aktuellen Bereich
 erst nach dem kompletten Laden des Dokumentes zulässig
 Hinweis: Selektion = Markierung z.B. von Textbereich (Block)
 Control = Element zur Steuerung analog zum HTML-Element (Tag)
 Input-Control = Element mit Eingabeeigenschaft

Syntax:

```
var Wert = object.execCommand(Command [, UserInterface] [, Value])
```

Command String als Kommando, wobei Gross-Kleinschreibung egal ist

"2D-Position"	absolute Positionierung von Elementen durch Bewegen im Dragging erlauben
"AbsolutePosition"	Element-Eigenschaft der Position auf "absolute" setzen nur für selektierbare Elemente nicht für STYLE-Deklarationen im Dokument
"BackColor"	lesen oder setzen der Hintergrundfarbe der aktuellen Selektion
"Bold"	Wechsel zwischen bold und nonbold in der aktuellen Selektion
"Copy"	Aktuelle Selektion in das Clipboard kopieren
"CreateBookmark"	Bookmark setzen oder lesen für aktuelle Selektion bzw. Einfügekpunkt
"CreateLink"	Hyperlink in der aktuellen Selektion setzen oder einfügen bei Einfügen erscheint Dialog-Box
"Cut"	Aktuelle Selektion in das Clipboard verschieben
"Delete"	Aktuelle Selektion löschen Hinweis: Dokument nicht löschar
"FontName"	Font für aktuelle Selektion setzen oder holen
"FontSize"	Fontsize für aktuelle Selektion setzen oder holen
"ForeColor"	Vordergrundfarbe (Textfarbe) für aktuelle Selektion setzen oder holen
"FormatBlock"	Blockformat setzen
"Indent"	Ident des selektierten Textes erhöhen
"InsertButton"	in Textselektion das Button-Control einfügen, wenn eines bereits vorhanden so überschreiben
"InsertFieldset"	in Textselektion die Box einfügen, wenn eine bereits vorhanden so überschreiben
"InsertHorizontalRule"	in Textselektion die horizontale Linie einfügen wenn eine bereits vorhanden so überschreiben
"InsertIFrame"	in Textselektion den inline-frame (IFRAME) einfügen wenn einer bereits vorhanden so überschreiben
"InsertImage"	in Textselektion das Image einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputButton"	in Textselektion das Input-Button-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputCheckbox"	in Textselektion das Input-Check-Box-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputFileUpload"	in Textselektion das Input-File-Upload-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputHidden"	in Textselektion das Input-Hidden-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputImage"	in Textselektion das Input-Image-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputPassword"	



		in Textselektion das Input-Password-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputRadio"		in Textselektion das Input-Radio-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputReset"		in Textselektion das Input-Reset-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputSubmit"		in Textselektion das Input-Submit-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertInputText"		in Textselektion das Input-Text-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertMarquee"		in Textselektion das Marquee einfügen wenn eines bereits vorhanden so überschreiben
"InsertOrderedList"		Wechsel zwischen ordered list und normalen Block für aktuelle Textselektion
"InsertParagraph"		in Textselektion Zeilenumbruch einfügen wenn eines bereits vorhanden so überschreiben
"InsertSelectDropdown"		in Textselektion das Drop-Down-Selections-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertSelectListbox"		in Textselektion die List-Box-Selektion einfügen wenn eines bereits vorhanden so überschreiben
"InsertTextArea"		in Textselektion das Input-Textarea-Control einfügen wenn eines bereits vorhanden so überschreiben
"InsertUnorderedList"		Wechsel zwischen unordered list und normalen Block für aktuelle Textselektion
"Italic"		Wechsel zwischen italic und non-italic für aktuelle Textselektion
"JustifyCenter"		zentrieren für aktuelle Textselektion
"JustifyLeft"		linksbündig für aktuelle Textselektion
"JustifyRight"		rechtsbündig für aktuelle Textselektion
"MultipleSelection"		Mehrfachselektion per CTRL+ bzw. Shift + erlauben
"Outdent"		Outdent des selektierten Textes erniedrigen
"OverWrite"		Wechsel zwischen überschreiben und nicht überschreiben
"Paste"		Aktuelle Selektion aus Clipboard überschreiben
"Print"		Druck-Dialogbox öffnen
"Refresh"		aktuelles Dokument refreshen
"RemoveFormat"		formatierende Tags der aktuellen Selektion entfernen
"SaveAs"		Aktuelles Dokument speichern als Datei
"SelectAll"		alles markieren (selektieren)
"UnBookmark"		Alle Bookmark der aktuellen Selektion entfernen
"Underline"		Wechsel zwischen underline und nicht-underline für aktuelle Textselektion
"Unlink"		Alle Hyperlink der aktuellen Selektion entfernen
"Unselect"		Alles demarkieren (de-selektieren)
UserInterface	false	Default user interface soll nicht angezeigt werden (Dialogbox)
	true	user interface soll angezeigt werden (falls Kommando das unterstützt)
Value		z.B. String, number immer passend zu Command, kann optional sein Kodierung null (nicht numerisch Null !!) als Wert entspricht Weglassen des optionalen Wertes

Kommando	IE ab	Dialogbox	Value
2D-Position	5.5	nein	true für on false für off
AbsolutePosition	5.5	nein	true für "absolut" false für nicht"absolut"
BackColor	4.x	nein	rrggbb OHNE führendes # vordefinierter Farbname (browserspezifisch)
Bold	4.x	nein	null oder omit oder weglassen
Copy	4.x	nein	null oder omit oder weglassen
CreateBookmark	4.x	nein	String mit Ankername keine Leerkette
CreateLink	4.x	ja	String mit Url



Cut	4.x	nein	keine Leerkette
Delete	4.x	nein	null oder omit oder weglassen
FontName	4.x	nein	null oder omit oder weglassen String mit Fontname oder Fontliste Fontliste: Folge von Fonteigenschaften
FontSize	4.x	nein	String mit Fontsize von einschliesslich 1 bis einschliesslich 7
ForeColor	4.x	nein	rrggbb OHNE führendes # vordefinierter Farbname (browserspezifisch)
FormatBlock	4.x	nein	String mit Block-Tag
Indent	4.x	nein	null oder omit oder weglassen
InsertButton	4.x	nein	String mit Attributen des Button-Control
InsertFieldset	4.x	nein	String mit Attributen der Box
InsertHorizontalRule	4.x	nein	String mit Attributen der Linie
InsertIFrame	4.x	nein	String mit Attributen des IFRAME
InsertImage	5.x	ja	String mit Pfad und Dateiname
InsertInputButton	4.x	nein	String mit Attributen des Input- Button-Control
InsertInputCheckbox	4.x	nein	String mit Attributen des Input- Checkbox-Control
InsertInputFileUpload	4.x	nein !!!	String mit Attributen des Input- Fileupload-Control
InsertInputHidden	4.x	nein	String mit Attributen des Input- Hidden-Control
InsertInputImage	4.x	nein	String mit Attributen des Input- Image-Control
InsertInputPassword	4.x	nein	String mit Attributen des Input- Password-Control
InsertInputRadio	4.x	nein	String mit Attributen des Input- Radio-Control
Kommando	IE ab	Dialogbox Value	
InsertInputReset	4.x	nein	String mit Attributen des Input- Reset-Control
InsertInputSubmit	4.x	nein	String mit Attributen des Input- Submit-Control
InsertInputText	4.x	nein	String mit Attributen des Input- Text-Control
InsertMarquee	4.x	nein	String mit Attributen des Marquee-Control
InsertOrderedList	4.x	nein	String mit Attributen des Ordered-List-Control
InsertParagraph	4.x	nein	String mit Attributen des Paragraph-Control
InsertSelectDropdown	4.x	nein	String mit Attributen des Drop- Down-Control
InsertSelectListbox	4.x	nein	String mit Attributen des List- Box-Control
InsertTextArea	4.x	nein	String mit Attributen des Text- Area-Control
InsertUnorderedList	4.x	nein	String mit Attributen des Unordered-List-Control
Italic	4.x	nein	null oder omit oder weglassen
JustifyCenter	4.x	nein	null oder omit oder weglassen
JustifyLeft	4.x	nein	null oder omit oder weglassen
JustifyRight	4.x	nein	null oder omit oder weglassen
MultipleSelection	5.5	nein	true für on false für off
Outdent	4.x	nein	null oder omit oder weglassen
OverWrite	4.x	nein	true für Überschreiben false für Nicht-Überschreiben
Paste	4.x	nein	null oder omit oder weglassen
Print	5.5	ja	null oder omit oder weglassen
Refresh	4.x	nein	kein String !!! null oder omit oder weglassen



RemoveFormat	4.x	nein	null oder omit oder weglassen
SaveAs	4.x	ja	wenn Dialogbox anzeigen so kann Wert null sein wenn Dialogbox nicht anzeigen so muss Wert die Paramter enthalten
SelectAll	4.x	nein	null oder omit oder weglassen
UnBookmark	4.x	nein	null oder omit oder weglassen
Underline	4.x	nein	null oder omit oder weglassen
Unlink	4.x	nein	null oder omit oder weglassen
Unselect	4.x	nein	null oder omit oder weglassen
Wert	true	wenn Kommando ausgeführt wurde	
	false	wenn Kommando nicht ausgeführt wurde	

- .queryCommandEnabled() prüfen ob Kommando ausführbar ist
- .queryCommandIndeterm() prüfen ob Kommando-Status bestimmbar ist oder nicht
- .queryCommandState() Status des aktuellen Kommando ermitteln: ob ausgeführt wurde oder nicht
- .queryCommandSupported() prüfen ob Kommando im aktuellen Bereich unterstützt wird
- .queryCommandValue() Wert eines Kommandos liefern

4.3.2.2.4.2.2.2.2. Erzeugung eines privaten HTML-Elementes (custom Objekt des Internet Explorer)

Objekt, das ein durch den Programmierer (Customer) frei definiertes HTML-Element (Customer-Tag) repräsentiert auf einem XML-Namensraum (XMLNS) basiert
nur verwendet wird für Behavior-Definition per *.htc-Datei (nicht Standard-Behavior des IE)
globale Style-Definition im HEAD

ab IE 5.x
siehe Objekt style

Syntax:

```
<HTML namens_raum_liste>
<HEAD>
  <STYLE>
    @media all {
      namen_raum_listen_element_1\:freier_style_name_1 { liste_1 }
      .....
      namen_raum_listen_element_n\:freier_style_name_n { liste_n.}
    }
  </STYLE>
</HEAD>
<BODY>

</BODY>
</HTML>
```

<HTML namens_raum_liste> mit Leerzeichen getrennte Elemente
Element mit Aufbau

XMLNS:freier_tag [= wert]

XMLNS: festkodiert

freier_tag wie normaler HTML-Tag aber frei definiert
Empfehlung: Gross-Schreibung

wert von freier_tag
optional
String

@media all { } Pflichtkodierung

namens_raum_liste_element Der Doppelpunkt muss entwertet werden, also \: kodieren, da dieser bereits ein Trenner laut Syntax für Style-Eigenschaften in liste ist.

freier_style_name Bezeichner eines globalen Style zum Customer-Tag
Empfehlung: Gross-Schreibung

liste per Semikolon getrennte Style-Eigenschaften wie beim STYLE-Attribut, also den Trenner Doppelpunkt **nicht** entwerten !



Kodierung innerhalb BODY:

```

<freier_tag:freier_style_name>
.....
</freier_tag:freier_style_name>

```

Beispiel 1:

```

<HTML XMLNS:MY1
XMLNS:MY2="www.test.de"
>
.....

```

Beispiel 2:

```

<HTML XMLNS:MY>
<HEAD>
<STYLE>
    @media all {
        MY:JUSTIFY { text-align:justify; width:100 }
    }
</STYLE>
</HEAD>
<BODY>
<MY:JUSTIFY>
    Text mit Style laut userdefiniertem MY:JUSTIFY-Tag
</MY:JUSTIFY>
</BODY>
</HTML>

```

Beispiel 3:

```

<HTML XMLNS:MY1
XMLNS:MY2
XMLNS:MY3
>
<HEAD>
<STYLE>
    @media all {
        MY1:TEXT_FARBE_ROT { color: red; }
        MY2:TEXT_FARBE_GRUEN { color: green; }
        MY3:TEXT_FARBE_BLAU { color: blue; }
    }
</STYLE>
</HEAD>
<BODY>
    <MY1:TEXT_FARBE_ROT>
        Text1
    </MY1:TEXT_FARBE_ROT>

    <MY2:TEXT_FARBE_GRUEN>
        Text2
    </MY2:TEXT_FARBE_GRUEN>

    <MY3:TEXT_FARBE_BLAU>
        Text3
    </MY3:TEXT_FARBE_BLAU>
</BODY>
</HTML>

```

Hinweis: Da es sich in allen Styles immer um eine Textfarbe handelt und sich die freien Style-Bezeichner unterscheiden, hätte auch kodiert werden können:

```

<HTML XMLNS:MY>
....
    @media all {
        MY:TEXT_FARBE_ROT { color: red; }
        MY:TEXT_FARBE_GRUEN { color: green; }
        MY:TEXT_FARBE_BLAU { color: blue; }
    }
....
</BODY>

```



```

<MY:TEXT_FARBE_ROT>
  Text1
</MY:TEXT_FARBE_ROT>

<MY:TEXT_FARBE_GRUEN>
  Text2
</MY:TEXT_FARBE_GRUEN>

<MY:TEXT_FARBE_BLAU>
  Text3
</MY:TEXT_FARBE_BLAU>
</BODY>

```

Beispiel 4:

```

<HTML XMLNS:MY >
<HEAD>
<STYLE>
  @media all {
    MY\HTC_DATEI { behavior: url(test.htc); }
  }
</STYLE>
</HEAD>
<BODY>
  <MY:HTC_DATEI>
  .....
  </MY:HTC_DATEI>
</BODY>
</HTML>

```

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.blockDirection	Umfluss um ein Objekt "ltr" Umfluss von links nach rechts "rtl" Umfluss von rechts nach links
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.hideFocus	Focussierbarkeit true Focus ist nicht möglich false Default Focus ist möglich
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetags
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes,



	also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEditable	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeight, .offsetLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup
	Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA
	Anspringen default nicht per TAB-Taste für APPLETT, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.title	Tooltip-Text bei Mouse over über Objekt
UNSELECTABLE	Selektionsfähigkeit eines Objektes
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler



	Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar
.click()	DOM wird geändert simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_ bezeichnet zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.doScroll()	Klick auf Scrollbar simulieren Achtung: Scrollelemente müssen bereits vorhanden sein ! "scrollbarDown" oder "down" Down scroll Pfeil "scrollbarHThumb" horizontaler Scrollbalken "scrollbarLeft" oder "left" Left scroll Pfeil "scrollbarPageDown" oder "pageDown" Page-down Scrollbalken "scrollbarPageLeft" oder "pageLeft" Page-left Scrollbalken "scrollbarPageRight" oder "pageRight" Page-right Scrollbalken "scrollbarPageUp" oder "pageUp" Page-up Scrollbalken "scrollbarRight" oder "right" Right scroll Pfeil "scrollbarUp" oder "up" Up scroll Pfeil "scrollbarVThumb" vertikaler Scrollbalken
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementsByName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdrucks berechnen und liefern Style-Eigenschaft ist per Methoden



	expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)
.insertAdjacentHTML()	HTML-Code und/oder Script-Code in ein Element einfügen, wobei die Lage definiert sein kann nur nach dem kompletten Laden des Dokumentes möglich HTML- und Script-Code müssen syntaktisch korrekt sein wenn nicht, so wird das Einfügen nicht ausgeführt eingefügter Code wird nur dann sofort geparkt und ausgeführt, wenn syntaktisch korrekt ist bei Script-Code: <SCRIPT DEFER> muss kodiert werden DOM wird geändert
.insertAdjacentText()	Plain-Text (ohne HTML und Script) in ein Element einfügen, wobei die Lage definiert werden kann nur nach dem kompletten Laden des Dokumentes DOM wird geändert
.mergeAttributes()	alle Attribute eines Elementes in ein anderes Element kopieren und eventuell die Attribute im Ziel mischen Attribute sind: HTML Events Styles ab IE 5.01 auch ID, NAME Achtung: Diese Methode ist mir Vorsicht zu geniessen !! DOM wird geändert
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.releaseCapture()	Maus-Überwachung ausschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. Hinweis: einschalten per Methode .setCapture()
.removeAttribute()	entfernen eines per HTML erzeugten Attributes Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute! per Methode .createAttribute() erzeugte Attribute werden nicht erfasst DOM wird geändert
.removeAttributeNode()	entfernen von Attribut, egal ob es mit oder ohne HTML-Anweisung erzeugt wurde, und Referenz auf das entfernte Attribut liefern DOM wird geändert
.removeBehavior()	per Methode .addBehavior() einem Element hinzugefügte Verhaltenseigenschaft entfernen (stets VOR dem Entfernen des Elementes mit der zugeordneten Eigenschaft aus der Dokument-Hierarchie) DOM wird geändert
.removeExpression()	Ausdruck entfernen, der für die Berechnung des Wertes einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient. Ausdruck muss mit der Methode .setExpression() gesetzt worden sein DOM wird nicht geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern DOM wird nicht geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout. ab IE 5.5 Hinweis: ausschalten per Methode .releaseCapture()
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style- Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient Ausdruck nur als Script kodierbar DOM wird nicht geändert

4.3.2.2.4.2.2.3. Cookie-Verwaltung im HTML-Dokument (document.cookie Collection des Internet Explorer)

4.3.2.2.4.2.2.3.1. Zweck von Cookies

Cookies dienen dem Zwischenspeichern von Daten auf der Festplatte des Client (Users)
auch über beliebig viele Online-Sitzungen hinaus
auch permanent endlos (Cookie ohne Verfallsdatum)



können von einem oder mehreren Servern geschrieben und gelesen werden auf/von der Festplatte des Users
z.B. für Speicherung von Benutzerinfos,
Passwörtern,
Umgebungseinstellungen,
Formulardaten per Cookies

der Verfolgung der Surfgewohnheiten des Users dienen
von lokal ablaufenden Webseiten (ohne Internetzugriff) verwendet werden
müssen nicht Bestandteil einer Webseite sein, da sie kein Bestandteil von HTML sind
haben vorrangig einen Nutzen, der im Interesse des Webseiten-Anbieters und weniger im Interesse des User liegt.

Cookies können, wenn sie als Voraussetzung für die Lauffähigkeit einer Webseite verwendet werden, für den User eine Nötigung darstellen:
Eine Nötigung liegt nicht vor, solange der User die Wahl hat, die Webseite besuchen zu können bzw. der User in die vollendete Lage versetzt ist, die Cookieverwaltung der Webseite eigenständig und damit die Wirksamkeit von Cookies vorab unterbinden zu können.
Dagegen ist das automatische Aufpoppen einer Webseite mit Cookieverwaltung pure Nötigung, wenn der User das Aufpoppen bzw. die Cookieverwaltung objektiv nicht verhindern kann.

Die Zielsetzung der Cookie-Verwendung ist im Regelfall durch den User nicht ersichtlich, da Cookies zwar ein einheitliches Format besitzen, aber beliebige Inhalte, die dem User nicht unbedingt vorab zur Cookieverwendung zur Kenntnis gelangen. Als Missbrauch sind Cookies einzuschätzen, die kein Verfallsdatum besitzen und/oder ohne Kenntnis des Users permanent auf der Festplatte existieren.

Die Kenntnisse des Users, weshalb und in welcher Form Cookies verwendet werden, ist Sache des Users, wenn der Hersteller der Browsersoftware bzw. der Hersteller der Webseite keine Aufklärung des Users zu den verwendeten Cookies betreibt. Das Risiko der Cookieverwendung liegt beim User, wenn der User nicht in die vollendete Lage versetzt ist, sich mit für ihn üblichen und seiner Kenntnis zugänglichen Mitteln gegen die Cookie-Verwaltung zu wehren. Die Verhältnismäßigkeit der Cookie-Verwendung zum Surfzweck des Users lässt sich z.T. anzweifeln. Im letzteren Fall ist die Cookieverwendung eine aggressives Verhalten des Webseitenanbieters gegen über dem User und dessen Eigentum, wenn für den User ein Risiko auf Erleiden von Schaden besteht und der User das Risiko sowie den Schaden im Voraus nicht abschätzen kann. Cookieverwendung ohne Transparenz für den User kann aus Usersicht als Risiko und als unerwünscht eingestuft werden
als Eingriff in die Privatsphäre und in das Datenschutzbedürfnis des Users aufgefasst werden
rechtlich relevant werden.

Der User sollte wissen, ob und wie der Browser Cookies zulässt. Es besteht die Gefahr, dass der User aufgrund der weit verbreiteten Cookie-Nutzung von Webseitenanbietern die Cookieverwendung generell zulässt, um nervende Meldungen oder eventuelle Surf-Misserfolge bei ohne Cookies nicht lauffähigen Seiten zu vermeiden. Der Einsatz einer Firewall erfordert zusätzliche Massnahmen und Kenntnisse des Users, um den Cookiezugriff steuern zu können, wenn die Firewall-Software nicht Bestandteil des Software ist, die der User beim Surfen nutzt, bzw. die Firewall-Software unverhältnismäßig kompliziert zum Surfzweck des Users ist. Es ist dem User anzuraten, Cookies nach einer Online-Sitzung auf der Festplatte physisch zu löschen bzw. Cookies über Browsereinstellungen und/oder über eine Firewall zu steuern (Cookiefreigabe vorab nur für durch den User bewusst ausgewählte Webseiten).

Ein Analogon zur Bedeutung der Cookie-Nutzung ist die Verlinkung auf fremde Webseiten-Angebote:

Die **pauschale** Erklärung des Webseitenanbieters zur ausdrücklichen Distanzierung von Inhalten zu verlinkten Fremdanbieterseiten (und damit auch von der möglichen Cookieverwaltung durch Fremdanbieter) ist inzwischen eine **veraltete** Rechtsnorm geworden und sieht beispielhaft wie folgt aus:

"Für alle Links dieser gesamten Homepage gilt: Ich möchte ausdrücklich betonen, daß ich keinerlei Einfluß auf die Gestaltung und die Inhalte der gelinkten Seiten habe. Deshalb distanziere ich mich hiermit ausdrücklich von allen Inhalten aller von dieser Homepage aus gelinkten Seiten."

Dabei ist es unerheblich, welcher Fremdanbieter (z.B. eine Partner-Webseite) betroffen ist.

Es **kann** aber der Fall eintreten, dass die pauschale Distanzierung nichts anderes als eine leider übliche Form des Weiterreichens des Risikos an den User und unabhängig davon ist, welche realen Voraussetzungen sich zum Zeitpunkt des Webseitenbesuches durch den User vorfinden lassen würden bzw. vorgefunden werden bzw. wurden. Die Zweizügigkeit und Praxisferne der pauschalen Distanzierung lässt sich salopp auch so sagen: Irren ist menschlich, vorallem für den Endverbraucher in der Kette, den User.

Achtung: Zu prüfen ist auch immer, ob überhaupt eine Verlinkung generell oder im Einzelfall stattfinden darf, da Rechtsnormen **permanent** und nicht unbedingt logisch oder gemäß den üblichen Gepflogenheiten im Web verändert werden: Missbräuchliche Webnutzung ist nicht der Regelfall, kann aber in der Rechtspflege als schwerwiegende, reale oder wahrscheinliche Ausnahme unterstellt werden, die eine juristische Regulierung des **Regelfalls** herbeiführt.

Aufgrund der Uneinheitlichkeit der Rechtspflege ist der Inhaber einer Webseite im unangenehmen Zugzwang: Es bestehen eventuell Interessenskonflikte zwischen dem Anliegen des Webseiteninhabers und den Rechtsnormen (aus Sicht der den Regelfall dominierenden Ausnahme des Regelfalles). Desweiteren ist davon auszugehen, dass private Webseitenanbieter ohne kommerzielle Interessen mit der Verfolgung der Rechtsnormenveränderung überfordert sind und dann das Risiko der Veröffentlichung der Webseite(n) nicht wissen und trotzdem rechtskonform auftreten **müssen**.

Die einfachste Lösung, das Rechtsproblem zu umgehen, ist die Darstellung des Links ohne Klickbarkeit, also ohne seine Aktivierung durch User-Klick auf das Link-Element des Dokumentes in der Webseite (z.B. anstelle der Link-Aktivierung ein



Popup-Fenster anzeigen, das den Link kommentiert, aber nicht aktiviert, also einen Link-Tip offeriert).

Wenn der User die Url selbst aktiviert, z.B. durch Eingabe in der Kommandozeile des Browsers, ist der User verantwortlich. Problem dabei: Konformität des im Link-Tip angegebenen Ziels zu den Rechtsnormen und Kenntnis des User darüber (Ausschluss der Irreführung/Täuschung des Users).

Tip: Die Realisierung eines Gästebuches unterliegt gleichen Kriterien, wenn der Gästebucheintrag durch den Inhaber des Gästebuches unbesehen, weil automatisch, im Gästebuch publiziert wird und der Inhaber des Gästebuches danach Kenntnis zu diesem Eintrag erlangt, der z.B. einen durch den Leser des Gästebuches aktivierbaren Link enthält.

Die Nutzung von Cookies sollte bezüglich Zweck, Dauer und Inhalt dem User mitgeteilt werden und nachvollziehbar sein. Erfahrungsgemäß ist das **nicht** der Regelfall. Auch die Möglichkeit der Browsersoftware, die Cookie-Verwaltung transparent zu halten, entlastet nicht. Der Regelfall ist, dass der User entweder von Cookies keine Kenntnis oder Cookies zum ungestörten Surfen genehmigt hat. Damit tritt nur selten der Fall ein, dass eine vom Webseiten-Inhaber programmierte Meldung zur Aktivierung der abgeschalteten Cookieverwaltung und zum Grund der Cookienutzung auftreten kann. Zumal es für den User eine Zumutung ist, die vom Webseiteninhaber gewollte Cookieverwendung nachvollziehen zu können/müssen, wenn der User die Webseite, die Cookies verwendet, besuchen will. Nochmals sie darauf hingewiesen: Der User trägt das eventuelle Risiko und den Aufwand, die der Webseitenanbieter bewirken könnte.

Hinweis: Als zwingend rechtlich geboten ist die Unterlassung der Darstellung fremder Webseiten in einem Frame, der kein eigenständiges Fenster außerhalb des Kontextes der aufrufenden Webseite darstellt, wenn der Fremdanbieter nicht explizit mit der Frame-Darstellung einverstanden ist. Letzteres ist nicht zu erwarten, da in der Praxis die o.g. pauschale Erklärung oft ins Leere führen wird (weil es schwer nachprüfbar ist, welche Frame-Darstellungen und deren jeweiliger Kontext existieren) und im Moment der Risiko- und Schadenssituation für den User irrelevant, weil nicht helfend ist.

4.3.2.2.4.2.2.3.2. Lage der Cookies am Beispiel von Microsoft Windows 9.x

liegen unter \Windows in einem speziellen Ordner
 meist C:\Windows\Cookies (Ordner, der systemgeschützt ist)
 und/oder im Browser-Cache (Ordner, der systemgeschützt ist)

Hinweis: Der Windows-Pfad mit Laufwerksangabe ist als System-Umgebungs-Variable hinterlegt, auch wenn ein anderes Verzeichnis als \Windows benutzt wird.

Ein Cookie kann bis zu 4 KBytes groß sein.
 Es sind bis zu 300 verschiedene Cookies gleichzeitig speicherbar.

4.3.2.2.4.2.2.3.3. Vermeidung von Cookies

Cookieabschaltung für Onlinesitzung(en)

Wenn die Cookiesverwaltung nicht erwünscht ist, so muss sie im Browser abgeschaltet werden, falls es der Browser zulässt sollte eine Firewall-Software benutzt werden, wenn diese im Betriebssystem nicht bereits integriert ist bzw. die betriebssystemeigene Firewall bzw. die Browsereinstellungen bezüglich der Cookieverwaltung unzureichend sind.

Achtung: Eine permanente Cookie-Abschaltung bewirkt bei manchen Webseiten, dass diese nicht lauffähig sind, obwohl Cookies kein Bestandteil von HTML sind.

Cookies löschen nach einer Onlinesitzung

nach der Onlinesitzung Löschung des INHALTES
 des Cookie-Ordners: keine DAT-Dateien
 keine system-geschützte Dateien
 keinesfalls des Ordners selbst
 und des Browser-Cache: keine system-geschützte Dateien
 keinesfalls des Ordners selbst

Sichere Löschung kann nur z.T. über Features des Browsers erfolgen (je nach Browserhersteller und Browser-Version).

Falls der Browser keine Features besitzt, so sollte man anstelle der manuellen Löschung vorhandene Features des Betriebssystems bzw. Fremdsoftware verwenden.

4.3.2.2.4.2.2.3.4. Cookie verwalten (Beispiele)

Cookie schreiben

Im Beispiel wird davon ausgegangen, dass NUR der Wert des Cookies und keine Cookiekomponenten geschrieben werden sollen.

```
function schreibe_cookie(freier_bezeichner,wert_des_cookie,anzahl_der_tage)
{
  // Verfalldatum als Datum-Variable erzeugen
  verfall_datum= new Date();

  //Verfalldatum-Variable mit Wert belegen
  verfall_datum.setTime(      verfall_datum.getTime()
                           + (86400000*anzahl_der_tage)
                           );
  // setTime verlangt Millisekunden
  // pro Tag 86400000 Millisekunden = 24 * 60 * 60 * 1000
  // getTime  Zeit des Schreibens
```



```

// wenn anzahl_der_tage=0 ist sofortiges Löschen des Cookie
// nach der Online-Sitzung

// Cookie-Format erzeugen (nur Wert und Verfalldatum)
// keine Blanks in das Format einbauen !!
document.cookie= freier_bezeichner
                 + "="
                 + escape(wert_des_cookie)
                   + ";expires="
                 + verfall_datum.toGMTString();
}

```

Cookie lesen

Im Beispiel wird davon ausgegangen: Wenn mindestens eine Komponente vorhanden ist, so wird nur die ERSTE gelesen.

```

function lese_cookie(freier_bezeichner)
{
    // Funktionswert initialisieren
    // Annahme: Cookie NICHT gefunden
    funktionswert=null;

    // Suchbegriff anhand Cookie-Name bilden
    // darf keine Blanks enthalten !!
    suchebegriff=freier_bezeichner + "=";
    // und die Länge des Suchbegriffes holen
    suchbegriff_laenge=suchbegriff.length;

    // Anzahl der Bytes aller Cookies im Puffer des Dokumentes holen
    puffer_laenge=document.cookie.length;

    // und suchen: Pufferinhalt wird häppchenweise durchsucht per Teilkette
    teilkette_startposition=0; // Startposition der Teilkette im Puffer

    while (teilkette_startposition < puffer_laenge) // solange nicht Pufferende erkannt
    {
        // Pufferinhalt als Häppchen, also Teilkette adressieren
        teilkette_endeposition=teilkette_startposition + suchbegriff_laenge;

        // Pufferinhalt als Häppchen holen
        teilkette=document.cookie.substring(teilkette_startposition,teilkette_endeposition);

        // Vergleich der Teilkette mit dem Suchbegriff
        if (teilkette == suchebegriff)
        {
            // Cookie wurde gefunden
            // Rest der Teilkette (Häppchen) nach einem Semikolon durchsuchen
            // also prüfen, ob mindestens eine Komponente vorhanden ist
            semi_position= document.cookie.indexOf(";",teilkette_endeposition);

            if (semi_position == -1)
            {
                // kein Semikolon gefunden wenn NICHT >=0
                funktionswert=
                unescape(document.cookie.substring(
                    teilkette_endeposition,document.cookie.length
                ));
            }
            else
            {
                // Semikolon gefunden
                // NUR die ERSTE Komponente lesen
                funktionswert=
                unescape(document.cookie.substring(
                    teilkette_endeposition,semi_position
                ));
            }
        }

        return(funktionswert); // gesamte Funktion beenden
    }
    else
    {

```



```

// Cookie wurde NICHT gefunden
// Trennblank zum nächsten Cookie suchen
blank_position=
    document.cookie.indexOf(" ",teilkette_startposition);
if (blank_position == -1)
{break;} // kein weiteres Cookie vorhanden, also while abbrechen
else
{teilkette_startposition= blank_position+1;}
// neue Startposition der Teilkette holen
    }
}
return(funktionswert);
}

```

4.3.2.4.2.2.3.5. document.cookie Collection

Feld der Zeiger auf alle Cookies zum Dokument
Ein Cookie kann bis zu 4 KBytes groß sein.
Es sind bis zu 300 verschiedene Cookies gleichzeitig speicherbar.

Cookies haben den gleichen Aufbau aber verschiedene Inhalte:

Optionale Eigenschaften müssen ein Semikolon vorgesetzt bekommen und können beliebig kombiniert werden.

Erzeugung:

durch den Browser

Zugriff:

```

[ var ZeigerAufFeld = ] document.cookie
[ var ZeigerAufFeldElement = ] document.cookie[Index]

```

Index	Integer ab 0 muss in [] kodiert sein
-------	------------------------------------------

Mindestdeklaration eines Cookie (Wert eines Cookies):

```
document.cookie[0] = wert_des_cookie_1;
```

```

oder
var cookie_feld = document.cookie;
cookie_feld[0] = wert_des_cookie_1;

```

wert_des_cookies: Zeichenkette
kann HTML-gerecht kodierte Sonderzeichen enthalten
darf kein Blank enthalten, da jedes Cookie intern als Endekennzeichen ein Blank besitzt

Aufbau: freierwert[;eigenschaften_liste]

Semikolon trennt die optionale Eigenschaftenliste ab
Eigenschaften in der Liste durch Semikolon trennen

freierwert: cookie_titel=freier_wert

= ist festkodiert

Bsp: "Filmbeginn=23:15;path=...;expires=....."

Anwendungsbereich des Cookie also nicht die Lage auf der Festplatte der Users:

path=pfad_angabe

path= ist festkodiert

pfad_angabe:

Bsp.: 'Otto=der_erste;c:\privat\html_files'
also Anwendbarkeit nur möglich, wenn HTML-Dokument
innerhalb bzw. unterhalb von c:\privat\html_files liegt
wenn nicht kodiert, wo wird der Pfad des Browser-Cache verwendet

Servergruppezugehörigkeit:

domain=server_name_fragment

domain= ist festkodiert

server_name_fragment:

Bsp.: "Otto=der_erste;domain=online.de"
also alle Server, die im Namen den Rest
online.de haben
wenn nicht kodiert, so wird der volle Name
von dem Server abgelegt, der das Cookie



verwaltet und nutzt

Verfallsdatum:`expires=datum_angabe``expires=` ist festkodiert`datum_angabe:` wochentag, tt-mm-jj hh-mm-ss
nach Greenwich-TimeBsp.: `var loeschdatum = new Date();``"Otto=der_erste;epires=" +``loeschdatum.toGMTString();`

wenn nicht kodiert, so Cookie mit Ende der

Onlinesitzung von der Festplatte des

Users automatisch gelöscht

wenn kodiert, so am Verfallsdatum bzw. danach

von der Festplatte des Users automatisch

gelöscht (danach, wenn der User des PC

nach dem Verfallsdatum erst wieder

nutzt)

Cookie nur senden, wenn HTTPS-Protokoll also sichere Verbindung benutzt wird:`secure`

kein Wert

Beispiel für Cookie verwalten:

```

<SCRIPT>
function ErzeugeCookie(Name, Value) // Name und Wert sind Strings
{
    var date = new Date();
    var document.cookie = Name
                        + "="
                        + escape(Value)
                        + "; expires=" + date.toGMTString(); // aktuelles Datum
}

function LoescheCookie (Name, Value) // Name und Wert sind Strings
{
    var document.cookie = Name
                        + "="
                        + escape(Value)
                        + "; expires=Fri, 31 Dec 1999 23:59:59GMT;"; // altes Datum
}

function LeseCookieWert(Name)
// Name des zu lesenden Cookie ist String
// liefert Cookiewert aus unescape() als String
{
    var CookieWert=""; // Annahme: Cookie nicht gefunden oder mit Leerkette als Wert

    // Feld der Cookies referenzieren
    // erzeugt durch Separation anhand Semikolon
    var CookieFeld = document.cookie.split("; ");
    var AnzahlCookies = CookieFeld.length;

    // Feldelement umfasst Name und Wert des Cookie
    // Aufbau Cookie_Name = Cookie_Wert
    // Separator ist Gleichheitszeichen
    // Index 0 für Cookie_Name
    // Index 1 für Cookie_Wert
    var CookieNameUndWertAlsFeld;

    // CookieFeld auslesen und auf Cookie laut Name prüfen
    var Gefunden=false;
    var Index = 0;
    do
    {
        // aktuelles Cookie lesen
        CookieNameUndWertAlsFeld = CookieFeld [Index].split("=");

        // und auf Name prüfen
        if (Name == CookieNameUndWertAlsFeld [0])
        {

```



```

        CookieWert = unescape(CookieNameUndWertAlsFeld [1]);
        Gefunden=true;
    }
    else
    {Index++;}
}
while ( ( !Gefunden )
    && ( Index < AnzahlCookies)
);

return CookieWert;
}
</SCRIPT>

```

Eigenschaften:

.length Anzahl der Cookies, ab 1

Methoden:

keine

Auf den Wert des Cookies sind alle Methoden des String-Objektes anwendbar, da der Wert immer eine Zeichenkette ist.

4.3.2.2.4.2.2.4. Zusätzliche Verhaltensweisen eines HTML-Elementes bzw. des HTML-Dokumentes**im Internet Explorer ab 5.x**

Nachfolgend werden Möglichkeiten der Implementation von zusätzlichen Verhaltensweisen vom HTML-Dokument und/oder HTML-Elemente im HTML-Bokument beschrieben, die nur der Internet Explorer ab 5.x aufweist.

Behaviors sind Verhaltensweisen von Elementen.

Verhaltensweisen werden mit externen *.htc-Dateien beschrieben, die importiert werden müssen.

Nur die Standard-Behavior (z.B. .style.time2) des IE benötigen keine *.htc-Datei

4.3.2.2.4.2.2.4.1. Standard-Behavior

im Browser implementiert Standard-Verhaltensweisen z.B. anhand des Objektes style.time2 (siehe dort)

4.3.2.2.4.2.2.4.2. element Objekt des Internet Explorer (HTC-Datei)

Objekt einer HTC-Komponente aus einer HTC-Datei

ab IE 5.x

Behavior einbinden:

Beispiel 1:

test.htc enthält:

```

<PUBLIC:COMPONENT NAME="HTC_Komponente">
  <PUBLIC:EVENT NAME="onResultChange" ID="ID_Event"/>

  <SCRIPT LANGUAGE="JScript">
    function Berechne () // Bezug über ID_Event
    {
      var EventObjekt = createEventObject();
      EventObjekt.result = "Wert";

      ID_Event.fire (EventObjekt);
    }
  </SCRIPT>
</PUBLIC:COMPONENT>

```

HTML-Dokument enthält:

```

<HTML XMLNS:privater_tag_namensraum>
<HEAD>
<STYLE>
  @media all { privater_tag_namensraum:privater_tag {behavior:url(test.htc); } }
</STYLE>
</HEAD>
<BODY>
<privater_tag_namensraum:privater_tag ID="ID_privater_tag"
  onResultChange="ID_Div.innerText=window.event.result"
>
  <TABLE>
    <TR>
      <DIV ID="ID_Div"
        ALIGN=RIGHT
        STYLE="border: '.025cm solid gray'"
      >
        0.
      </DIV>
    </TR>
  </TABLE>

```



```

<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 7 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 8 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 9 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" / ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" C ">
  </TD>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 4 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 5 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 6 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" * ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" %" DISABLED>
  </TD>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 1 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 2 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 3 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" - ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 1/x " DISABLED>
  </TD>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 0 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" +/- ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" . ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" + ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" = ">
  </TD>
</TR>
</TABLE>

```

</privater_tag_namensraum:privater_tag>
 <BODY>



</HTML>

Beispiel 2:

test.htc enthält:

```

<PUBLIC:COMPONENT NAME="HTC_Komponente">
  <PUBLIC:METHOD NAME="private function" />
    <SCRIPT LANGUAGE="JScript" >
      function private function ()
      {
        // .....
      }
    </SCRIPT>
  </PUBLIC:METHOD>
</PUBLIC:COMPONENT>

```

HTML-Dokument enthält:

```

<HTML>
<HEAD>
  <STYLE>
    .klasse {behavior:url(test.htc)}
  </STYLE>
</HEAD>
<BODY ID="ID_Body"
  CLASS="klasse"
>
  <DIV onclick=" ID_Body.private function ()">
    .....
  </DIV>
</BODY>
</HTML>

```

HTC-Datei:

dient der Implementierung von Verhaltensweisen (Behavior) in das HTML-Dokument
Suffix *.htc
beinhaltet Script und HTC-Komponenten

HTC-Komponente:

dient der Implementierung von Verhaltensweisen (Behavior) in das HTML-Dokument
ist Bestandteil der HTC-Datei

HTC-Komponenten als element-Objekt erzeugen per PUBLIC:COMPONENT (siehe dort)
HTC-Komponente als Objekt in der Collection document.all im HTML-Dokument erzeugen per PUBLIC:PROPERTY (siehe dort)

HTC-Komponente	kann	besitzen:	Eigenschaften	(PUBLIC:PROPERTY)
			Methoden	(PUBLIC:METHOD)
			Event	(PUBLIC:EVENT)
			Eventhandler	(PUBLIC:ATTACH)
			JScript-Code	
	verarbeiten		Event	(PUBLIC:ATTACH und PUBLIC:EVENT)

ist der Container (PUBLIC:COMPONENT) der o.g. Bestandteile für die Kodierung in der HTC-Datei

PUBLIC:ATTACH für HTC-Komponente laut PUBLIC:COMPONENT einem Event den Eventhandler zuordnen und diese Zuordnung dem HTML-Dokument bekanntgeben
Mehrfachkodierung für ein und dasselbe Event vermeiden, da sonst Zufallsauswahl erfolgt
Syntax:

```

<PUBLIC:ATTACH EVENT = Kette1
  [FOR = Kette2]
  [ID = Kette3]
  ONEVENT = Kette4
>
</PUBLIC:ATTACH>

```

muss im Container PUBLIC:COMPONENT kodiert sein

</> als Ende-Tag möglich

Kette1 Bezeichner eines Events in der Form onXXX
laut HTC-Referenz
Achtung: Gross-Klein wird unterschieden



Kette2 optional
Referenz auf die Eventquelle
Standard: Objekt, das das Behavior besitzt
auch kodierbar "document" oder "window" kodierbar

Kette3 optional
kein Standard
Funktionalität wie ID-Attribut im HTML-Tag

Kette4 Bezeichner der Eventhandler-Script-Funktion
in der Funktion die Referenzen auf Methoden der Objekte wie window
oder document immer im Pfad komplett kodieren also
mit window. bzw. document.

Beispiel:

```
<PUBLIC:ATTACH EVENT="ondetach" ONEVENT="EventEntfernen()"/>

<SCRIPT LANGUAGE="JScript">
function CursorNeu()
{
    if (event.srcElement == element)
    {
        normalColor = style.color;
        runtimeStyle.color = "red";
        runtimeStyle.cursor = "hand";
    }
}

function CursorNormal()
{
    if (event.srcElement == element)
    {
        runtimeStyle.color = normalColor;
        runtimeStyle.cursor = "";
    }
}

function EventEntfernen()
{
    detachEvent ('onmouseover', CursorNeu); // nicht () kodieren !!!
    detachEvent ('onmouseout', CursorNormal); // nicht () kodieren !!
}

attachEvent ('onmouseover', CursorNeu);
attachEvent ('onmouseout', CursorNormal);
</SCRIPT>
```

PUBLIC:COMPONENT HTC-Komponente definieren (Container) und dem HTML-Dokument bekanntgeben
Kinder: einmalig PUBLIC:DEFAULTS
ein-oder mehrmalig PUBLIC:ATTACH,
PUBLIC:EVENT,
PUBLIC:METHOD,
PUBLIC:PROPERTY

Syntax:

```
<PUBLIC:COMPONENT
[ID = Kette1]
[lightWeight = Kette2]
[literalContent = Kette3]
[NAME = Kette4]
[supportsEditMode = Kette5]
[tagName = Kette6]
[URN = Kette7]
>
<!-- hier alle anderen HTC-Komponenten kodieren -->
</PUBLIC:COMPONENT>
```

Kette1 optional
kein Standard
Funktionalität wie ID-Attribut im HTML-Tag

Kette2 optional
"true" kein Parsen und eventuelle Anzeige aller Tags



(Inhalte zwischen den Start- und Endetags)
 "false" Standard
 Parsen und eventuelle Anzeige aller Tags
 (Inhalte zwischen den Start- und Endetags)
 Parsen und Anzeige ein/aus für einen speziellen Tag: siehe Kette3

Kette3 optional
 Auswertung des Inhaltes zwischenTags laut tagName=Kette6
 "false" Standard
 Inhalt zwischen Start- und Ende-Tag laut Kette6 wird geparkt und eventuell angezeigt
 "nested" ab IE 6.x
 Inhalt zwischen dem **ersten** Start- und dem **letzten** Ende-Tag laut Kette6 wird nicht geparkt und nicht angezeigt, sondern als Dateninsel (Datenquelle) verarbeitet
 "true" Inhalt zwischen dem **ersten** Start- und dem **ersten** Ende-Tag laut Kette6 wird nicht geparkt und nicht angezeigt, sondern als Dateninsel (Datenquelle) verarbeitet

Kette4 optional
 analog zu ID

Kette5 optional
 generelle Veränderbarkeit des Kontextes aller HTC-Komponenten
 "true" veränderbar
 "false" Standard
 nicht veränderbar

Kette6 optional
 Bezeichner eines Tags einer oder mehrerer HTC-Komponenten
 Verwendung des Tags : siehe literalContent = Kette3
 userdefinierter Tag ab IE 5.5

Kette7 optional
 Uniform Resource Name (URN) für Eventsteuerung:
 Quelle des Events (srcUrn) laut URN des Behavior,
 der das Event erzeugt

PUBLIC:DEFAULTS

für eine HTC-Komponenten die Standardangaben setzen und dem HTML-Dokument bekanntgeben darf nur 1x kodiert werden im Container PUBLIC:COMPONENT
 Syntax:

```
<PUBLIC:DEFAULTS
  [canHaveHTML      = Kette1]
  [contentEditable  = Kette2]
  [style             = Kette3]
  [tabStop           = Kette4]
  [viewInheritStyle = Kette5]
  [viewLinkContent  = Kette6]
  [viewMasterTab    = Kette7]
>
</PUBLIC:DEFAULTS>
```

muss im Container PUBLIC:COMPONENT kodiert sein

/> als Ende-Tag möglich

Kette1 optional
 "false" Komponente darf zwischen Start- und Endetag kein HTML besitzen
 "true" Komponente darf zwischen Start- und Endetag HTML besitzen

Kette2 optional
 "inherit" Veränderlichkeit des Kontext von Eltern geerbt
 "false" Veränderlichkeit nicht möglich
 "true" Veränderlichkeit möglich

Kette3 optional
 Werte wie laut HTML-STYLE-Attribut

Kette4 optional



	"false"	Standard
		Komponente per TAB-Taste nicht aktivierbar
	"true"	Komponente per TAB-Taste aktivierbar
Kette5	optional	
	"false"	Styles vom HTML-Dokument nicht übernehmen
	"true"	Styles vom HTML-Dokument übernehmen
Kette6	optional	
	"false"	Standard
		Viewlink nicht benutzt
	"true"	Viewlink nutzbar
Kette7	optional	
	"false"	Master-Element des Viewlink in der Tab-Tastenfolge des HTML-Dokumentes nicht enthalten
	"true"	Master-Element des Viewlink in der Tab-Tastenfolge des HTML-Dokumentes enthalten

PUBLIC:EVENT für eine HTC-Komponente ein Event definieren und dem HTML-Dokument bekanntgeben
 Syntax:

```
<PUBLIC:EVENT
  [ID      = Kette1]
  NAME    = Kette2
>
</PUBLIC:EVENT>
```

muss im Container PUBLIC:COMPONENT kodiert sein

```
</> als Ende-Tag möglich
```

Kette1 optional
kein Standard
Funktionalität wie ID-Attribut im HTML-Tag

Kette2 analog zu ID

Methode: Event erzeugen und an das HTML-Dokument weiterreichen

```
Syntax:
  Kette1.fire( [zeiger_auf_event_objekt] )
```

zeiger_auf_event_objekt per createEventObject() erzeugen

Beispiel:
test.htc enthält:

```
<PUBLIC:COMPONENT NAME="HTC_Komponente">
  <PUBLIC:EVENT NAME="onResultChange" ID="ID_Event"/>

  <SCRIPT LANGUAGE="JScript">
    function Berechne()// Bezug über ID_Event
    {
      var EventObjekt = createEventObject();
      EventObjekt.result = "Wert";

      ID_Event.fire (EventObjekt);
    }
  </SCRIPT>
</PUBLIC:COMPONENT>
```

HTML-Dokument enthält:

```
<HTML XMLNS:privater_tag_namensraum>
<HEAD>
<STYLE>
  @media all { privater_tag_namensraum:privater_tag {behavior:url(test.htc)} }
</STYLE>
</HEAD>
<BODY>
  <privater_tag_namensraum:privater_tag ID="ID_privater_tag"
    onResultChange="ID_Div.innerText=window.event.result"
  >
  <TABLE>
```



```

<TR>
  <DIV ID="ID_Div"
        ALIGN=RIGHT
        STYLE="border: '.025cm solid gray'"
  >
    0.
  </DIV>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 7 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 8 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 9 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" / ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" C ">
  </TD>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 4 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 5 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 6 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" * ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" %" DISABLED>
  </TD>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 1 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 2 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 3 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" - ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 1/x " DISABLED>
  </TD>
</TR>
<TR>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" 0 ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" +/- ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" . ">
  </TD>
  <TD>
    <INPUT TYPE=BUTTON VALUE=" + ">
  </TD>
</TR>

```



```

</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" ">
</TD>
</TR>
</TABLE>
</privater_tag_namensraum:privater_tag>
<BODY>
</HTML>

```

PUBLIC:METHOD für ein HTC-Komponente eine Methode definieren und dem HTML-Dokument bekanntgeben
 Syntax:

```

<PUBLIC:METHOD
  [ID = Kette1]
  [INTERNALNAME = Kette2]
  NAME = "sName"
>
  <!-- Script-Code der Methode mit Bezeichner laut Kette3 //-->
</PUBLIC:METHOD>

```

muss im Container PUBLIC:COMPONENT kodiert sein

- Kette1 optional
kein Standard
Funktionalität wie ID-Attribut im HTML-Tag
- Kette2 optional
komponenten-lokaler Bezeichner der Methode
Standard ist Kette3
- Kette3 öffentlicher Name der Methode im HTML-Dokument
analog zu ID

Beispiel:
test.htc enthält:

```

<PUBLIC:COMPONENT NAME="HTC_Komponente">
  <PUBLIC:METHOD NAME="private function" />
  <SCRIPT LANGUAGE="JScript" >
    function private function ()
    {
      // .....
    }
  </SCRIPT>
</PUBLIC:METHOD>
</PUBLIC:COMPONENT>

```

HTML-Dokument enthält:

```

<HTML>
<HEAD>
  <STYLE>
    .klasse {behavior:url(test.htc)}
  </STYLE>
</HEAD>
<BODY ID="ID_Body" CLASS="klasse">
  >
  <DIV onclick="ID_Body.private function ()">
    .....
  </DIV>
</BODY>
</HTML>

```

PUBLIC:PROPERTY für eine HTC-Komponente eine Eigenschaften definieren
 und dem HTML-Dokument bekanntgeben
 und automatisch als Element der Collection document.all erzeugen
 Zugriff auf HTC-Komponente im HTML-Dokument:

```

element.document.all(name_der_komponente)
name_der_komponente laut Attribut NAME

```

```

Syntax:
<PUBLIC:PROPERTY
  [GET = Kette1]

```



```

[ID           = Kette2]
[INTERNALNAME = Kette3]
[NAME        = Kette4]
[PERSIST     = Kette5]
[PUT         = Kette6]
[VALUE      = Kette7]
>
</PUBLIC:PROPERTY>

```

muss im Container PUBLIC:COMPONENT kodiert sein

/> als Ende-Tag möglich

Kette1	optional	Bezeichner der Funktion, die im HTML-Dokument aufzurufen ist, wenn der Wert der HTC-Komponenten-Eigenschaft ermittelt werden soll (Lesen der HTC-Komponenten-Eigenschaft)
Kette2	optional	kein Standard Funktionalität wie ID-Attribut im HTML-Tag
Kette3	optional	komponenten-lokaler Bezeichner der Methode Standard ist Kette4
Kette4		öffentlicher Name der Komponente im HTML-Dokument analog zu ID
Kette5	optional	"true" Eigenschaft dem HTML-Dokument bekanntgeben "false" Eigenschaft dem HTML-Dokument nicht bekanntgeben
Kette6	optional	Bezeichner der Funktion, die im HTML-Dokument aufzurufen ist, wenn der Wert der HTC-Komponenten-Eigenschaft gesetzt werden soll (Schreiben der HTC-Komponenten-Eigenschaft)
Kette7	optional	Initial-Wert bzw. aktueller Wert der Eigenschaft

Methode: Kette2.fireChange() erzeugt Event onpropertychange und gibt es an das HTML-Dokument, sobald der Wert der Eigenschaft verändert wird, also PUT aktiviert wird
muss in der Routine zu PUT kodiert werden

Beispiel 1:

```

<PUBLIC:PROPERTY NAME="HTC_Komponente"/>
<PUBLIC:ATTACH EVENT="onclick" ONEVENT="EventHandler()" />

<SCRIPT LANGUAGE="JScript">
function EventHandler()
{
    var HTC_KomponenteObjekt = element.document.all(HTC_Komponente);

    var HTC_KomponenteObjekt_Sichtbarkeit_RunTimeStyle =
        HTC_KomponenteObjekt.runtimeStyle.display;

    if (HTC_KomponenteObjekt_Sichtbarkeit_RunTimeStyle == "none")
    {
        runtimeStyle.listStyleImage = "url('test1.gif)";
        HTC_KomponenteObjekt_Sichtbarkeit_RunTimeStyle = "";
    }
    else
    {
        runtimeStyle.listStyleImage = "url('test2.gif)";
        HTC_KomponenteObjekt_Sichtbarkeit_RunTimeStyle = "none";
    }
}
</SCRIPT>

```




```

</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" 7 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" 8 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" 9 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" / ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" C ">
</TD>
</TR>
<TR>
<TD>
<INPUT TYPE=BUTTON VALUE=" 4 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" 5 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" 6 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" * ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" % " DISABLED>
</TD>
</TR>
<TR>
<TD>
<INPUT TYPE=BUTTON VALUE=" 1 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" 2 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" 3 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" - ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" 1/x " DISABLED>
</TD>
</TR>
<TR>
<TD>
<INPUT TYPE=BUTTON VALUE=" 0 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" +/- ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" . ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" + ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" = ">
</TD>
</TR>
</TABLE>
</privater_tag_namensraum:privater_tag>
<BODY>
</HTML>

```



.fire() ein in der HTC-Komponente definiertes Event erzeugen und an das HTML-Dokument weiterreichen
siehe Objekt element und HTC-Datei
Syntax:

```
ID_Event.fire( [zeiger_auf_event_objekt] )
```

ID_Event laut Wert des Attributes ID in <PUBLIC:EVENT ...>

zeiger_auf_event_objekt per createEventObject() erzeugen

Beispiel:
test.htc enthält:

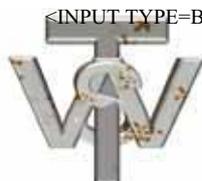
```
<PUBLIC:COMPONENT NAME="HTC_Komponente">
  <PUBLIC:EVENT NAME="onResultChange" ID="ID_Event"/>

  <SCRIPT LANGUAGE="JScript">
    function Berechne()// Bezug über ID_Event
    {
      var EventObjekt = createEventObject();
      EventObjekt.result = "Wert";

      ID_Event.fire (EventObjekt);
    }
  </SCRIPT>
</PUBLIC:COMPONENT>
```

HTML-Dokument enthält:

```
<HTML XMLNS:privater_tag_namensraum>
<HEAD>
<STYLE>
  @media all { privater_tag_namensraum\:privater_tag {behavior:url(test.htc)} }
</STYLE>
</HEAD>
<BODY>
  <privater_tag_namensraum:privater_tag ID="ID_privater_tag"
    onResultChange="ID_Div.innerText=window.event.result"
  >
    <TABLE>
      <TR>
        <TD>
          <DIV ID="ID_Div"
            ALIGN=RIGHT
            STYLE="border: '.025cm solid gray'"
          >
            0.
          </DIV>
        </TD>
        <TD>
          <INPUT TYPE=BUTTON VALUE=" 7 ">
        </TD>
        <TD>
          <INPUT TYPE=BUTTON VALUE=" 8 ">
        </TD>
        <TD>
          <INPUT TYPE=BUTTON VALUE=" 9 ">
        </TD>
        <TD>
          <INPUT TYPE=BUTTON VALUE=" / ">
        </TD>
        <TD>
          <INPUT TYPE=BUTTON VALUE=" C ">
        </TD>
      </TR>
      <TR>
        <TD>
          <INPUT TYPE=BUTTON VALUE=" 4 ">
        </TD>
        <TD>
          <INPUT TYPE=BUTTON VALUE=" 5 ">
        </TD>
        <TD>
          <INPUT TYPE=BUTTON VALUE=" 6 ">
        </TD>
```



```

</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" * ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" % " DISABLED>
</TD>
</TR>
<TR>
<TD>
<INPUT TYPE=BUTTON VALUE=" 1 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" 2 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" 3 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" - ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" 1/x " DISABLED>
</TD>
</TR>
<TR>
<TD>
<INPUT TYPE=BUTTON VALUE=" 0 ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" +/- ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" . ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" + ">
</TD>
<TD>
<INPUT TYPE=BUTTON VALUE=" = ">
</TD>
</TR>
</TABLE>
</privater_tag_namensraum:privater_tag>
<BODY>
</HTML>

```

.fireChange() erzeugt Event onpropertychange und gibt es an das HTML-Dokument, sobald der Wert der Eigenschaft per Aufruf der Funktion laut Attribut PUT verändert wird
 Aufruf von .fireChange() muss programmiert werden
 siehe Objekt element und HTC-Datei
 Syntax:

```
ID_HTC_Komponente.fireChange();
```

ID_HTC_Komponente laut Attribut ID in <PUBLIC:PROPERTY ...>

Beispiel:

```

<PUBLIC:PROPERTY
  ID="ID_HTC_Komponente"
  NAME="HTC_Komponente"
  PUT="Eigenschaft_Schreiben"
  GET="Eigenschaft_Lesen"
/>

<SCRIPT LANGUAGE="JScript">
  var Eigenschaft_Wert =null;

  function Eigenschaft_Schreiben (Wert)
  {
    // Wert zuweisen
    Eigenschaft_Wert = Wert;
  }

```



```

        // und Event erzeugen
        ID_HTC_Komponente.fireChange();
    }

    function Eigenschaft_Lesen ()
    {return Eigenschaft_Wert; }
</SCRIPT>

```

Events:

für <PUBLIC:ATTACH EVENT => (außer onpropertychange)

oncontentready erzeugt, wenn auf die HTC-Komponente zugegriffen wird und dieses komplett geparkt ist und das Event window.onload erzeugt wurde
Hinweis: Das Parsen einer HTC-Komponente kann durch die Komponente selbst verhindert werden.
siehe Objekt element und HTC-Datei

Beispiel:

```

<PUBLIC:ATTACH EVENT="oncontentready" ONEVENT="Anzeige()" />
<SCRIPT LANGUAGE="JScript">
    function Anzeige()
    {window.alert ('Objekt element mit innerHTML = ' + element.innerHTML); }
</SCRIPT>

```

oncontentsave erzeugt, wenn die Umgebung der HTC-Komponente gespeichert oder kopiert wird
Hinweis: Änderung einer Eigenschaft der HTC-Komponente wird per Event onpropertychange angezeigt
siehe Objekt element und HTC-Datei

ondetach erzeugt, wenn die Event-Überwachung in einer HTC-Komponente abgeschaltet wird
siehe Objekt element und HTC-Datei

Beispiel:

```

<PUBLIC:ATTACH EVENT="ondetach" ONEVENT="EventEntfernen()"/>
<SCRIPT LANGUAGE="JScript">
    function CursorNeu()
    {
        if (event.srcElement == element)
        {
            normalColor    = style.color;
            runtimeStyle.color = "red";
            runtimeStyle.cursor = "hand";
        }
    }

    function CursorNormal()
    {
        if (event.srcElement == element)
        {
            runtimeStyle.color = normalColor;
            runtimeStyle.cursor = "";
        }
    }

    function EventEntfernen()
    {
        detachEvent ('onmouseover', CursorNeu); // nicht () kodieren !!!
        detachEvent ('onmouseout', CursorNormal); // nicht () kodieren !!
    }

    attachEvent ('onmouseover', CursorNeu);
    attachEvent ('onmouseout', CursorNormal);
</SCRIPT>

```

ondocumentready erzeugt, wenn das HTML-Dokument, das die HTC-Komponente implementiert hat, komplett geparkt ist und das Event window.onload erzeugt wurde
Parsen des HTML-Dokumentes schliesst Implementierung der HTC-Datei und Parsen der HTC-Komponenten ein
Hinweis: Das Parsen einer HTC-Komponente kann durch die Komponente selbst verhindert werden.
siehe Objekt element und HTC-Datei

onpropertychange erzeugt, wenn die Methode **ID_HTC_Komponente.fireChange()** in der Funktion laut Attribut PUT aufgerufen wird (Aufruf muss programmiert werden), wobei die Funktion eine Eigenschaft der HTC-Komponente mit neuem Wert belegt



siehe Objekt element und HTC-Datei

4.3.2.2.4.2.2.4.3. *behaviorUrns* Collection des Internet Explorer

Feld aller Uniform Resource Name (URN) als String

Behaviors sind Verhaltensweisen von Elementen.

siehe auch document.namespace Objekt

Syntax:

```
[ var ZeigerAufFeld = ] object.behaviorUrns
[ var ZeigerAufElementAusFeld = ] object.behaviorUrns[Index]
```

Index Integer, ab 0
muss in [] kodiert sein

Beispiel

```
<HEAD>
<STYLE>
  DIV { behavior:url(fly.htc) url (zoom.htc) url (fade.htc);}
</STYLE>
<SCRIPT>
  function Anzeige()
  {
    var ZeigerAufFeld = ID_Div.behaviorUrns;
    if (ZeigerAufFeld != null)
    {
      for (var i=0; i < ZeigerAufFeld.length; i++)
      {alert (ZeigerAufFeld.item[i]);}
    }
  }
</SCRIPT>
</HEAD>
<BODY onload="Anzeige()">
  <DIV ID="ID_Div">Test </DIV>
</BODY>
```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des
Attributnamen (analog zu ID oder NAME-Attribut) liefern
außer bei Formular mit <INPUT TYPE=image ...>
da dafür die children-Collection verwendet werden muss !!!

4.3.2.2.4.2.2.4.4. *document.namespace* Objekt des Internet Explorer

Dieses Objekt dient der dynamischen Verwaltung von Behavior (Verhalten) eines Elementes ab IE 5.5 (Behavior sind ab IE 5.x möglich).

Der Programmierer kann private Tags erzeugen und deren Verhalten (Behavior) selbst bestimmen. Problem dabei ist, dass der Browser private Tags nicht kennt. Daher muss dazu folgendes realisiert werden:

Ein Tag hat im Browser einen Namensraum. Tags die der Browser laut HTML kennt, liegen also im HTML-Namensraum. Private Tags, die natürlich nicht aus dem HTML-Namensraum stammen, müssen also einen eigenen Namensraum erhalten, damit der Browser diese Tags parsen kann.

Ein Tag hat im Browser eine feste Definition. Tags, die aus dem HTML-Namensraum kommen (z.B. HEAD) müssen dem Browser nicht weiter mitgeteilt werden. Er kennt sie und kann sie parsen. Private Tags dagegen müssen als Element des privaten Namensraumes definiert werden.

Ein Tag hat im Browser eine genau definierte Aktion, die nach dem Parsen durch den Browser abgearbeitet wird, wobei der Browser dabei die Abarbeitungsfolge optimiert (siehe DOM). Ein privates Tag muss natürlich auch eine Aktion auslösen, die aber dem Browser erst mitgeteilt werden muss, damit er sie starten kann.

Ein Tag hat im Browser eine genau definierte Position im DOM. DOM ist browserabhängig und stellt die Struktur aller Objekte eines Dokumentes derart dar, dass der Browser objektorientiert das Dokument abarbeiten kann. Ein privates Tag muss natürlich auch Teil vom DOM sein, also integriert werden. Der Ort der Integration ist die Einbettung des Tags in das body Objekt.

Um diese Aktionen in das Dokument einzubetten, werden benutzt

```
HEAD,
BODY,
Script,
XML
```

für die Auslagerung der Aktionen des Tags als Programmcode das *.htc-Dateiformat.

Das Feld aller Behavior im Dokument ist die Collection document.namespaces (siehe dort).

Das einzigste Event ist onreadystatechange.



Der Internet Explorer kann von Hause aus Behavior nachladen, die von Microsoft geschrieben wurden. Wird ein Behavior verlangt, so erfolgt der Download der *.htc-Datei.

Nachfolgendes Beispiel zeigt den Import per Script:

```

<HEAD>
<HTML XMLNS: Mein_NamensRaum>
</HEAD>
<BODY onload=StartImport()>
<SCRIPT>
    var NamensRaumObjekt;

    function StartImport()
    {
        // HTML XMLNS füllt die Collection namespaces
        //     da nur ein Namensraum erzeugt wurde, ist also nur 1 Element
        //     in der Collection namespaces

        // Namensraum laut Collection referenzieren
        NamensRaumObjekt = document.namespaces[0];

        // Import des Programmcodes aus der Datei test.htc starten (Download)
        NamensRaumObjekt.doImport("test.htc");

        // prüfen ob Import (Download) beendet ist
        if (NamensRaumObjekt.readyState != "complete")
        {
            // Import läuft noch also Endeereignis abfangen per Eventhandler
            //     wenn Import beendet ist, soll die Einbindung des
            //     Tags in das body-Objekt erfolgen
            NamensRaumObjekt.attachEvent("onreadystatechange",
                TagInDasBodyObjektEinbinden
            ); // Funktion ohne () kodieren !!!
        }
        else
        {
            // Import erledigt, also den Tag in das BODY-Objekt einbinden
            addTagNamesToBody();
        }

        return true;
    }

    function TagInDasBodyObjektEinbinden()
    {
        // prüfen ob Import (Download) beendet ist
        if (NamensRaumObjekt.readyState != "complete")
        {return;}
        else
        {
            // Eventabfangen abschalten
            NamensRaumObjekt.detachEvent( "onreadystatechange",
                TagInDasBodyObjektEinbinden
            );

            // Tagobjekt erzeugen
            //     Tag ist ein Textelement
            var TagObjekt = document.createElement("Mein_NamensRaum:MeinTag");

            // Textelement füllen zum Rendern
            TagObjekt.innerText = "ElementBehavior";

            // und in das body-Objekt einbinden
            document.body.appendChild(TagObjekt);
        }
    }
</SCRIPT>
</BODY>
</HTML>

```

Inhalt der test.htc



```

<public:component tagName=MeinTag>
<public:attach event=onreadystatechange onevent=TagAktion()/>
</public:component>
<script>
    function TagAktion()
    {
        element.document.backgroundColor = "red";
    }
</script>

```

Nachfolgendes Beispiel für den Import per HTML:

```

<HTML XMLNS:MeinNamensRaum1
XMLNS:MeinNamensRaum2
XMLNS:MeinNamensRaum3
>
<HEAD>
<?IMPORT NAMESPACE="MeinNamensRaum2" IMPLEMENTATION="namespace1.htc">
<?IMPORT NAMESPACE="MeinNamensRaum2" IMPLEMENTATION="namespace2.htc">
<?IMPORT NAMESPACE="MeinNamensRaum3" IMPLEMENTATION="namespace3.htc">

<SCRIPT>
    function TabelleDynamischErweitern()
    {
        var TabellenZeile;
        var TabellenZelleDerZeile;

        for(var Index = 0; Index < document.namespaces.length; Index ++)
        {
            TabellenZeile = ID_Table.insertRow();

            // Zelle 1 mit Index
            TabellenZelleDerZeile = TabellenZeile.insertCell();
            TabellenZelleDerZeile.align = 'center';
            TabellenZelleDerZeile.innerText = Index; // oder .toString()

            // Zelle 2 mit Item-Methode
            document.namespaces.item(iIndex).name
            TabellenZelleDerZeile = TabellenZeile.insertCell();
            TabellenZelleDerZeile.align = 'center';
            TabellenZelleDerZeile.innerText =
                document.namespaces.item[Index].name;

            // Zelle 3 mit Index
            TabellenZelleDerZeile = TabellenZeile.insertCell();
            TabellenZelleDerZeile.align = 'center';
            TabellenZelleDerZeile.innerText = document.namespaces[Index].name;
        }
    }
</SCRIPT>
</HEAD>
<BODY>
<TABLE ID="ID_Table" BORDER=1>
<TR>
<TH>Zelle 1 Index</TH>
<TH>Zelle 2 Feldelement per Item-Methode</TH>
<TH>Zelle 3 Feldelement per Index</TH>
</TR>
</TABLE>
<BR>
<BUTTON onclick="TabelleDynamischErweitern();">Behavior anzeigen</BUTTON>
</BODY>
</HTML>

```

Zugriff:

nur über die Collection document.namespaces

```

var ZeigerAufFeldElement = document.namespaces [Index]
ZeigerAufFeldElement.eigenschaft
ZeigerAufFeldElement.methode()

```

Index

Integer ab 0



oder String Name oder ID des Elementes (analog zu NAME und ID-Attribut) muss in [] kodiert sein

Eigenschaften:

.name Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.readyState aktueller Status des Objektes beim Füllen des Objektes mit Daten
.urn Uniform Resource Name (URN) des Dokumentes

Methoden:

.attachEvent() Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider NICHT verkettet sondern in Zufallsfolge, es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.detachEvent() Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.doImport() Start des Import des Programmcodes aus einer *.htc-Datei eines Behavior (Verhaltens) eines Elementes / Objektes

4.3.2.2.4.2.2.4.5. document.namespaces Collection des Internet Explorer

Feld der Zeiger aller namespace Objekte

Syntax:

```
[ var ZeigerAufFeld = ] document.namespaces
[ var ZeigerAufFeldElement = ] document.namespaces [Index]
```

Index Integer ab 0
oder String Name oder ID des Elementes (analog zu NAME und ID-Attribut) muss in [] kodiert sein

ZeigerAufFeldElement ist null, wenn Feldelement nicht vorhanden

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.add() ein bereits erzeugtes Element einer Collection hinzufügen hinzufügen erst nach dem kompletten Laden des Dokumentes Element erzeugen per Methode .createElement()
.item() Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!

Beispiel:

```
<HTML XMLNS:MeinNamensRaum1
XMLNS:MeinNamensRaum2
XMLNS:MeinNamensRaum3
>
<HEAD>
<?IMPORT NAMESPACE="MeinNamensRaum2" IMPLEMENTATION="namespace1.htc">
<?IMPORT NAMESPACE="MeinNamensRaum2" IMPLEMENTATION="namespace2.htc">
<?IMPORT NAMESPACE="MeinNamensRaum3" IMPLEMENTATION="namespace3.htc">

<SCRIPT>
function TabelleDynamischErweitern()
{
    var TabellenZeile;
    var TabellenZelleDerZeile;

    for(var Index = 0; Index < document.namespaces.length; Index ++)
    {
        TabellenZeile = ID_Table.insertRow();

        // Zelle 1 mit Index
        TabellenZelleDerZeile = TabellenZeile.insertCell();
        TabellenZelleDerZeile.align = 'center';
        TabellenZelleDerZeile.innerText = Index; // oder .toString()

        // Zelle 2 mit Item-Methode
```



```

document.namespaces.item(iIndex).name
TabellenZelleDerZeile      = TabellenZeile.insertCell();
TabellenZelleDerZeile.align = 'center';
TabellenZelleDerZeile.innerHTML = document.namespaces.item[Index].name;

// Zelle 3 mit Index
TabellenZelleDerZeile      = TabellenZeile.insertCell();
TabellenZelleDerZeile.align = 'center';
TabellenZelleDerZeile.innerHTML = document.namespaces[Index].name;
    }
}
</SCRIPT>
</HEAD>
<BODY>
    <TABLE ID="ID_Table" BORDER=1>
        <TR>
            <TH>Zelle 1 Index</TH>
            <TH>Zelle 2 Feldelement per Item-Methode</TH>
            <TH>Zelle 3 Feldelement per Index</TH>
        </TR>
    </TABLE>
    <BR>
    <BUTTON onclick="TabelleDynamischErweitern();">Behavior anzeigen</BUTTON>
</BODY>
</HTML>

```

4.3.2.2.4.2.2.5. Filter eines HTML-Elementes im Internet Explorer (filter Objekt des Internet Explorer)

4.3.2.2.4.2.2.5.1. Einführung

Der Begriff Filter ist einem typischen Anwendungsbeispiel entliehen: Bekanntlicherweise sind Bildeffekte durch Herausfilterung von Bildkomponenten wie Farbe etc. sehr beliebt, so wie es auch ein Bildverarbeitungsprogramm mit Filter-Plugins anbietet. Als dynamisches Ergebnis der Bildverarbeitung ist z.B. die GIF-Datei (*.gif) zu nennen, deren Animationsumfang aber begrenzt ist.

Der Internet Explorer unterstützt dynamische Filter zur Laufzeit des HTML-Dokumentes, wobei nicht **nur** Bild-Objekte animiert werden können. Selbst auf den Einsatz von GIF-Dateien kann verzichtet werden.

Filter sind ebenfalls zusätzliche Verhaltensweisen von HTML-Elementen.

Filter sind nicht per externer Datei definierbar, sondern komplett im Browser implementiert.

Der Internet Explorer bietet Filter in verschiedenen Formen an:

- Objekt filter Es muss vom Objekt, das animiert werden soll, unterstützt werden.
Ein Objekt hat einen genau definierten Umfang an unterstützten Filtern, z.T. gar keine Filterunterstützung
- Objekt .style.time2.transitionFilter Behavior-Objekt des Standard-Beavior .style.time2
Dieser Filter unterstützt diverse HTML-Elemente, auch diejenigen, die das filter Objekt nicht unterstützen.

Nachfolgend wird das **Objekt filter** beschrieben (style.time2.transitionFilter siehe dort)

Das Objekt filter ist ein symbolisches Objekt. Es umfasst diverse Filterarten, deren wichtigsten nachfolgend beschrieben werden. Filter dienen zur sichtbaren Animation eines Objektes, das selbst **sichtbar sein muss**.

Nachteile von Filtern:

Filter benutzen z.T. intensiv CPU-Ressourcen, z.B. der Fade-Filter (Blenden von Bildern). Da dem Programmierer nicht bekannt ist, welche CPU der User gerade benutzt, sind Filter mit Vorsicht einzusetzen. Am Beispiel des Fade-Filters sei genannt, dass eine zeitgenaue Programmierung unmöglich ist, da die Abarbeitungsgeschwindigkeit des Filters neben den Zeitparametern auch von der CPU und sogar von der Version des Betriebssystems Windows abhängt (WXP rendert zügiger als W9x bei identischer CPU und Browserversion).

Die Synchronisierung des Filters z.B. mit Sound und ausgewählten Soundteilen ist aus o.g. Grund unmöglich, da es passieren kann, dass der Sound "schneller" ist. Mit anderen Worten: Der Fade-Filter braucht mehr Zeit, als seine Zeitparameter vorgeben, und ist somit nicht mehr synchron zu den Zeitparametern, die durch den Programmierer mit der Sounddauer abgestimmt sind. Daher sollte dann auf den Fade-Filter verzichtet und der Einsatz von transparenten Gif-Bildern oder einer in JScript animierten Bildfolge mit Soundsynchronisation z.B. per setTimeout() überdacht werden. Filter ist daher nur ein Dekorationselement.

Für ein Objekt, das einen Filter erhalten soll,

- muss per CSS mindestens eines der nachfolgenden Attribute kodiert sein: hihgt, width, position
wobei position absolute oder relative sein kann (falls nicht anderweitig festgelegt auf absolute).
- muss die Eigenschaft auf "tb-rl" oder Eigenschaft auf true gesetzt sein.



Folgende Objekte können **keinen** Filter bekommen: embed, applet, select im Formular, option im Formular, tr, thead, tbody, tfoot

Eine Zuordnung von Filtern zu den HTML-Elementen ist weiter unten zu ersehen.

Event ist onfilterchange.

dokumentbezogener Filter:

Variante 1 per Meta-Tag
möglich für Page-Enter, Page-Exit, Site-Enter, Site-Exit.

Beispiele

```
<META http-equiv="Page-Enter" CONTENT="progid:DXImageTransform.Microsoft.Blinds(Duration=4)" >
<META http-equiv="Page-Exit"
CONTENT="progid:DXImageTransform.Microsoft.Slide(Duration=2.500,slidestyle='HIDE')">
```

Variante 2 per HTML-Element-Tag mit STYLE-Attribut

BODY-Objekt kann Filter bekommen.

Beispiel

```
<BODY MARGINHEIGHT="0" TOPMARGIN="0" LEFTMARGIN="0" >
  <DIV STYLE="width:100%;height:100%;
              filter: progid:DXImageTransform.Microsoft.gradient
              (startColorstr=#550000FF, endColorstr=#55FFFF00)"
  >
  .....
</DIV>
</BODY>
```

Hinweise zum Kodieren von Filtern:

Wenn mehrere Filter für ein Objekt kodiert werden, so entspricht ist die Abarbeitungsfolge der der Filterfolge laut Kodierung.

Filter immer als letzte Eigenschaft des Objektes kodieren

Für die Ausführung eines Filters muss das Objekt sichtbar sein (.style.visibility).

ELEMENT in der Tag-Anweisung ist durch ein konkretes Tag zu ersetzen.

object in der Script-Anweisung ist durch die konkrete Referenz auf ein Objekt zu ersetzen.

Syntax im IE 4.0 <Element STYLE="filter:filtername(Kette)" >

Syntax im IE 5.5 <Element STYLE="filter: progid:DXImageTransform.Microsoft.filtername(Kette)" >

Kette String mit Liste aus den Eigenschaften des jeweiligen Filters
filtername laut Filterart

Nachfolgende Filterbeschreibungen verwenden den Syntax ab IE 5.5.

Der Syntax ab IE 5.5 macht deutlich, wie Filter realisiert werden: Als windows-eigene Komponente.

Filterreferenz per Collection filters:

Beispiel

```
function setfilter()
{
  myimage.filters.item("DXImageTransform.Microsoft.alpha").opacity = document.forms(0).opacity.value
  myimage.filters.item("DXImageTransform.Microsoft.alpha").finishOpacity = document.forms(0).finishOpacity.value
  myimage.filters.item("DXImageTransform.Microsoft.alpha").style = document.forms(0).setstyle.value
}
```

Filterbezeichner vom IE 4.x und ab IE 5.5:

4.0 filter name	Alternate 5.5 implementation
alpha	DXImageTransform.Microsoft.Alpha
BlendTrans	DXImageTransform.Microsoft.Fade
blur	DXImageTransform.Microsoft.MotionBlur
chroma	DXImageTransform.Microsoft.Chroma
dropShadow	DXImageTransform.Microsoft.DropShadow
FlipH	DXImageTransform.Microsoft.BasicImage(rotation=2, mirror=1)
FlipV	DXImageTransform.Microsoft.BasicImage(mirror=1)
glow	DXImageTransform.Microsoft.Glow



Gray	DXImageTransform.Microsoft.BasicImage(grayscale=1)
Invert	DXImageTransform.Microsoft.BasicImage(invert=1)
light	DXImageTransform.Microsoft.Light
mask	DXImageTransform.Microsoft.MaskFilter
shadow	DXImageTransform.Microsoft.Shadow
wave	DXImageTransform.Microsoft.Wave
Xray	DXImageTransform.Microsoft.BasicImage(xray=1)
RevealTrans	siehe unten

Filter RevealTrans bei IE 4.x und ab IE 5.5:

<u>numerischer Wert der Eigenschaft transition</u>	<u>alternative Kodierung ab IE 5.5</u>
0 - Box von außen nach innen	DXImageTransform.Microsoft.Iris(irisstyle='SQUARE', motion='in')
1 - Box von innen nach außen	DXImageTransform.Microsoft.Iris(irisstyle='SQUARE', motion='out')
2 - Circle von außen nach innen	DXImageTransform.Microsoft.Iris(irisstyle='CIRCLE', motion='in')
3 - Circle von innen nach außen	DXImageTransform.Microsoft.Iris(irisstyle='CIRCLE', motion='out')
4 - Wischen nach oben	DXImageTransform.Microsoft.Blinds(direction='up', bands=1)
5 - Wischen nach unten	DXImageTransform.Microsoft.Blinds(direction='down', bands=1)
6 - Wischen nach rechts	DXImageTransform.Microsoft.Blinds(direction='right', bands=1)
7 - Wischen nach links	DXImageTransform.Microsoft.Blinds(direction='left', bands=1)
8 - vertikale Jalousie	DXImageTransform.Microsoft.Blinds(direction='right')
9 - horizontale Jalousie	DXImageTransform.Microsoft.Blinds(direction='down')
10 - Schachbrett von links oben nach rechts unten	DXImageTransform.Microsoft.CheckerBoard(direction='right')
11 - Schachbrett von oben nach unten	DXImageTransform.Microsoft.CheckerBoard(direction='down')
12 - zufällige Auflösung	DXImageTransform.Microsoft.RandomDissolve
13 - Split vertikal von außen nach innen	DXImageTransform.Microsoft.Barn(orientation='vertical', motion='in')
14 - Split vertikal von innen nach außen	DXImageTransform.Microsoft.Barn(orientation='vertical', motion='out')
15 - Split horizontal von außen nach innen	DXImageTransform.Microsoft.Barn(orientation='horizontal', motion='in')
16 - Split horizontal von innen nach außen	DXImageTransform.Microsoft.Barn(orientation='horizontal', motion='out')
17 - Streifen von rechts oben nach links unten	DXImageTransform.Microsoft.Strips(motion='leftdown')
18 - Streifen von rechts unten nach links oben	DXImageTransform.Microsoft.Strips(motion='leftup')
19 - Streifen von links oben nach rechts unten	DXImageTransform.Microsoft.Strips(motion='rightdown')
20 - Streifen von links unten nach rechts oben	DXImageTransform.Microsoft.Strips(motion='rightup')
21 - Streifen horizontal zufällig	DXImageTransform.Microsoft.RandomBars(orientation='horizontal')
22 - Streifen vertikal zufällig	DXImageTransform.Microsoft.RandomBars(orientation='vertical')
23 - Zufallsauswahl aus einem der Effekte 0 bis 22	

Filter-Attribut des Style-Objekt:

.filter	Filter des Internetexplorer (siehe Objekt Filter) Syntax: <pre>object.style.filter = Kette [var Kette =] object.style.filter Kette Filter (siehe Objekt Filter)</pre> lesen und schreiben schreiben: immer überschreiben oder hinzufügen wird danach sofort geparkt: nachträgliches schreiben kostet Performance !!
Beispiel	Filter lesen <pre><SCRIPT> alert(ID_Div.style.filter); // zeigt den String an // "filter:progid:DXImageTransform.Microsoft.Wave(strength=100) // progid:DXImageTransform.Microsoft.CheckerBoard(duration=4)" </SCRIPT> <DIV ID="ID_Div" STYLE="height:50;width:50; filter:progid:DXImageTransform.Microsoft.Wave(strength=100) progid:DXImageTransform.Microsoft.CheckerBoard(duration=4)" > Test-Text </DIV></pre>
Beispiel	Filter schreiben <pre><DIV ID="ID_Div" STYLE="height:50;width:50;filter: progid:DXImageTransform.Microsoft.Wave(strength=100) progid:DXImageTransform.Microsoft.BasicImage(rotation=2, mirror=1)" > Test-Text </DIV> <SCRIPT> // hinzufügen ID_Div.style.filter += "progid:DXImageTransform.Microsoft.Iris(irisstyle='STAR',duration=4)"</pre>



</SCRIPT>

4.3.2.2.4.2.2.5.2. filters Collection des Internet Explorer

Feld aller Filter vom Objekt

Syntax:

```
[ var ZeigerAufFeld = ] object.filters
[ var ZeigerAufFeldElement = ] object.filters[Index, SubIndex]

Index          Integer ab 0
                oder String Name oder ID des Elementes
                (analog zu NAME und ID-Attribut)
                muss in [ ] kodiert sein

SubIndex       optional
                Integer
                Unterindex also Unterelement eines Elementes
                nur kodieren wenn Index ein String ist

ZeigerAufFeldElement
                ist null, wenn Feldelement nicht vorhanden
```

Beispiel 1

```
<IMG SRC="test.jpg"
STYLE="filter: progid:DXImageTransform.Microsoft.BasicImage(rotation=2, mirror=1)
progid:DXImageTransform.Microsoft.Alpha(opacity=50)
progid:DXImageTransform.Microsoft.Blur(strength=10); position: relative"
>
<SCRIPT LANGUAGE="JavaScript">
    sample.filters.item(0).enabled = 1 // Numerischer Index
    sample.filters.item("DXImageTransform.Microsoft.Alpha").enabled = 0 // Namen-Index
</SCRIPT>
```

Beispiel 2

```
function setfilter()
{
    myimage.filters.item('DXImageTransform.Microsoft.alpha').opacity = document.forms(0).opacity.value
    myimage.filters.item('DXImageTransform.Microsoft.alpha').finishOpacity = document.forms(0).finishOpacity.value
    myimage.filters.item('DXImageTransform.Microsoft.alpha').style = document.forms(0).setstyle.value
}
```

Eigenschaften:

.length Anzahl der Feldelemente also Feldlänge z.B. bei Collection

Methoden:

.item() Referenz auf Feldelement anhand des Integer-Indexes oder des Attributnamen (analog zu ID oder NAME-Attribut) liefern außer bei Formular mit <INPUT TYPE=image ...> da dafür die children-Collection verwendet werden muss !!!

.namedItem() Referenz auf Eintrag (FeldElement) anhand des Namen (analog zu ID oder NAME-Attribut) liefern

4.3.2.2.4.2.2.5.3. Filtereigenschaften

Nachfolgende Eigenschaften werden nicht von allen Filtern benötigt. Eigenschaften werden z.T. mit identischen Bezeichnern anders wertmässig belegt (je nach Filter).

.add Überlagerung des Filters über das Originalbild

.bands Anzahl der Streifen

.Bias Prozentsatz der Farbwerterhöhung je höher der Bildkontrast um so geringer der sichtbare Effekt

.color Farbe

.colorSpace Pfad der *.icm-Datei

.direction Filterrichtung nicht Filter Blinds und CheckerBoard

Beispiel

```
<SCRIPT>
function Setze()
{
    with (window.event.srcElement.filters[0])
    {
        if (strength < 100)
        {
            strength += 1;
            direction += 45;
        }
    }
}
```



```

    }
  }
</SCRIPT>
<IMG SRC="test.gif" onfilterchange="Setze()"
STYLE="filter:progid:DXImageTransform.Microsoft.MotionBlur(STRENGTH=1, DIRECTION=0)"
>
.Direction          Richtung
                    nur Filter Blinds und CheckerBoard
.duration           Dauer der Animation des Filters
Beispiel
<SCRIPT LANGUAGE=JavaScript>
var StartBild = "/graphics/test1.jpg";
var EndeBild = "/graphics/test2.jpg";

function FilterAusfuehren()
{
    ID_Img.filters.item(0).Apply();
    ID_Img.src = EndeBild;
    EndeBild = StartBild;      // neues Endebild
    StartBild = ID_Img.src;    // altes Endebild
    ID_Img.filters.item(0).Play();
}
</SCRIPT>
<IMG ID="ID_Img" SRC="/graphics/test1.jpg";
style="width:200; height:200; filter:progid:DXImageTransform.Microsoft.Blinds(Duration=2)"
>
<BR>
<INPUT TYPE="button" value="Start Transition" onClick="FilterAusfuehren()">
.Direction          X-Komponente des Vektor der linearen Transformation
                    wird ignoriert wenn Eigenschaft .SizingMethod mit Wert "auto expand"
.Dy                 Y-Komponente des Vektor der linearen Transformation
                    wird ignoriert wenn Eigenschaft .SizingMethod mit Wert "auto expand"
.enabled           Filter-Anwendbarkeit
Beispiel
<IMG ID="ID_Img"
SRC="/graphics/test.gif"
STYLE="filter:blur(STRENGTH=50) flipv()"
onmouseover="ID_Img.filters.flipv.enabled = false;"
onmouseout = "ID_Img.filters.flipv.enabled = true;"
>
.EndColor          Ende-Deckungskraft als Integer
.EndColorStr       Ende-Deckungskraft als String
.FilterType        Filtertyp
.finishOpacity     Deckungskraft, mit der Filter endet
.finishX           Horizontale Position als Prozent der Objektbreite in der Filter mit Deckungskraft endet
.finishY           Vertikale Position als Prozent der Objekthöhe in der Filter mit Deckungskraft endet
.freq             Anzahl der Wellen
.function          Nummer der Komposition
.gradientSize      Faktor der Objektbreite als Gradientbreite
.GradientType      Orientierung des Gradienten
.grayScale         Grauskala ein/aus
Beispiel
<DIV ID="ID_Div" STYLE="position:absolute; left:270px;" >
<IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(grayScale=1)'">
  Gray
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter=''">
  Clear Filter
</BUTTON>
.gridSizeX         Anzahl Spalten des Gitters
.gridSizeY         Anzahl Zeilen des Gitters
.intent            Farbverwendungsart
.Invert           Komplementärfarben ein/aus
Beispiel
<DIV ID="ID_Div" STYLE="position:absolute; left:270px;" >
<IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(invert=1)'">

```



```

        Invert
    </BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter="">
    Clear Filter
</BUTTON><BR>
.irisStyle          Schärfe des Filters
.lightStrength     prozentuale Differenz der Licht-Intensität zwischen Wellengipfel und Wellentäler
.M11              Matrixfeld M11
.M12              Matrixfeld M12
.M21              Matrixfeld M21
.M22              Matrixfeld M22
.makeShadow       Schatten ein/aus

```

Beispiel

```

<DIV STYLE="position:absolute; left:70px; filter:
    progid:DXImageTransform.Microsoft.blur(pixelradius=3.0, makeshadow='true',
    ShadowOpacity=1.0)"
>
    <IMG SRC="/test.gif" >
</DIV>

```

.Mask Beispiel Transparenzänderung auf Wert laut Eigenschaft .MaskColor ein/aus

```

<DIV ID="ID_Div" STYLE="position:absolute; left:270px;filter:
    progid:DXImageTransform.Microsoft.BasicImage(mask=1)" >
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0Xff0000">
    Red Mask
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0X00ff00">
    Green Mask
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0X0000ff">
    Blue Mask
</BUTTON>

```

.MaskColor Farbmaske

Beispiel

```

<DIV ID="ID_Div" STYLE="position:absolute; left:270px;filter:
    progid:DXImageTransform.Microsoft.BasicImage(mask=1)" >
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0Xff0000">
    Red Mask
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0X00ff00">
    Green Mask
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.filters.item(0).MaskColor = 0X0000ff">
    Blue Mask
</BUTTON>

```

```

.maxSquare       Maximale Anzahl der Pixels im Quadrat
.mirror          Rervers ein/aus

```

Beispiel

```

<DIV ID="ID_Div" STYLE="position:absolute; left:270px;" >
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(mirror=1)'">
    Rotate 270 degrees
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter="">Clear Filter</BUTTON>

```

```

.motion          Bewegungsrichtung
                nicht Filter GradientWipe und Strips
.motion          Art der Bewegung
                nur Filter GradientWipe

```



.motion Ecke
 nur Filter Strips
 .offX horizontaler Abstand des Schattens vom Objekt UND Schattenrichtung
 .offY vertikaler Abstand des Schattens vom Objekt UND Schattenrichtung
 .opacity Deckungskraft mit der der Filter startet
 nicht Filter BasicImage
 .opacity Deckungskraft- bz.w Transparenz-Niveau (Opacity-Niveau)
 nur Filter BasicImage

Beispiel

```
<DIV ID="ID_Div" STYLE="position:absolute; left:270px;" >
  <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(opacity=.2)'">
  Opaque
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter=''">Clear Filter</BUTTON>
```

Klick Opaque-Button für Transparenz

.orientation Orientierung
 .overlap Anteil der Überlappungs-Dauer am Wert laut Eigenschaft .duration

Beispiel

wenn .overlap auf 0.4 und .duration auf 10 Sekunden
 dann 40 % der 10 Sekunden findet überlappen statt = 4 Sekunden
 60 % der 10 Sekunden ohne überlappen = 6 Sekunden, also
 30 % der 10 Sekunden nur Fadeout = 3 Sekunden
 30 % der 10 Sekunden nur Fadein = 3 Sekunden
 also Ablauf:
 Sekunde 1-3 Fadeout aber nicht komplett
 Sekunde 4-7 Fadeout komplett UND Fadein beginnt = Overlap
 Sekunde 8-10 Fadein komplett
 .Percent Prozentualer Umfang der Anzeige mit dem der Filter enden soll

Beispiel

```
<SCRIPT>
function FilterSetzen()
{
  ID_Div.filters[0].Apply();

  // Achtung: nach Applay() kodierte Verämderungen vom DIV werden ERST
  // mit der Abarbeitung der Methode .play() sichtbar
  ID_Div.style.backgroundColor="blue";
  ID_Div.filters[0].percent=50;
  ID_Div.filters[0].enabled=true;
}
</SCRIPT>
<BUTTON onclick="FilterSetzen(); onclick=''">FilterSetzen</BUTTON>
<BR>
<DIV ID="ID_Div"
  STYLE="height:250px; width:250px; background-color: gold;
  filter:progid:DXImageTransform.Microsoft.checkerboard(duration=5, transition=7);"
  >
</DIV>
```

.phase prozentuale Phasenverschiebung der Wellen,
 also Verschiebung der Wellenüberlagerung als Prozentanteil des Wellenzyklus
 .pixelRadius Radius des Blur Filter-Raumes

Beispiel

```
<DIV STYLE="position:absolute; left:70px; filter:
  progid:DXImageTransform.Microsoft.blur(pixelradius=3.0, makeshadow='true',
  ShadowOpacity=1.0)"
  >
  <IMG SRC="/test.gif" >
</DIV>
```

.positive Schattentransparenz
 .Rotation Faktor der Rotation in 90-Grad-Schritten

Beispiel

```
<DIV ID="ID_Div" STYLE="position:absolute; left:270px;" >
  <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(rotation=3)'">
  Rotate 270 degrees
</BUTTON>
```



```

<BR>
<BUTTON onclick="ID_Div.style.filter="">Clear Filter</BUTTON>
.shadowOpacity Schatten-Deckungskraft
                wirkt nur wenn Eigenschaft makeShadow auf true gesetzt ist
    Beispiel
<DIV STYLE="position:absolute; left:70px; filter:
                progid:DXImageTransform.Microsoft.blur(pixelradius=3.0, makeshadow='true',
                ShadowOpacity=1.0)"
>
                <IMG SRC="/test.gif" >
</DIV>
.sizingMethod Art der Anpassung
                nicht Filter Matrix
    Beispiel
<SCRIPT>
                function FilterAendern(ArtDerAnpassung)
                { ID_Div.filters(0).sizingMethod= ArtDerAnpassung;}
</SCRIPT>
<DIV ID="ID_Div"
                STYLE="position:absolute; left:140px; height:150; width:200;
                filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(
                SRC='test.jpg', sizingMethod='scale');"
>
</DIV>
<BUTTON onclick=" FilterAendern('image');"> kein Anpassen, also Original-Dimension</BUTTON>
<BR>
<BUTTON onclick="FilterAendern ('scale');">Anpassen ohne Abschneiden</BUTTON>
<BR>
<BUTTON onclick=" FilterAendern('crop');">Anpassen durch Abschneiden</BUTTON>
.SizingMethod Container-Resize ein/aus
                nur Filter Matrix
.slideStyle Art
    Beispiel
<DIV STYLE="height:250px; width:250px; background-color: gold;
                filter:progid:DXImageTransform.Microsoft.Slide(
                duration=3, bands='8', slideStyle='PUSH');"
>
.spokes Anzahl Speichen
.squaresX Anzahl der Spalten
    Beispiel
<SCRIPT>
                var Flag = 0;

                function Aendern()
                {
                        ID_Div.filters[0].Apply();

                        if (Flag) // 1 entspricht true
                        {
                                Flag = 0;
                                ID_Div.style.backgroundColor="gold";
                        }
                        else // 0 entspricht false
                        {
                                Flag = 1;
                                ID_Div.style.backgroundColor="blue";
                        }

                        ID_Div.filters[0].Play();
                }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
                filter:progid:DXImageTransform.Microsoft.CheckerBoard(
                duration=5, direction='left', squaresX=4);">
</DIV>
.squaresY Anzahl der Zeilen
    Beispiel
<SCRIPT>

```



```

var Flag = 0;

function Aendern()
{
    ID_Div.filters[0].Apply();

    if (Flag) // 1 entspricht true
    {
        Flag = 0;
        ID_Div.style.backgroundColor="gold";
    }
    else // 0 entspricht false
    {
        Flag = 1;
        ID_Div.style.backgroundColor="blue";
    }

    ID_Div.filters[0].Play();
}
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
filter:progid:DXImageTransform.Microsoft.CheckerBoard(
duration=5, direction='left', squaresY=4);">
</DIV>

```

.SRC Beispiel Url des Image

```

<SCRIPT>
function FilterAendern(ArtDerAnpassung)
{ ID_Div.filters(0).sizingMethod= ArtDerAnpassung;}
</SCRIPT>
<DIV ID="ID_Div"
STYLE="position:absolute; left:140px; height:150; width:200;
filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(
SRC='test.jpg', sizingMethod='scale');"
>
</DIV>
<BUTTON onclick=" FilterAendern('image');"> kein Anpassen, also Original-Dimension</BUTTON>
<BR>
<BUTTON onclick="FilterAendern ('scale');">Anpassen ohne Abschneiden</BUTTON>
<BR>
<BUTTON onclick=" FilterAendern('crop');">Anpassen durch Abschneiden</BUTTON>

```

- .StartColor Start-Deckungskraft als Integer
- .StartColorStr Start-Deckungskraft als String
- .startX Horizontale Position als Prozent der Objektbreite in der Filter mit Deckungskraft startet
- .startY Vertikale Position als Prozent der Objekthöhe in der Filter mit Deckungskraft startet
- .status Status der Filteranimation
- .strength Ausdehnung des Glühens

Beispiel

```

<SCRIPT>
function FilterSetzen()
{
    with (window.event.srcElement.filters[0])
    {
        if (strength < 200)
        {
            strength += 1;
            direction += 45;
        }
    }
}
</SCRIPT>
<IMG SRC="test.gif"
STYLE="filter:progid:DXImageTransform.Microsoft.motionBlur(STRENGTH=1, DIRECTION=0)"
onfilterchange="FilterSetzen()"
>

```

.stretchStyle Art

Beispiel

```

<DIV STYLE="height:250px; width:250px; background-color: gold;

```



filter:progid:DXImageTransform.Microsoft.Stretch(duration=2, stretchStyle='PUSH');"

>
.style Art der Deckungskraft
.transition Filternummer

numerischer Wert der Eigenschaft transition alternative Kodierung ab IE 5.5

0 - Box von außen nach innen	DXImageTransform.Microsoft.Iris(irisstyle='SQUARE', motion='in')
1 - Box von innen nach außen	DXImageTransform.Microsoft.Iris(irisstyle='SQUARE', motion='out')
2 - Circle von außen nach innen	DXImageTransform.Microsoft.Iris(irisstyle='CIRCLE', motion='in')
3 - Circle von innen nach außen	DXImageTransform.Microsoft.Iris(irisstyle='CIRCLE', motion='out')
4 - Wischen nach oben	DXImageTransform.Microsoft.Blinds(direction='up', bands=1)
5 - Wischen nach unten	DXImageTransform.Microsoft.Blinds(direction='down', bands=1)
6 - Wischen nach rechts	DXImageTransform.Microsoft.Blinds(direction='right', bands=1)
7 - Wischen nach links	DXImageTransform.Microsoft.Blinds(direction='left', bands=1)
8 - vertikale Jalousie	DXImageTransform.Microsoft.Blinds(direction='right')
9 - horizontale Jalousie	DXImageTransform.Microsoft.Blinds(direction='down')
10 - Schachbrett von links oben nach rechts unten	DXImageTransform.Microsoft.CheckerBoard(direction='right')
11 - Schachbrett von oben nach unten	DXImageTransform.Microsoft.CheckerBoard(direction='down')
12 - zufällige Auflösung	DXImageTransform.Microsoft.RandomDissolve
13 - Split vertikal von außen nach innen	DXImageTransform.Microsoft.Barn(orientation='vertical', motion='in')
14 - Split vertikal von innen nach außen	DXImageTransform.Microsoft.Barn(orientation='vertical', motion='out')
15 - Split horizontal von außen nach innen	DXImageTransform.Microsoft.Barn(orientation='horizontal', motion='in')
16 - Split horizontal von innen nach außen	DXImageTransform.Microsoft.Barn(orientation='horizontal', motion='out')
17 - Streifen von rechts oben nach links unten	DXImageTransform.Microsoft.Strips(motion='leftdown')
18 - Streifen von rechts unten nach links oben	DXImageTransform.Microsoft.Strips(motion='leftup')
19 - Streifen von links oben nach rechts unten	DXImageTransform.Microsoft.Strips(motion='rightdown')
20 - Streifen von links unten nach rechts oben	DXImageTransform.Microsoft.Strips(motion='rightup')
21 - Streifen horizontal zufällig	DXImageTransform.Microsoft.RandomBars(orientation='horizontal')
22 - Streifen vertikal zufällig	DXImageTransform.Microsoft.RandomBars(orientation='vertical')
23 - Zufallsauswahl aus einem der Effekte 0 bis 22	

Beispiel

```

<SCRIPT>
function go()
{
    ID_Span.filters[0].Apply();

    if (ID_Span.style.visibility == "visible")
    {
        ID_Span.style.visibility = "hidden";
        ID_Span.filters.revealTrans.transition=2;
    }
    else
    {
        ID_Span.style.visibility = "visible";
        ID_Span.filters[0].transition=3;
    }

    ID_Span.filters[0].Play();
}
</SCRIPT>
<INPUT TYPE="button" VALUE="Play Transition" onClick="go();"></INPUT>
<SPAN ID="ID_Span" STYLE="position:absolute; visibility:visible;
filter:revealTrans(DURATION=2, TRANSITION=3);
width:300; height:300; background-color: lightgreen"

```

```

>
<CENTER>
<DIV STYLE="background-color:red;height=100;width:100;
position:relative;top:100
"
>
</DIV>
</CENTER>
</SPAN>

```

.WipeStyle Richtung nicht Filter RadialWipe
.wipeStyle Style des Wischens nur Filter RadialWipe
.Xray Grauskale aus Mittelwert vom Rot- und Grünanteil ein/aus

Beispiel

```

<DIV ID="ID_Div" STYLE="position:absolute; left:270px; color:tan;" >
<IMG SRC="/test.gif" >

```



```

</DIV>
<BUTTON onclick="ID_Div.style.filter= 'progid:DXImageTransform.Microsoft.BasicImage(XRay=1)'">
  Show Xray
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter="">Clear Filter</BUTTON>

```

4.3.2.2.4.2.2.5.4. Filtermethoden

.addAmbient()	indirektes Umgebungslicht (Ambiente)
.addCone()	direktes 3D-Licht als Lichttunnel oder Spot
.addPoint()	D-Lichtpunkt als Lichtquelle, die in alle Richtungen strahlt
.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart die Methode .play() benutzen (sichtbarmachen der Filtereinstellungen)
.changeColor()	Lichtfarbe nachträglich ändern
.changeStrength()	Lichtintensität nachträglich ändern
.clear()	alle Lichtquellen löschen
.moveLight()	Focus des Lichtkegels bzw. Lichtpunkt bewegen
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.5. Alpha Filter

Deckungskraft eines sichtbaren Objektes
ab IE 5.5

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Alpha(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Alpha(Kette)"

```

Kette String mit Eigenschaftenliste

Beispiel für transparentes INPUT:

```

<STYLE>
  INPUT.aFilter {filter:progid:DXImageTransform.Microsoft.Alpha(opacity=50);}
</STYLE>
<INPUT TYPE="button" VALUE="Button" CLASS="aFilter">

```

Beispiel für transparentes Bild auf einem Hintergrundbild:

```

<STYLE>
  IMG.filter_name {filter:Alpha(opacity=y);}
</STYLE>
<BODY BACKGROUND="hintergrund_bild.gif">
  <IMG SRC="bild.gif".... CLASS="filter_name">
</BODY>

```

Beispiel für Uhr:

```

<HTML>
<HEAD>
<SCRIPT LANGUAGE="JScript">
<!--
//          24-Stunden Digitaluhr mit Spiegelbild nuf für IE
//
// *****
//
//          vom Programmierer zu belegende Variablen
//
// *****
var Uhr_Frequenz=960; // Bitte an die echte Genauigkeit anpassen auf Sekundentakt

var Uhr_Breite      = 10; // in Pixel
var Uhr_Hoeh     = 10; // in Pixel
var Uhr_AbstandVomFensterRandOben = 10; // in Pixel
var Uhr_AbstandVomFensterRandLinks = 50; // in Pixel

var Uhr_UngespiegeltFontFamily      = "Comic Sans Ms";
var Uhr_UngespiegeltFontSize        = 14; // in pt
var Uhr_UngespiegeltFontStyle      = "italic";
var Uhr_UngespiegeltFontWeigth     = "bold";

```



```

var Uhr_UngespiegeltFontFarbe      = "#00008D";

var Uhr_GespiegeltFontFamily       = "Comic Sans Ms";
var Uhr_GespiegeltFontSize         = 14;                // in pt
var Uhr_GespiegeltFontStyle        = "italic";
var Uhr_GespiegeltFontWeigth       = "bold";
var Uhr_GespiegeltFontFarbe        = "#00008D";
var Uhr_GespiegeltDeckungskraft    = 11;                // je höher um so stärker die Farbe denkend
// je kleiner um so blasser die Farbe

var Uhr_GespiegeltAbstandVonUngespiegelt = Uhr_UngespiegeltFontSize + 6;

var LeeresBildUrl="1x1tran.gif"    // muss 1x1 Pixel sein, ist transparent

// *****
//
//      nachfolgenden Code nicht verändern
//
// *****
// Dokument mit leeren Bild instanzieren
//      Instanzierung erfolgt normalerweise ERST mit Lesen des BOD>-Tags, aber
//      das Dokument muss spätestens zur Ermittlung der Fensterdimensionen
//      per Javascript instanziiert sein, also auch in Javascript.
//      ACHTUNG: Der GESAMTE BODY-Teil des Dokumentes wird damit ignoriert und MUSS leer bleiben !

document.write("<IMG SRC=\"" + LeeresBildUrl + "\" HEIGHT=1 WIDTH=1>");

//      Prüfung auf Browsertyp
//      Dieser Quellcode muss VOR allen anderen Routinen codiert sein, damit zuerst abgearbeitet
var ns = document.layers ? true : false;
var ie = document.all ? true : false;

var TimeoutID;

function UhrAnzeigen()
{
    var DatumObjekt    = new Date();
    var Stunde         = DatumObjekt.getHours();
    var Minute         = DatumObjekt.getMinutes();
    var Sekunde        = DatumObjekt.getSeconds();

    if (Stunde < 10 )
    {Stunde="0" + Stunde;}

    if (Minute <10 )
    {Minute="0" + Minute;}

    if (Sekunde <10 )
    {Sekunde="0" + Sekunde;}

    DIV_ID_Uhr_ungespiegelt.innerHTML = Stunde + ":" + Minute + ":" + Sekunde;
    DIV_ID_Uhr_gespiegelt.innerHTML  = Stunde + ":" + Minute + ":" + Sekunde;

    TimeoutID=setTimeout('UhrAnzeigen()',Uhr_Frequenz);
}

if (ie)
{
    document.write(    '<DIV ID="DIV_ID_Uhr_ungespiegelt"
+      ' STYLE="position:absolute;
+          'width:'    + Uhr_Breite          + 'px;'
+          'height:'   + Uhr_Hoehe         + 'px;'
+          'top:'      + Uhr_AbstandVomFensterRandOben + 'px;'
+          'left:'     + Uhr_AbstandVomFensterRandLinks + 'px;'
+          'font-family:' + Uhr_UngespiegeltFontFamily + ','
+          'font-size:'  + Uhr_UngespiegeltFontSize + 'pt;'
+          'font-style:' + Uhr_UngespiegeltFontStyle + ','
+          'font-weight:' + Uhr_UngespiegeltFontWeigth + ','
+          'color:'     + Uhr_UngespiegeltFontFarbe + ','
+          ""
+      '>
+ '</DIV>'

```



```

    );
    document.write(
        '<DIV ID="DIV_ID_Uhr_gespiegelt"'
        + ' STYLE="position:absolute;'
        + ' width:' + Uhr_Breite + 'px;'
        + ' height:' + Uhr_Hoehe + 'px;'
        + ' top:'
        + (Uhr_AbstandVomFensterRandOben + Uhr_GespiegeltAbstandVonUngespiegelt)
        + 'px;'
        + ' left:' + Uhr_AbstandVomFensterRandLinks + 'px;'
        + ' font-family:' + Uhr_GespiegeltFontFamily + ';'
        + ' font-size:' + Uhr_GespiegeltFontSize + 'pt;'
        + ' font-style:' + Uhr_GespiegeltFontStyle + ';'
        + ' font-weight:' + Uhr_GespiegeltFontWeigth + ';'
        + ' color:' + Uhr_GespiegeltFontFarbe + ';'
        + ' filter:flipv() alpha(opacity=' + Uhr_GespiegeltDeckungskraft + ');'
        + ' >'
        + '</DIV>'
    );
    UhrAnzeigen();
}
//-->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>

```

Eigenschaften:

.enabled	Filter-Anwendbarkeit
.finishOpacity	Deckungskraft, mit der Filter endet
.finishX	Horizontale Position als Prozent der Objektbreite in der Filter mit Deckungskraft endet
.finishY	Vertikale Position als Prozent der Objekthöhe in der Filter mit Deckungskraft endet
.opacity	Deckungskraft mit der der Filter startet
.startX	Horizontale Position als Prozent der Objektbreite in der Filter mit Deckungskraft startet
.startY	Vertikale Position als Prozent der Objekthöhe in der Filter mit Deckungskraft startet
.style	Art der Deckungskraft

Methoden:

keine

4.3.2.2.4.2.2.5.6. *AlphaImageLoader Filter*

Imageanpassung an Container-Dimension

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.AlphaImageLoader(Kette)"

```

Kette String mit Eigenschaftenliste

Eigenschaften:

.enabled	Filter-Anwendbarkeit
.SRC	Url des Image
.sizingMethod	Art der Anpassung

Methoden:

keine

4.3.2.2.4.2.2.5.7. *Barn Filter*

Scheunentür-Animation (öffnen/schliessen wie Türen)

ab IE 5.5

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Barn(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Barn(Kette)"

```

Kette String mit Eigenschaftenliste

Beispiel

```

<SCRIPT>
    var Flag = 0;

    function Aendern()
    {
        // Filter init
        ID_Div.filters[0].Apply();
    }

```



```

// Filter ändern
if (Flag) // true entspricht numerisch 1
{
    Flag = 0;
    ID_Div.style.backgroundColor="gold";
}
else // false entspricht numerisch 0
{
    Flag = 1;
    ID_Div.style.backgroundColor="blue";
}

// Filter animieren
ID_Div.filters[0].Play();
}
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
filter:progid:DXImageTransform.Microsoft.Barn(
duration=2, motion='out', orientation='vertical');"
>
</DIV>

```

Eigenschaften:

- .duration Dauer der Animation des Filters
- .enabled Filter-Anwendbarkeit
- .motion Bewegungsrichtung
- .orientation Orientierung
- .Percent Prozentualer Umfang der Anzeige mit dem der Filter enden soll
- .status Status der Filteranimation

Methoden:

- .apply() Initialisiert den Filter und setzt Anzeige zurück
nach apply sind die Filtereinstellungen zu kodieren
für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
- .play() Animiert den Filter in seinen aktuellen Einstellungen
benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
- .stop() Stop der Filteranimation vor dem normalen Ende
erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.8. BasicImage Filter

Farbverläufe, Bildrotation, Deckkraft
ab IE 5.5

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.BasicImage(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.BasicImage(Kette)"

```

Beispiel Kette String mit Eigenschaftenliste

```

<STYLE>
    button {width:250px;}
</STYLE>
<DIV ID="ID_Div" oDiv" STYLE="position:absolute; left:270px;" >
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(mirror=1)'">
    Mirror
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(invert=1)'">
    Invert
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(grayScale=1)'">
    GrayScale
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(xray=1)'">
    X-Ray
</BUTTON>
<BR>
<BUTTON onclick="ID_Div.style.filter='progid:DXImageTransform.Microsoft.BasicImage(mask=1,

```



```

                                maskColor=255)"">
                                Blue MaskColor
                                </BUTTON>
                                <BR>
                                <BUTTON onclick=" ID_Div.style.filter="">
                                Clear Filter
                                </BUTTON>
                                <BR>

```

Eigenschaften:

.enabled	Filter-Anwendbarkeit
.grayScale	Grauskala ein/aus
.Invert	Komplementarfarben ein/aus
.Mask	Transparenzänderung auf Wert laut Eigenschaft .MaskColor ein/aus
.MaskColor	Farbmaske
.mirror	Rervers ein/aus
.opacity	Deckungskraft- bz.w Transparenz-Niveau (Opacity-Niveau)
.Rotation	Achtung: nur die Eigenschaft vom Filter BasicImage
.Xray	Faktor der Rotation in 90-Grad-Schritten
.Xray	Grauskale aus Mittelwert vom Rot- und Grünanteil ein/aus

Methoden:

keine

4.3.2.2.4.2.2.5.9. *BlendTrans Filter*

Blenden-Filter (Fading) für ein sichtbares Objekt

ab IE 4.x

Syntax:

```

HTML <ELEMENT STYLE="filter:BlendTrans(Kette)" ... >
Scripting object.style.filter ="BlendTrans(Kette)"

```

Beispiel
Kette String mit Eigenschaftenliste

```

<HEAD>
<SCRIPT>
function fadeOut()
{
    ID_Div.style.filter="blendTrans(duration=2)";
    // prüfen ob Filter nicht gerade animiert
    if (ID_Div.filters.blendTrans.status != 2)
    {
        // Filter animiert nicht
        ID_Div.filters.blendTrans.apply();
        ID_Div.style.visibility="hidden";
        ID_Div.filters.blendTrans.play();
    }
}

function fadeIn()
{
    ID_Div.style.filter="blendTrans(duration=2)";
    // prüfen ob Filter nicht gerade animiert
    if (ID_Div.filters.blendTrans.status != 2)
    {
        // Filter animiert nicht
        ID_Div.filters.blendTrans.apply();
        ID_Div.style.visibility="visible";
        ID_Div.filters.blendTrans.play();
    }
}
</SCRIPT>
<HEAD>
<BODY>
<!-- DIV muss mit Layout deklariert sein z.B. Breite -->
<DIV ID="ID_Div" STYLE="width: 200">
    Text fuer faded in und out.
</DIV>
<P>
    <BUTTON onclick="fadeOut()">Fade Text Out</BUTTON>
    <BUTTON onclick="fadeIn()">Fade Text In</BUTTON>
</P>
</BODY>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
-----------	---------------------------------



.enabled	Filter-Anwendbarkeit
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation
Methoden:	
.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.10. Blinds Filter

Blendenfilter (Blenden öffnen/schliessen)

ab IE 5.5

Syntax:

```
HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Blinds(Kette)" ... >
Scripting object.style.filter = "progid:DXImageTransform.Microsoft.Blinds(Kette)"
```

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT>
var Flag = 0;
function Aendern()
{
    // Filter init
    ID_Div.filters[0].Apply();
    // Anzahl der Streifen zufällig ermitteln
    ID_Div.filters[0].bands = Math.random()*12 + 3;
    // Filter ändern
    if (Flag) // 1 entspricht true
    {
        Flag = 0;
        ID_Div.style.backgroundColor="gold";
    }
    else // 0 entspricht false
    {
        Flag = 1;
        ID_Div.style.backgroundColor="blue";
    }
}

// Filter starten
ID_Div.filters[0].Play(duration=3);
}
</SCRIPT>
<BUTTON onclick="Aendern ()">Aendern</BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
filter:progid:DXImageTransform.Microsoft.Blinds(direction='down');"
>
</DIV>
```

Eigenschaften:

.bands	Anzahl der Streifen
.Direction	Richtung nur Filter Blinds und CheckerBoard
.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.11. Blur Filter

Filter für Objekt, wenn es keinen Focus hat



ab IE 5.5

Syntax:

```
HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Blur(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Blur(Kette)"
```

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT LANGUAGE="JScript">
    function Aendern(ButtonZeiger)
    {
        if (ID_Div.filters(0).enabled)
        {
            ID_Div.filters(0).enabled='false';
            ButtonZeiger.innerText='blur';
        }
        else
        {
            ID_Div.filters(0).enabled='true';
            ButtonZeiger.innerText='normal';
        }
    }
</SCRIPT>
<DIV ID="ID_Div" STYLE="position:absolute; left:270px; filter:
    progid:DXImageTransform.Microsoft.blur(pixelradius=3, enabled='false')"
>
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="Aendern(this)">blur</BUTTON><BR>
```

Eigenschaften:

.enabled	Filter-Anwendbarkeit
.makeShadow	Schatten ein/aus
.pixelRadius	Radius des Blurfilter-Raumes
.shadowOpacity	Schatten-Deckungskraft
	wirkt nur wenn Eigenschaft makeShadow auf true gesetzt ist

Methoden:

keine

4.3.2.2.4.2.2.5.12. CheckerBoard Filter

Schachbrettfiter

ab IE 5.5

Syntax:

```
HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.CheckerBoard(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.CheckerBoard(Kette)"
```

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT>
    var Flag = 0;

    function Aendern()
    {
        ID_Div.filters[0].Apply();

        if (Flag) // 1 entspricht true
        {
            Flag = 0;
            ID_Div.style.backgroundColor="gold";
        }
        else // 0 entspricht false
        {
            Flag = 1;
            ID_Div.style.backgroundColor="blue";
        }

        ID_Div.filters[0].Play();
    }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.CheckerBoard(
    duration=5, direction='left');">
```



</DIV>

Eigenschaften:

.Direction	Richtung
	nur Filter Blinds und CheckerBoard
.squaresX	Anzahl der Spalten
.squaresY	Anzahl der Zeilen

Methoden:

keine

4.3.2.2.4.2.2.5.13. Chroma Filter

Anzeige einer Farbe aus der Farbtiefe als transparent
 wenn Farbe bereits transparent so durch Filter als deckend erzeugt
 bei JPG-Files kaum wirksam !!!!, da Farbtiefe reduziert ist
 bei Rand-geglätteten Objekten nicht wirksam, da Glättung durch Anpassung an umgebende Pixel-Farbwerte erfolgt
 (Verschwommenheit, Weichmachen)

ab IE 5.5

Syntax:

HTML	<ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Chroma(Kette)" ... >
Scripting	object.style.filter ="progid:DXImageTransform.Microsoft.Chroma(Kette)"

Beispiel

```

Kette    String mit Eigenschaftenliste
<BUTTON onclick=" ID_Div.filters.item('DXImageTransform.Microsoft.Chroma').color = 'gray';">
  grau zu blau
</BUTTON>
<BR>
<BUTTON onclick=" ID_Div.filters.item('DXImageTransform.Microsoft.Chroma').color = 'blue';">
  blau zu grau
</BUTTON>
<DIV ID="ID_Div" STYLE="position:absolute; top:100; left:10; width:240; height:100;
  filter:progid:DXImageTransform.Microsoft.Chroma(color='yellow'),
  progid:DXImageTransform.Microsoft.Chroma(color='red')"
>
  <BR>
  <SPAN STYLE="position:absolute; top:50; left:10; width:240; height:60;color:gray;">
    grau
  </SPAN>
  <BR>
  <SPAN STYLE="position:absolute; top:70; left:10; width:240; height:60; color:blue;">
    blau
  </SPAN>
</DIV>

```

Eigenschaften:

.color	Farbe
.enabled	Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.14. Compositor Filter

Farbkombination

ab IE 5.5

Syntax:

HTML	<ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Compositor(Kette)" ... >
Scripting	object.style.filter ="progid:DXImageTransform.Microsoft.Compositor(Kette)"

Kette String mit Eigenschaftenliste

Ablauf:

Funktion wählen	
Input A erzeugen	Image
	per Methode .apply()
Input B erzeugen	Image
	Eigenschaften des Objektes (eventuelle von Kindern des Objektes) verändern
	z.B. visibility, innerText, backgroundColor, border
Methode play() aufrufen	

Eigenschaften:

.function Nummer der Komposition

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück
	nach apply sind die Filtereinstellungen zu kodieren
	für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen
	benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen



4.3.2.2.4.2.2.5.15. DropShadow Filter

Anzeige eines Konturenrahmens mit verschiedener Schattenrichtung
ab IE 5.5

Syntax:

```
HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.DropShadow(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.DropShadow(Kette)"
```

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT>
function Negieren()
{
    var Status = ID_Div.filters.item('DXImageTransform.Microsoft.dropshadow').enabled;
    ID_Div.filters.item('DXImageTransform.Microsoft.dropshadow').enabled = !Status;
}
</SCRIPT>
<DIV ID="ID_Div"
STYLE="position:absolute; top:50; left:10; width:240;
height:160; padding:10px; font:bold 13pt verdana;
filter:progid:DXImageTransform.Microsoft.dropshadow(OffX=5, OffY=5,
Color='gray', Positive='true')
onclick="Negieren();"
>
Test-Text
</DIV>
```

Eigenschaften:

.color	Farbe
.enabled	Filter-Anwendbarkeit
.offX	horizontaler Abstand des Schatten vom Objekt UND Schattenrichtung
.offY	vertikaler Abstand des Schatten vom Objekt UND Schattenrichtung
.positive	Schattentransparenz

Methoden:

keine

4.3.2.2.4.2.2.5.16. Emboss Filter

emossed Texttur mit Grauskala

ab IE 5.5

Syntax:

```
HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Emboss(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Emboss(Kette)"
```

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT>
function Aendern(ButtonZeiger)
{
    if (ID_Div.filters(0).enabled)
    {
        ID_Div.filters(0).enabled='false';
        ButtonZeiger.innerText='embossed';
    }
    else
    {
        ID_Div.filters(0).enabled='true';
        ButtonZeiger.innerText='normal';}
}
</SCRIPT>
<DIV ID="ID_Div" STYLE="position:absolute; left:270px; filter:
progid:DXImageTransform.Microsoft.emboss(enabled='false')"
>
<IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="Aendern (this)"> Aendern </BUTTON>
```

Eigenschaften:

.Bias	Prozentsatz der Farbwerterhöhung je höher der Bildkontrast um so geringer der sichtbare Effekt
.enabled	Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.17. Engrave Filter

engraved Texttur mit Grauskala



ab IE 5.5

Syntax:

```
HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Engrave(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Engrave(Kette)"
```

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT>
function Aendern (ButtonZeiger)
{
    if (ID_Div.filters(0).enabled)
    {
        ID_Div.filters(0).enabled='false';
        ButtonZeiger.innerText='Make Engraved';
    }
    else
    {
        ID_Div.filters(0).enabled='true';
        ButtonZeiger.innerText='Make Normal';
    }
}
</SCRIPT>
<DIV ID="ID_Div" STYLE="position:absolute; left:270px; filter:
progid:DXImageTransform.Microsoft.engrave(enabled='false')"
>
    <IMG SRC="/test.gif" >
</DIV>
<BUTTON onclick="Aendern(this)"> Aendern </BUTTON>
```

Eigenschaften:

.Bias Prozensatz der Farbwerterhöhung
je höher der Bildkontrast um so geringer der sichtbare Effekt

.enabled Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.5.18. Fade Filter

Fadeout und Fadeln mit überlappen

ab IE 5.5

Syntax:

```
HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Fade(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Fade(Kette)"
```

Kette String mit Eigenschaftenliste

Beispiel für Fade eines DIV:

```
<SCRIPT>
var Flag = 0;

function Aendern()
{
    ID_Div.filters[0].Apply();

    if (Flag) // 1 entspricht true
    {
        Flag = 0;
        ID_Div.style.backgroundColor="gold";
    }
    else // 0 entspricht false
    {
        Flag = 1;
        ID_Div.style.backgroundColor="blue";
    }

    ID_Div.filters[0].Play();
}
</SCRIPT>
<BUTTON onclick="Aendern ()">Aendern </BUTTON><BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
filter:progid:DXImageTransform.Microsoft.Fade(duration=2);"
>
```



</DIV>

Beispiel für Aus- und Einblende eines Bildes:

```

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

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

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

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

```

Beispiel für Aus- und Einblende von Bildern mit Bildwechsel:

Variante 1

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

```

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

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

```



```

        if (DIV_ID1_Sichtbar)
        {
            DIV_ID1_Sichtbar = false;
            DIV_ID1.style.visibility="hidden";
            DIV_ID2.style.visibility="visible";
        }
        else
        {
            DIV_ID1_Sichtbar = true;
            DIV_ID1.style.visibility="visible";
            DIV_ID2.style.visibility="hidden";
        }
        // Fade starten
        DIV_ID1.filters[0].Play();
        DIV_ID2.filters[0].Play();
    }
// -->
</SCRIPT>
</HEAD>
<BODY>
    <BUTTON onclick="BildWechsel()">
        Bild wechseln mit Aus- und Einblenden
    </BUTTON>
    <BR>
    <BR>
    <DIV ID="DIV_ID1"
        STYLE="height:250px;position:absolute;top:50;width:250px;
            background-color:red; visibility:visible
            filter:progid:DXImageTransform.Microsoft.Fade(duration=1);"
        // alles EINE Zeile
    >
        <IMG SRC="DIV_ID1_Bild.gif" ....HEIGHT=250 WIDTH=250>
        // Bild-Dimension muss in die DIV-Dimension reinpassen !
    </DIV>

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

```

Variante 2

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

```

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

    var BildUrlFeld_Laenge= BildUrlFeld.length;

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

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

```



```

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

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

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

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

    // nächstes Bild einstellen
    Index++;

    // Index korrigieren
    if (Index >= BildUrlFeld_Laenge)
    {Index=0;}
}

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

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

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

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

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

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

// Button zum Bildwechsel anzeigen
document.write(    '<BUTTON onclick="BildWechsel()">'
+      'Nächstes Bild'
+      '</BUTTON>\n'
);

// -->
</SCRIPT>
</HEAD>
<BODY>
</BODY>
</HTML>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.overlap	Anteil der Überlappungs-Dauer am Wert laut Eigenschaft .duration
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation

Methoden:

.apply() Initialisiert den Filter und setzt Anzeige zurück
nach apply sind die Filtereinstellungen zu kodieren
für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)

.play() Animiert den Filter in seinen aktuellen Einstellungen
benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen

.stop() Stop der Filteranimation vor dem normalen Ende
erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.19. FlipH Filter

horizontale Objekt-Spiegelung

ab IE 4.x

Syntax:

```
HTML <ELEMENT STYLE="filter:FlipH" ... >
Scripting object.style.filter ="FlipH"
```

Eigenschaften:

.enabled Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.20. FlipV Filter

vertikale Objekt-Spiegelung

ab IE 4.x

Syntax:

```
HTML <ELEMENT STYLE="filter:FlipV" ... >
Scripting object.style.filter ="FlipV"
```

Eigenschaften:

.enabled Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.21. Glow Filter

Glühen an den Objekträndern

ab IE 5.5

Syntax:

```
HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Glow(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Glow(Kette)"
```

Kette String mit Eigenschaftenliste

Hinweise: Objekt mit Text ohne Hintergrundfarbe --> Textzeichen glühen
Objekt mit Hintergrundfarbe oder Image --> Objektränder glühen
Objekt mit Kind, das aber außerhalb des Elternobjektes liegt --> Kind glüht nicht !!

Beispiel

```
<STYLE>
DIV.aFilter {filter: glow(Color=blue,Strength=5); width: 150;}
</STYLE>
<DIV CLASS="aFilter">gluehender Text</DIV>
```

Eigenschaften:

.color Farbe
.enabled Filter-Anwendbarkeit
.strength Ausdehnung des Glühens

Methoden:

keine

4.3.2.2.4.2.2.5.22. Gradient Filter

Farbe zwischen Hintergrund und Objektfarbe

ab IE 5.5

Syntax:

```
HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Gradient(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Gradient(Kette)"
```

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT>

function Aendern(ButtonZeiger)
{
    if (ID_Div.filters(0).enabled)
    {
        ID_Div.filters(0).enabled='false';
        ButtonZeiger.innerText='gradient';
    }
}
```



```

    }
    else
    {
        ID_Div.filters(0).enabled='true';
        ButtonZeiger.innerText='normal';
    }
}
</SCRIPT>
<DIV ID="ID_Div" STYLE="height:120px; color:green; filter:
    progid:DXImageTransform.Microsoft.gradient(enabled='false',
    startColorstr=#550000FF, endColorstr=#55FFF00)"
>
    Test-Text
</DIV>
<BUTTON onclick="Aendern(this)"> Aendern </BUTTON>

```

Eigenschaften:

- .enabled Filter-Anwendbarkeit
- .EndColor Ende-Deckungskraft als Integer
- .EndColorStr Ende-Deckungskraft als String
- .GradientType Orientierung des Gradienten
- .StartColor Start-Deckungskraft als Integer
- .StartColorStr Start-Deckungskraft als String

Methoden:

keine

4.3.2.2.4.2.2.5.23. GradientWipe Filter

Gradientstreifen

ab IE 5.5

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.GradientWipe(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.GradientWipe(Kette)"

```

Beispiel Kette String mit Eigenschaftenliste

```

<SCRIPT>
    var Flag = 0;

    function Aendern()
    {
        ID_Div.filters[0].Apply();

        if (Flag)
        {
            Flag = 0;
            ID_Div.style.backgroundColor="orange";
        }
        else
        {
            Flag = 1;
            ID_Div.style.backgroundColor="blue";
        }

        ID_Div.filters[0].Play();
    }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON><BR><BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: orange;
    filter:progid:DXImageTransform.Microsoft.gradientWipe(
    duration=3, gradientsize=0.5);"
>
</DIV>

```

Eigenschaften:

- .duration Dauer der Animation des Filters
- .enabled Filter-Anwendbarkeit
- .gradientSize Faktor der Objektbreite als Gradientbreite
- .motion Art der Bewegung
 - nur Filter GradientWipe
- .Percent Prozentualer Umfang der Anzeige mit dem der Filter enden soll
- .status Status der Filteranimation
- .WipeStyle Richtung

Methoden:

.apply() Initialisiert den Filter und setzt Anzeige zurück



nach apply sind die Filtereinstellungen zu kodieren
für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play() Animiert den Filter in seinen aktuellen Einstellungen
benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop() Stop der Filteranimation vor dem normalen Ende
erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.24. Gray Filter

Eingrauen per Graustufenskala

ab IE 4.x

Syntax:

HTML <ELEMENT STYLE="filter:Gray" ... >
Scripting object.style.filter ="Gray"

Eigenschaften:

.enabled Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.25. ICMFilter Filter

Farbkonvertierung auf Basis von Image Color Management Profilen (*.ICM-Dateien) von Hardware wie z.B. Printer etc..

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.ICMFilter(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.ICMFilter(Kette)"

Kette String mit Eigenschaftenliste

Eigenschaften:

.colorSpace Pfad der *.icm-Datei

.intent Farbverwendungsart

Methoden:

keine

4.3.2.2.4.2.2.5.26. Inset Filter

Diagonalfilter

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Inset(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Inset(Kette)"

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT>
var Flag = 0;

function Aendern()
{
    ID_Div.filters[0].Apply();

    if (Flag)
    {
        Flag = 0;
        ID_Div.style.backgroundColor="gold";
    }
    else
    {
        Flag = 1;
        ID_Div.style.backgroundColor="blue";
    }
    ID_Div.filters[0].Play(duration=2);
}
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
filter:progid:DXImageTransform.Microsoft.Inset( );"
>
</DIV>
```

Eigenschaften:

.duration Dauer der Animation des Filters

.enabled Filter-Anwendbarkeit



.Percent Prozentualer Umfang der Anzeige mit dem der Filter enden soll
 .status Status der Filteranimation

Methoden:

.apply() Initialisiert den Filter und setzt Anzeige zurück
 nach apply sind die Filtereinstellungen zu kodieren
 für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
 .play() Animiert den Filter in seinen aktuellen Einstellungen
 benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
 .stop() Stop der Filteranimation vor dem normalen Ende
 erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.27. Invert Filter

Invertieren (Negativ)

ab IE 4.x

Syntax:

HTML <ELEMENT STYLE="filter:Invert" ... >
 Scripting object.style.filter = "Invert"

Eigenschaften:

.enabled Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.28. Iris Filter

Kamera-Optik-Filter (wie die Pupille)

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Iris(Kette)" ... >
 Scripting object.style.filter = "progid:DXImageTransform.Microsoft.Iris(Kette)"

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT>
var FeldDerIrisStyles = new Array();
FeldDerIrisStyles = ['DIAMOND','CIRCLE','CROSS','PLUS','SQUARE','STAR'];

var Index = 0;
var Flag = 0;

function Aendern()
{
    var AktuellerIndex = Index % 6 ; // MOD-Funktion

    ID_Div.filters[0].irisStyle = FeldDerIrisStyles[AktuellerIndex];
    ID_Span.innerText = 'IrisStyle = ' + FeldDerIrisStyles [AktuellerIndex] + '";

    ID_Div.filters[0].Apply();

    if (Flag)
    {
        Flag = 0;
        ID_Div.style.backgroundColor="gold";
    }
    else
    {
        Flag = 1;
        ID_Div.style.backgroundColor="green";
    }

    ID_Div.filters[0].Play();

    Index += 1;
}
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
filter:progid:DXImageTransform.Microsoft.Iris(duration=3);"
>
    Test-Text
<BR>
    Test-Text
<BR>
```



```

Test-Text
<BR>
</DIV>
<SPAN ID="ID_Span"></SPAN>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.irisStyle	Schärfe des Filters
.motion	Art der Bewegung
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.29. *Light Filter*

Lichtschein (Lichtquelle positionieren)

ab IE 5.5

Achtung: pro Webseite maximal 10 Lichtfilter als STYLE-Attribut kodierbar !!!
wenn mehr dann NUR Filterklasse definieren

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Light(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Light(Kette)"

```

Kette String mit Eigenschaftenliste

Beispiel

```

<STYLE>
.aFilter {background-color: #FFFFFF; filter:light();color: cyan; width: 150;}
</STYLE>
<SCRIPT>
function Init()
{
var iX2= ID_Div.offsetWidth;
var iY2= ID_Div.offsetHeight;
ID_Div.filters[0].addCone(0,0,1,iX2,iY2,255,0,0,20,180);
}
window.onload=Init;
</SCRIPT>
<DIV CLASS="aFilter" ID="ID_Div">
Test-Text
</DIV>

```

Eigenschaften:

.enabled	Filter-Anwendbarkeit
----------	----------------------

Methoden:

.addAmbient()	indirektes Umgebungslicht (Ambiente)
.addCone()	direktes 3D-Licht als Lichttunnel oder Spot
.addPoint()	3D-Lichtpunkt als Lichtquelle, die in alle Richtungen strahlt
.changeColor()	Lichtfarbe nachträglich ändern
.changeStrength()	Lichtintensität nachträglich ändern
.clear()	alle Lichtquellen löschen
.moveLight()	Focus des Lichtkegels bzw. Lichtpunkt bewegen

4.3.2.2.4.2.2.5.30. *MaskFilter Filter*

transparente Pixelfarben maskieren so dass Umwandlung erfolgt von transparent zu nicht transparent UND nicht-transparente Pixelfarben zu transparent

ab IE 5.5

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.MaskFilter(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.MaskFilter(Kette)"

```

Kette String mit Eigenschaftenliste

Beispiel

```

<STYLE>
DIV.aFilter {filter:mask(color=#008800); width: 100;}
</STYLE>
<DIV CLASS="aFilter">

```



Test-Text
</DIV>

Eigenschaften:

.color Farbe
.enabled Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.31. Matrix Filter

Filter zur rechenintensiven Drehung, Größenänderung, Rotation
nutzt Interpolation und lineare Transformation anhand einer Matrix mit Dimension 2 mal 2

M11	M12
M21	M22

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Matrix(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Matrix(Kette)"

Beispiel Kette String mit Eigenschaftenliste
Drehen und Ausdehnen (AutoExpand)

```

<SCRIPT>
    var Wert = Math.PI * (2 / 360 ); // Umrechnung für Winkelfunktionen

    function Rotation(DivZeiger, Faktor)
    {
        var Parameter = Faktor * Wert ;
        var CosinusWert = Math.cos(Parameter);
        var SinuWert = Math.sin(Parameter);

        // Matrix belegen
        DivZeiger.filters.item(0).M11 = CosinusWert;
        DivZeiger.filters.item(0).M12 = -1 * SinuWert;
        DivZeiger.filters.item(0).M21 = SinuWert;
        DivZeiger.filters.item(0).M22 = CosinusWert;
    }

    function Ausdehnen(DivZeiger, Faktor)
    {
        // Matrix belegen
        DivZeiger.filters.item(0).M11 *= Faktor;
        DivZeiger.filters.item(0).M12 *= Faktor;
        DivZeiger.filters.item(0).M21 *= Faktor;
        DivZeiger.filters.item(0).M22 *= Faktor;
    }

    var Zahler = 400;

    function Start(DivZeiger)
    {
        var AusdehnFaktor = Zahler / 720;
        Zahler += 4;

        // wenn 3 volle Rotationen, so kein Filterereignis mehr verarbeiten
        if (Zahler >= 360*3)
        { DivZeiger.onfilterchange = null; }

        Rotation(DivZeiger, Zahler);
        Ausdehnen(DivZeiger, AusdehnFaktor);
    }
</SCRIPT>

<DIV STYLE="position:absolute;
    filter:progid:DXImageTransform.Microsoft.Matrix(sizingMethod='auto expand')"
    onfilterchange="Start(this)" >
    <DIV STYLE=" background-color: lightblue; padding:5;">
        Test-Text
    <BR>
        Test-Text
    <BR>

```



```

        Test-Text
        <BR>
        Test-Text
    </DIV>
</DIV>

```

Eigenschaften:

.Dx	X-Komponente des Vektor der linearen Transformation wird ignoriert wenn Eigenschaft . SizingMethod mit Wert "auto expand"
.Dy	Y-Komponente des Vektor der linearen Transformation wird ignoriert wenn Eigenschaft . SizingMethod mit Wert "auto expand"
.enabled	Filter-Anwendbarkeit
.FilterType	Filtertyp
.M11	Matrixfeld M11 (siehe oben)
.M12	Matrixfeld M12 (siehe oben)
.M21	Matrixfeld M21 (siehe oben)
.M22	Matrixfeld M22 (siehe oben)
.SizingMethod	Container-Resize ein/aus

Methoden:

keine

4.3.2.2.4.2.2.5.32. *MotionBlur Filter*

Bewegung im Objekt-Content

ab IE 5.5

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.MotionBlur(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.MotionBlur(Kette)"

```

Kette String mit Eigenschaftenliste

Beispiel

```

<STYLE>
    DIV.aFilter {filter:progid:DXImageTransform.Microsoft.MotionBlur(Strength=5,Direction=90);}
</STYLE>
<DIV CLASS="aFilter" STYLE="width:200">
    Test-Text
</DIV>

```

Eigenschaften:

.add	Überlagerung des Filters über das Originalbild
.direction	Filterrichtung
.enabled	Filter-Anwendbarkeit
.strength	Ausdehnung des Glühens

Methoden:

keine

4.3.2.2.4.2.2.5.33. *Pixelate Filter*

farbiges Quadrat mit Pixeltransition und Deckkraftänderung

ab IE 5.5

benötigt Eigenschaft .enabled auf false

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Pixelate(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Pixelate(Kette)"

```

Kette String mit Eigenschaftenliste

Beispiel

```

<SCRIPT>
    var Flag = 0;

    function Aendern()
    {
        ID_Div.filters[0].Apply();

        if (Flag) // 1 entspricht true
        {
            Flag = 0;
            ID_Div.style.visibility="visible";
        }
        else
        {
            Flag = 1;
            ID_Div.style.visibility="hidden";
        }
    }

```



```

    }
    ID_Div.filters[0].Play();
  }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="width:100px; background-color: lightblue;
    filter:progid:DXImageTransform.Microsoft.Pixelate(duration=3, enabled='false');"
>
    Test-Text<BR>
    Test-Text<BR>
    Test-Text
</DIV>

```

Eigenschaften:

- .duration Dauer der Animation des Filters
- .enabled Filter-Anwendbarkeit
- .maxSquare Maximale Anzahl der Pixels im Quadrat
- .Percent Prozentualer Umfang der Anzeige mit dem der Filter enden soll
- .status Status der Filteranimation

Methoden:

- .apply() Initialisiert den Filter und setzt Anzeige zurück
nach apply sind die Filtereinstellungen zu kodieren
für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
- .play() Animiert den Filter in seinen aktuellen Einstellungen
benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
- .stop() Stop der Filteranimation vor dem normalen Ende
erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.34. RadialWipe Filter

Scheibenwischer-Filter

ab IE 5.5

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.RadialWipe(Kette)" ... >
Scripting object.style.filter = "progid:DXImageTransform.Microsoft.RadialWipe(Kette)"

```

Beispiel

Kette String mit Eigenschaftenliste

```

<SCRIPT>
var FeldDerWipeStyles = new Array();
FeldDerWipeStyles = ['CLOCK', 'WEDGE', 'RADIAL'];

var Zahler = 0;
var Flag = 0;

function Aendern()
{
    var Index = Zahler % 3 ; // MOD Function

    ID_Div.filters[0].wipeStyle = FeldDerWipeStyles [Index];
    ID_Span.innerText = '.wipeStyle = "' + FeldDerWipeStyles [Index] + "'";

    ID_Div.filters[0].Apply();

    if (Flag) // 1 entspricht true
    {
        Flag = 0;
        ID_Div.style.backgroundColor="gold";
    }
    else
    {
        Flag = 1;
        ID_Div.style.backgroundColor="green";
    }

    ID_Div.filters[0].Play();
    Zahler += 1;
}
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;

```



```

filter:progid:DXImageTransform.Microsoft.RadialWipe(duration=2);"
>
    Test-Text
    <BR>
    Test-Text
    <BR>
    Test-Text
    <BR>
    Test-Text
</DIV>
<SPAN ID="ID_Span"></SPAN>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation
.wipeStyle	Style des Wischens

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.35. *RandomBars Filter*

Zufallslinien

ab IE 5.5

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.RandomBars(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.RandomBars(Kette)"

```

Kette String mit Eigenschaftenliste

Beispiel

```

<SCRIPT>
var Flag = 0;

function Aendern ()
{
    ID_Div.filters[0].orientation="vertical";
    ID_Div.filters[0].Apply();

    if (Flag) // 1 entspricht true
    {
        Flag = 0;
        ID_Div.style.backgroundColor="gold";
    }
    else
    {
        Flag = 1;
        ID_Div.style.backgroundColor="blue";
    }

    ID_Div.filters[0].Play();
}
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
filter:progid:DXImageTransform.Microsoft.RandomBars(duration=5);"
>
</DIV>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.orientation	Orientierung
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück
----------	---------------------------------------------------



nach apply sind die Filtereinstellungen zu kodieren
 für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
 .play() Animiert den Filter in seinen aktuellen Einstellungen
 benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
 .stop() Stop der Filteranimation vor dem normalen Ende
 erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.36. *RandomDissolve Filter*

Zufallspixel
ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.RandomDissolve(Kette)" ... >
 Scripting object.style.filter ="progid:DXImageTransform.Microsoft.RandomDissolve(Kette)"

Kette String mit Eigenschaftenliste
 Beispiel

```
<SCRIPT>
var Flag = 0;

function Aendern()
{
    ID_Div.filters[0].Apply();

    if (Flag)
    {
        Flag = 0;
        ID_Div.style.visibility="visible";
    }
    else
    {
        Flag = 1;
        ID_Div.style.visibility="hidden";
    }

    ID_Div.filters[0].Play();
}
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
filter:progid:DXImageTransform.Microsoft.RandomDissolve(duration=3);"
>
    Test-Text
<BR>
    Test-Text
<BR>
    Test-Text
</DIV>
```

Eigenschaften:

.duration Dauer der Animation des Filters
 .enabled Filter-Anwendbarkeit
 .Percent Prozentualer Umfang der Anzeige mit dem der Filter enden soll
 .status Status der Filteranimation

Methoden:

.apply() Initialisiert den Filter und setzt Anzeige zurück
 nach apply sind die Filtereinstellungen zu kodieren
 für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
 .play() Animiert den Filter in seinen aktuellen Einstellungen
 benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
 .stop() Stop der Filteranimation vor dem normalen Ende
 erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.37. *RevealTrans Filter*

Sammlung von 24 Überblend-Filtereffekten
sichtbar wenn System-Farbpalette (System-Farbtiefe) mindestens 256 Farben
ab IE 4.x

Syntax:

HTML <ELEMENT STYLE="filter:RevealTrans(Kette)" ... >
 Scripting object.style.filter ="RevealTrans(Kette)"
 Kette String mit Eigenschaftenliste

Beispiel



```

<SCRIPT>
    function go()
    {
        ID_Span.filters[0].Apply();
        if (ID_Span.style.visibility == "visible")
        {
            ID_Span.style.visibility = "hidden";
            ID_Span.filters.revealTrans.transition=2;
        }
        else
        {
            ID_Span.style.visibility = "visible";
            ID_Span.filters[0].transition=3;
        }
        ID_Span.filters[0].Play();
    }
</SCRIPT>
<INPUT TYPE="BUTTON" VALUE="Play Transistion" onclick="go();">
<SPAN ID="ID_Span" STYLE="position:absolute; Visibility:visible;
    Filter:revealTrans(duration=2, transition=3);
    width:300; height:300; background-color: lightgreen;"
>
    <CENTER style="background-color=red; height=100; width:100; position:relative; top:100">
    </CENTER>
</SPAN>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation
.transition	Filternummer

numerischer Wert der Eigenschaft transition**alternative Kodierung ab IE 5.5**

0 - Box von außen nach innen	DXImageTransform.Microsoft.Iris(irisstyle='SQUARE', motion='in')
1 - Box von innen nach außen	DXImageTransform.Microsoft.Iris(irisstyle='SQUARE', motion='out')
2 - Circle von außen nach innen	DXImageTransform.Microsoft.Iris(irisstyle='CIRCLE', motion='in')
3 - Circle von innen nach außen	DXImageTransform.Microsoft.Iris(irisstyle='CIRCLE', motion='out')
4 - Wischen nach oben	DXImageTransform.Microsoft.Blinds(direction='up', bands=1)
5 - Wischen nach unten	DXImageTransform.Microsoft.Blinds(direction='down', bands=1)
6 - Wischen nach rechts	DXImageTransform.Microsoft.Blinds(direction='right', bands=1)
7 - Wischen nach links	DXImageTransform.Microsoft.Blinds(direction='left', bands=1)
8 - vertikale Jalousie	DXImageTransform.Microsoft.Blinds(direction='right')
9 - horizontale Jalousie	DXImageTransform.Microsoft.Blinds(direction='down')
10 - Schachbrett von links oben nach rechts unten	DXImageTransform.Microsoft.CheckerBoard(direction='right')
11 - Schachbrett von oben nach unten	DXImageTransform.Microsoft.CheckerBoard(direction='down')
12 - zufällige Auflösung	DXImageTransform.Microsoft.RandomDissolve
13 - Split vertikal von außen nach innen	DXImageTransform.Microsoft.Barn(orientation='vertical', motion='in')
14 - Split vertikal von innen nach außen	DXImageTransform.Microsoft.Barn(orientation='vertical', motion='out')
15 - Split horizontal von außen nach innen	DXImageTransform.Microsoft.Barn(orientation='horizontal', motion='in')
16 - Split horizontal von innen nach außen	DXImageTransform.Microsoft.Barn(orientation='horizontal', motion='out')
17 - Streifen von rechts oben nach links unten	DXImageTransform.Microsoft.Strips(motion='leftdown')
18 - Streifen von rechts unten nach links oben	DXImageTransform.Microsoft.Strips(motion='leftup')
19 - Streifen von links oben nach rechts unten	DXImageTransform.Microsoft.Strips(motion='rightdown')
20 - Streifen von links unten nach rechts oben	DXImageTransform.Microsoft.Strips(motion='rightup')
21 - Streifen horizontal zufällig	DXImageTransform.Microsoft.RandomBars(orientation='horizontal')
22 - Streifen vertikal zufällig	DXImageTransform.Microsoft.RandomBars(orientation='vertical')
23 - Zufallsauswahl aus einem der Effekte 0 bis 22	

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.38. Shadow Filter

Konturschatten

ab IE 5.5

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Shadow(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Shadow(Kette)"

```



Kette String mit Eigenschaftenliste

Beispiel für Schatten eines DIV:

```
<STYLE>
  DIV.aFilter {filter:shadow(color=#0000FF,direction=45); width: 150; color: FF0000;}
</STYLE>
<DIV CLASS="aFilter">
  Test-Text
</DIV>
```

Beispiel für Schatten eines Bildes und eines Textes:

```
<HTML>
<HEAD>
<STYLE>
  DIV.filter_name {filter:shadow(color=#000000,direction=120); width:100; color: #FF0000;}
</STYLE>
</HEAD>
<BODY>
  <DIV CLASS="filter_name" STYLE="width:400;">
    <IMG SRC="bild.gif">
    Das Bild und dieser Text haben einen Schatten
  <BR>
  Die Bild-Dimension muss mit dem des DIV uebereinstimmen !
  </DIV>
</BODY>
</HTML>
```

Beispiel für 3D-Effekt eines Bildes:

Dieser Effekt dient dem verschobenen Überlagern ein und desselben Bilder, so dass ein 3D-Effekt mit Schatten erzeugt wird. Die Verschiebung ist damit die Schattendicke bzw. die Effekt-Stärke des 3D-Effektes. Besonders günstig sieht man diesen Effekt, wenn die Grafik einen Schriftzug enthält.
Die Überlagerung erfolgt anhand zweier DIV's, die je eine CSS-Klasse erhalten und ansonsten identisch sind.

```
<HTML>
<HEAD>
<STYLE>
  DIV.DIV1_filter: {filter:shadow(color=#FFFF00,direction=300, strength=5); width:100; color: #FF0000;}
  DIV.DIV2_filter: {filter:shadow(color=#000000,direction=120, strength=8); width:100; color: #FF0000;}
</STYLE>
</HEAD>
<BODY>
  <DIV CLASS="DIV1_filter " STYLE="width:400; position:absolute; top:50px; left:50px;">
    <BLOCKQUOTE>
      <IMG SRC="bild.gif">
      Das Bild und dieser Text haben einen Schatten
    <BR>
      Die Bild-Dimension muss mit dem des DIV uebereinstimmen !
    </BLOCKQUOTE>
  </DIV>
  <DIV CLASS="DIV2_filter " STYLE="width:400; position:absolute; top:50px; left:50px;">
    <BLOCKQUOTE>
      <IMG SRC="bild.gif">
      Das Bild und dieser Text haben einen Schatten
    <BR>
      Die Bild-Dimension muss mit dem des DIV uebereinstimmen !
    </BLOCKQUOTE>
  </DIV>
</BODY>
</HTML>
```

Eigenschaften:

.color	Farbe
.direction	Filterrichtung
.enabled	Filter-Anwendbarkeit
.strength	Ausdehnung des Glühens

Methoden:

keine

4.3.2.2.4.2.2.5.39. Slide Filter

ab IE 5.5



Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Slide(Kette)" ... >
 Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Slide(Kette)"

Kette String mit Eigenschaftenliste

Beispiel

```
<SCRIPT>
var FeldDerSlideTyles = new Array();
FeldDerSlideTyles = ['HIDE', 'PUSH', 'SWAP'];

var Zahler = 0;
var Flag = 0;

function Aendern()
{
    var Index = Zahler % 3 ; // MOD-Funktion
    ID_Div.filters[0].slideStyle = FeldDerSlideTyles[Index];
    ID_Span.innerText = '.slideStyle = "' + FeldDerSlideTyles[Index] + '"';

    ID_Div.filters[0].Apply();

    if (Flag)
    {
        Flag = 0;
        ID_Div.style.backgroundColor="gold";
    }
    else
    {
        Flag = 1;
        ID_Div.style.backgroundColor="green";
    }

    ID_Div.filters[0].Play();

    Zahler += 1;
}
</SCRIPT>
<BUTTON onclick="Aendern ()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
filter:progid:DXImageTransform.Microsoft.Slide(duration=3, bands=8);"
>
    Test-Text
<BR>
    Test-Text
<BR>
    Test-Text
<BR>
    Test-Text
<BR>
    Test-Text
</DIV>
<SPAN ID="ID_Span" oSpan"></SPAN>
```

Eigenschaften:

- .bands Anzahl der Streifen
- .duration Dauer der Animation des Filters
- .enabled Filter-Anwendbarkeit
- .Percent Prozentualer Umfang der Anzeige mit dem der Filter enden soll
- .slideStyle Art
- .status Status der Filteranimation

Methoden:

- .apply() Initialisiert den Filter und setzt Anzeige zurück
nach apply sind die Filtereinstellungen zu kodieren
für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
- .play() Animiert den Filter in seinen aktuellen Einstellungen
benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
- .stop() Stop der Filteranimation vor dem normalen Ende
erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.40. Spiral Filter

ab IE 5.5

Syntax:

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Spiral(Kette)" ... >
 Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Spiral(Kette)"



```

Beispiel   Kette   String mit Eigenschaftenliste
<SCRIPT>
    var Flag = 0;
    function Aendern()
    {
        ID_Div.filters[0].Apply();
        if (Flag)
        {
            Flag = 0;
            ID_Div.style.visibility="visible";
        }
        else
        {
            Flag = 1;
            ID_Div.style.visibility="hidden";
        }
        ID_Div.filters[0].Play();
    }
</SCRIPT>
<BUTTON onclick="Aendern ()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.Spiral(
    duration=3, GridSizeX=25, GridSizeY=25);"
>
    Test-Text
    <BR>
    Test-Text
    <BR>
    Test-Text
    <BR>
    Test-Text
    </DIV>

```

Eigenschaften:

- .duration Dauer der Animation des Filters
- .enabled Filter-Anwendbarkeit
- .gridSizeX Anzahl Spalten des Gitters
- .gridSizeY Anzahl Zeilen des Gitters
- .Percent Prozentualer Umfang der Anzeige mit dem der Filter enden soll
- .status Status der Filteranimation

Methoden:

- .apply() Initialisiert den Filter und setzt Anzeige zurück
nach apply sind die Filtereinstellungen zu kodieren
für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
- .play() Animiert den Filter in seinen aktuellen Einstellungen
benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
- .stop() Stop der Filteranimation vor dem normalen Ende
erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.41. *Stretch Filter*

ab IE 5.5

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Stretch(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Stretch(Kette)"

```

Kette String mit Eigenschaftenliste

```

Beispiel
<SCRIPT>
    var FeldDerStretchStyles = new Array();
    FeldDerStretchStyles = ['HIDE', 'PUSH', 'SPIN'];

    var Zahler = 0;
    var Flag = 0;

    function Aendern()
    {
        var Index = Zahler % 3 ; // MOD-Funktion
        ID_Div.filters[0].stretchstyle = FeldDerStretchStyles[Index];
    }

```



```

        ID_Span.innerText = '.stretchStyle = "' + FeldDerStretchStyles[Index] + '"';

        ID_Div.filters[0].Apply();

        if (Flag)
        {
            Flag = 0;
            ID_Div.style.backgroundColor="gold";
        }
        else
        {
            Flag = 1;
            ID_Div.style.backgroundColor="green";
        }

        ID_Div.filters[0].Play();

        Zahler += 1;
    }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
        filter:progid:DXImageTransform.Microsoft.Stretch(duration=3);"
>
    Test-Text
    <BR>
    Test-Text
    <BR>
    Test-Text
    <BR>
    Test-Text
</DIV>
<SPAN ID="ID_Span"></SPAN>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation
.stretchStyle	Art

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.42. Strips Filter

Diagonalstreifen

ab IE 5.5

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Strips(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Strips(Kette)"

```

Beispiel Kette String mit Eigenschaftenliste

```

<SCRIPT>
    var Flag = 0;

    function Aendern()
    {
        ID_Div.filters[0].Apply();

        if (Flag)
        {
            Flag = 0;
            ID_Div.style.backgroundColor="gold";
        }
        else
        {

```



```

        Flag = 1;
        ID_Div.style.backgroundColor="blue";
    }

    ID_Div.filters[0].Play();
}
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.Strips(
    duration=5, motion='rightdown');"
>
</DIV>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.motion	Ecke
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation
.stretchStyle	Art

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.43. Wave Filter

vertikale Wellen
ab IE 5.5

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Wave(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Wave(Kette)"

```

Kette String mit Eigenschaftenliste

Beispiel

```

<STYLE>
    DIV.aFilter {filter:wave(strength=2, freq=3, lightstrength=20, add=0, phase=90);
    width: 150; color: #FF0000;}
</STYLE>
<DIV CLASS="aFilter">
    Test-Text
</DIV>

```

Eigenschaften:

.add	Überlagerung des Filters über das Originalbild
.enabled	Filter-Anwendbarkeit
.freq	Anzahl der Wellen
.lightStrength	prozentuale Differenz der Licht-Intensität zwischen Wellengipfel und Wellentäler
.phase	prozentuale Phasenverschiebung der Wellen, also Verschiebung der Wellenüberlagerung als Prozentanteil des Wellenzyklus
.strength	Ausdehnung des Glühens

Methoden:

keine

4.3.2.2.4.2.2.5.44. Wheel Filter

Speichenrad-Dreh-Filter
ab IE 5.5

Syntax:

```

HTML <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Wheel(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Wheel(Kette)"

```

Kette String mit Eigenschaftenliste

Beispiel

```

<SCRIPT>
    var Flag = 0;

    function Aendern()
    {

```



```

        ID_Div.filters[0].Apply();

        if (Flag)
        {
            Flag = 0;
            ID_Div.style.backgroundColor="gold";
        }
        else
        {
            Flag = 1;
            ID_Div.style.backgroundColor="blue";
        }

        ID_Div.filters[0].Play();
    }
</SCRIPT>
<BUTTON onclick="Aendern()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
        filter:progid:DXImageTransform.Microsoft.Wheel(duration=2, spokes=8);"
>
</DIV>

```

Eigenschaften:

- .duration Dauer der Animation des Filters
- .enabled Filter-Anwendbarkeit
- .Percent Prozentualer Umfang der Anzeige mit dem der Filter enden soll
- .spokes Anzahl Speichen
- .status Status der Filteranimation

Methoden:

- .apply() Initialisiert den Filter und setzt Anzeige zurück
nach apply sind die Filtereinstellungen zu kodieren
für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
- .play() Animiert den Filter in seinen aktuellen Einstellungen
benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
- .stop() Stop der Filteranimation vor dem normalen Ende
erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.45. Xray Filter

Schwarz-Weiss Darstellung
ab IE 4.x

Syntax:

```

HTML     <ELEMENT STYLE="filter:Xray" ... >
Scripting object.style.filter ="Xray"

```

Beispiel

```

<STYLE>
        DIV.aFilter {filter:xray; width: 150; color: #FF0000;}
</STYLE>
<DIV CLASS="aFilter">
        Test-Text
</DIV>

```

Eigenschaften:

- .enabled Filter-Anwendbarkeit

Methoden:

keine

4.3.2.2.4.2.2.5.46. ZigZag Filter

Zigzag-Bewegung
ab IE 5.5

Syntax:

```

HTML     <ELEMENT STYLE="filter:progid:DXImageTransform.Microsoft.Zigzag(Kette)" ... >
Scripting object.style.filter ="progid:DXImageTransform.Microsoft.Zigzag(Kette)"

```

Kette String mit Eigenschaftenliste

Beispiel

```

<SCRIPT>
        var Flag = 0;
        function Aendern()
        {
            ID_Div.filters[0].Apply();
            if (Flag)
            {

```



```

        Flag = 0;
        ID_Div.style.visibility="visible";
    }
    else
    {
        Flag = 1;
        ID_Div.style.visibility="hidden";
    }
    ID_Div.filters[0].Play();
}
</SCRIPT>
<BUTTON onclick="Aendern ()">Aendern </BUTTON>
<BR>
<DIV ID="ID_Div" STYLE="height:250px; width:250px; background-color: gold;
    filter:progid:DXImageTransform.Microsoft.Zigzag(
    duration=3, GridSizeX=25, GridSizeY=25);"
>
    Test-Text
    <BR>
    Test-Text
    <BR>
    Test-Text
    <BR>
    Test-Text
    <BR>
    Test-Text
</DIV>

```

Eigenschaften:

.duration	Dauer der Animation des Filters
.enabled	Filter-Anwendbarkeit
.gridSizeX	Anzahl Spalten des Gitters
.gridSizeY	Anzahl Zeilen des Gitters
.Percent	Prozentualer Umfang der Anzeige mit dem der Filter enden soll
.status	Status der Filteranimation

Methoden:

.apply()	Initialisiert den Filter und setzt Anzeige zurück nach apply sind die Filtereinstellungen zu kodieren für Animationstart play-Methode benutzen (sichtbarmachen der Filtereinstellungen)
.play()	Animiert den Filter in seinen aktuellen Einstellungen benötigt zuvor Methode .apply() und dem .apply() nachfolgende Filtereinstellungen
.stop()	Stop der Filteranimation vor dem normalen Ende erzeugt Event onfilterchange

4.3.2.2.4.2.2.5.47. Filter bei wichtigen Objekten (Auswahl)

a Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

area Objekt

keine Filter

applet Objekt

keine Filter

body Objekt

Alpha	CheckerBoard	FlipV
AlphaImageLoader	Chroma	Glow
Barn	Compositor	Gradient
BasicImage	Emboss	GradientWipe
BlendTrans	Engrave	Gray
Blinds	Fade	ICMFilter
Blur	FlipH	Inset



Invert	RadialWipe	Stretch
Iris	RandomBars	Strips
Light	RandomDissolve	Wave
MaskFilter	RevealTrans	Wheel
Matrix	Shadow	Xray
MotionBlur	Slide	Zigzag
Pixelate	Spiral	

button Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

currentStyle

keine

custom Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

div Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

document Objekt

keine Filter

fieldset Objekt

Alpha	Chroma	Glow
AlphaImageLoader	Compositor	Gradient
Barn	DropShadow	GradientWipe
BasicImage	Emboss	Gray
BlendTrans	Engrave	ICMFilter
Blinds	Fade	Inset
Blur	FlipH	Invert
CheckerBoard	FlipV	Iris



Light	RandomBars	Stretch
MaskFilter	RandomDissolve	Strips
Matrix	RevealTrans	Wave
MotionBlur	Shadow	Wheel
Pixelate	Slide	Xray
RadialWipe	Spiral	Zigzag

font Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave Performs
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

form Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

frame Objekt

Alpha	FlipV	RadialWipe
AlphaImageLoader	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Invert	Spiral
Chroma	Iris	Stretch
Compositor	Light	Wave
DropShadow	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag
FlipH	Pixelate	

frameset Objekt

Barn	Gradient	RandomDissolve
BasicImage	GradientWipe	Slide
Blinds	ICMFilter	Spiral
Blur	Inset	Stretch
CheckerBoard	Iris	Strips
Compositor	Matrix	Wheel
Emboss	Pixelate	Zigzag
Engrave	RadialWipe	
Fade	RandomBars	

iframe Objekt

Alpha	Blinds	DropShadow
AlphaImageLoader	Blur	Emboss
Barn	CheckerBoard	Engrave
BasicImage	Chroma	Fade
BlendTrans	Compositor	FlipH



FlipV	Light	Shadow
Glow	MaskFilter	Slide
Gradient	Matrix	Spiral
GradientWipe	MotionBlur	Stretch
Gray	Pixelate	Strips
ICMFilter	RadialWipe	Wave
Inset	RandomBars	Wheel
Invert	RandomDissolve	Xray
Iris	RevealTrans	Zigzag

img Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	ZigZag

input Objekt

keine Events

input button Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

input checkbox Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

input file Objekt

Alpha	Compositor	GradientWipe
AlphaImageLoader	DropShadow	Gray
Barn	Emboss	ICMFilter
BasicImage	Engrave	Inset
BlendTrans	Fade	Invert
Blinds	FlipH	Iris
Blur	FlipV	Light
CheckerBoard	Glow	MaskFilter
Chroma	Gradient	Matrix



MotionBlur	RevealTrans	Strips
Pixelate	Shadow	Wave
RadialWipe	Slide	Wheel
RandomBars	Spiral	Xray
RandomDissolve	Stretch	Zigzag

input hidden Objekt

keine Filter

input image Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

input password Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

input radio Objekt

Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

input reset Objekt

Alpha	Fade	Matrix
AlphaImageLoader	FlipH	MotionBlur
Barn	FlipV	Pixelate
BasicImage	Glow	RadialWipe
BlendTrans	Gradient	RandomBars
Blinds	GradientWipe	RandomDissolve
Blur	Gray	RevealTrans
CheckerBoard	ICMFilter	Shadow
Chroma	Inset	Slide
Compositor	Invert	Spiral
DropShadow	Iris	Stretch
Emboss	Light	Strips
Engrave	MaskFilter	Wave



Wheel	Xray	Zigzag
<u>input submit Objekt</u>		
Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

<u>input text Objekt</u>		
Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

<u>marquee Objekt</u>		
Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

<u>object Objekt</u>		
Alpha	FlipH	Pixelate
AlphaImageLoader	FlipV	RadialWipe
Barn	Glow	RandomBars
BasicImage	Gradient	RandomDissolve
BlendTrans	GradientWipe	RevealTrans
Blinds	Gray	Shadow
Blur	ICMFilter	Slide
CheckerBoard	Inset	Spiral
Chroma	Invert	Stretch
Compositor	Iris	Strips
DropShadow	Light	Wave
Emboss	MaskFilter	Wheel
Engrave	Matrix	Xray
Fade	MotionBlur	Zigzag

option Objekt
keine

<u>runtimeStyle</u>		
Alpha	BlendTrans	Chroma



DropShadow
FlipH
FlipV
Glow
Gray

Invert
Light
MaskFilter
MotionBlur
Redirect

RevealTrans
Shadow
Wave
Xray

select Objekt

keine Filters

selection Objekt

keine Filters

span Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade

FlipH
FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur

Pixelate
RadialWipe
RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

style Objekt

Alpha
BlendTrans
Chroma
DropShadow
FlipH
FlipV

Glow
Gray
Invert
Light
MaskFilter
MotionBlur

RevealTrans
Shadow
Wave
Xray

table Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade

FlipH
FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur

Pixelate
RadialWipe
RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

table.caption Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade

FlipH
FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur

Pixelate
RadialWipe
RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

table.col Objekt

keine



table.colGroup Objekt

keine

table.tBody Objekt

keine

table.tFoot Objekt

keine

table.tHead Objekt

keine

table.tr Objekt

keine

table.tr.td Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade
FlipH

FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur
Pixelate
RadialWipe

RandomBars
RandomDissolve
Redirect
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

table.tr.th Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade
FlipH

FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur
Pixelate
RadialWipe

RandomBars
RandomDissolve
Redirect
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

textarea Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans
Blinds
Blur
CheckerBoard
Chroma
Compositor
DropShadow
Emboss
Engrave
Fade

FlipH
FlipV
Glow
Gradient
GradientWipe
Gray
ICMFilter
Inset
Invert
Iris
Light
MaskFilter
Matrix
MotionBlur

Pixelate
RadialWipe
RandomBars
RandomDissolve
RevealTrans
Shadow
Slide
Spiral
Stretch
Strips
Wave
Wheel
Xray
Zigzag

var Objekt

Alpha
AlphaImageLoader
Barn
BasicImage
BlendTrans

Blinds
Blur
CheckerBoard
Chroma
Compositor

DropShadow
Emboss
Engrave
Fade
FlipH



FlipV	Light	Shadow
Glow	MaskFilter	Slide
Gradient	Matrix	Spiral
GradientWipe	MotionBlur	Stretch
Gray	Pixelate	Strips
ICMFilter	RadialWipe	Wave
Inset	RandomBars	Wheel
Invert	RandomDissolve	Xray
Iris	RevealTrans	Zigzag

window Objekt

keine Filter

4.3.2.2.4.3. HTML-Elemente im HTML-Dokument des Internet Explorer (Auswahl)**Voraussetzungen für die programmtechnische Nutzung von HTML-Elementen als Objekte in JScript:**

Kenntnisse der Programmierers zu:

DOM
 Zeigerverwendung
 Vererbung
 Browserversionen/Script-Maschinen-Versionen
 JScript-Sprache
 HTML-Syntax zum HTML-Element (Attribute etc.)

Hinweise: JScript von Microsoft ist programmtechnisch nicht netzwerkfähig, es sei denn, es werden Komponenten des Betriebssystems Windows

in JScript aktiviert (falls überhaupt möglich). JScript ist nur client-orientiert und arbeitet direkt auf dem PC des Internet-Users, der eine Webseite mit JScript-Teilen besucht. Aus Sicht des Webseitenbetreibers (Hoster) und dessen Server ist der User der "Endverbraucher" des Internets, also der Klient ("Kunde") des Webseitenangebotes. Wer netzwerkfähige Produkte erzeugen will, sollte sich mit der NET-Software zu Windows XP von Microsoft beschäftigen, die JScript und XML auf Basis einer **installierten und aktiven** Server-Software netzwerkfähig machen.

Bezüglich des implementierten HTML-DOM vom Internet Explorer ist festzustellen, dass große Ähnlichkeiten bzw. teilweise Identität mit anderen DOM, z.B. von XML, vorliegen. Für den Programmierer, der nicht nur JScript nutzen will, ist es für den Einstieg in XML vorteilhaft, die Prinzipien des HTML-DOM und dessen Umsetzung in JScript verinnerlicht zu haben. Microsoft setzt ganz bewusst auf prinzipielle Gemeinsamkeiten und Integration der Script-Programmierung per JScript mit bzw. in XML. So gesehen wird XML das HTML problemlos ablösen.

JScript kann nur dann auf dem User-PC aktiv werden, wenn der User Javascript zugelassen hat. Die Zulassung von Javascript hängt von der Browser-Software und optional eingesetzter Firewall-Software ab.

JScript kann Komponenten des Betriebssystems Windows wie DirectX oder ActiveX nutzen. Auch hier gilt: Der User muss die Nutzung von DirectX und ActiveX zugelassen haben. Es gibt Webseitenanbieter, z.B. Hoster für kostenlose Homepages, deren Host-Angebote (Webseiten der Kunden des Hosters) **nur** unter Nutzung von ActiveX funktionieren, wenn die Homepage mit Komponenten des Hosters erstellt wurden. Dem Besucher solcher Webseiten ist in der Regel nicht bekannt, was die ActiveX-Komponenten bewirken. Solche Webseitenangebote sind dann aus Sicht des Schutzes der Privatsphäre des Users **wertlos**, wenn der User ActiveX gesperrt hat (weil kein Risiko eingehen will) und in der Webseite keine Alternativen zu ActiveX bzw. Hinweise zu Art und Zweck von eingesetzten ActiveX-Komponenten programmiert wurden.

Der JScript-Programmierer muss daher **regelmäßig** mit dem Einsatz von Firewall-Software auf dem User-PC rechnen und daher den Einsatz von Windowskomponenten für den User **transparent** halten, wenn der Programmierer möchte, dass **sicherheitsbewusste** User die erstellte Webseite nutzen können und der User **keine Risiko** tragen soll.

JScript ist im Gegensatz zu JavaScript von Netscape nur schwer mit JAVA verbindbar, da die Browserhersteller divergierende Interessen zu JAVA besitzen und somit Bezüge auf JAVA mehr oder weniger in die Scriptmaschine implementieren.

JScript ist wie jede JavaScript-Sprache eine rein objektorientierte Sprache, die mit Zeigern arbeitet., denn HTML-Elemente können **nur als Objekte**, also nur objektorientiert verarbeitet werden. Die Zeigernutzung erfolgt intern, aber im Regelfall auch durch den Programmierer per Script-Code, vorallem dann sehr intensiv, wenn der Programmierer einen dynamischen Inhalt der Webseite erstellen will.

Die Pflege und Dokumentation der Scriptsprache und deren Implementationen im Browser hängt vom Willen des Herstellers ab, der mehr oder weniger in der Lage sein möchte, für denjenigen Transparenz zu gewähren, der Script programmtechnisch nutzen will. Damit können Objektbeschreibungen kurz- oder langelig sein, letzteres nur dann, wenn Browser-Versionen des Herstellers kompatibel sind bzw. Unterschiede regelmäßig qualifiziert und leicht zugänglich publiziert werden.

Abbildung des HTML-Elementes als Objekt:

HTML-Elemente werden als Objekte im Rahmen des HTML-DOM erzeugt.

Attribute des HTML-Elementes laut HTML-Syntax sind im Objekt als Eigenschaften abgebildet, leider - je nach Art des HTML-Elementes - **nicht komplett (nicht alle Attribute)**. Und: Es existieren z.T. Eigenschaften, denen der jeweilige Pedant in HTML-Attribut-Form fehlt - besonders beim Internet Explorer, der damit wesentlich mehr als HTML bietet, das allerdings Microsoft-spezifisch für das Betriebssystem Windows. Diese Abbildungs-Varianten sind der Ausdruck des Grades einer HTML-Unterstützung im Browser, denn HTML-Elemente können **nur als Objekte**, also nur objektorientiert verarbeitet werden. JScript ist daher , wie jede JavaScript-Sprache, eine rein objektorientierte Sprache, die grundsätzlich mit Zeigern auf Instanzen eines Objektes arbeitet.

