

div Objekt des Internet Explorer

ab IE 4.x

Das DIV- bzw. SPAN-Objekt ähneln sich sehr stark.

DIV kann ein automatisches
 am DIV-Ende implizieren, so dass anstelle DIV das Objekt P verwendet werden kann, wenn es mit Eigenschaften versorgt ist, die auch der zu ersetzende DIV hat.

Die übliche Zugriffsweise ist über das ID-Attribut (Eigenschaft .id) des DIV bzw. SPAN.

DIV bzw. SPAN können im Elternobjekt (Dokument im Fenster oder übergeordneter DIV bzw. SPAN) erzeugt werden, als HTML-Element üblicherweise im BODY-Teil des Dokumentes per Script durch z.B. document.write("<DIV >...");

Die Anzeige (das Rendern) des DIV bzw.SPAN erfolgt erst mit dem Parsen des Ende-Tags, also </DIV> bzw. . Danach können Komponenten zur Laufzeit nachträglich verändert werden, die aber nur z.T. sofort sichtbar werden (teilweise erst nach dem Reload des betreffenden Dokumentes).

NS und IE können das DIV- und SPAN-Objekt erzeugen, implementieren und rendern diese aber verschiedenartig.

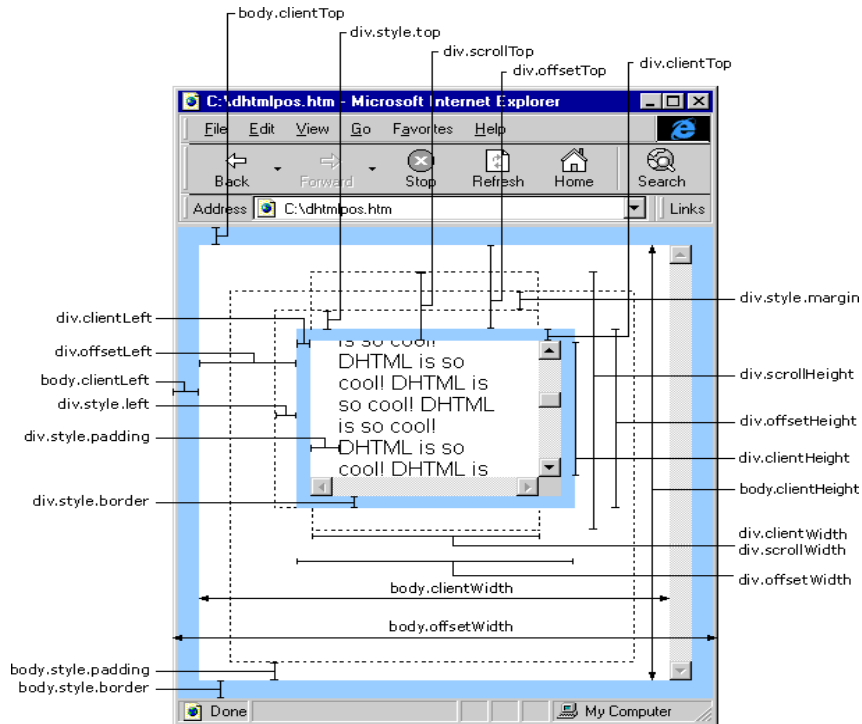
Beispiel: Nur der Internet Explorer erzeugt automatisch ein BODY-Objekt, wenn dieses im Dokument nicht kodiert wurde, weil z.B. sämtliche HTML-Elemente im HEAD des Dokumentes per Script erzeugt wurden.

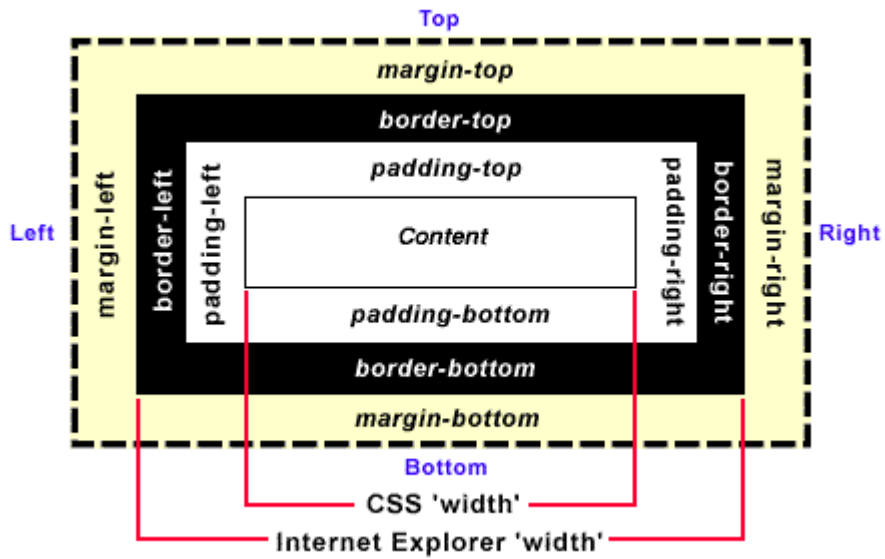
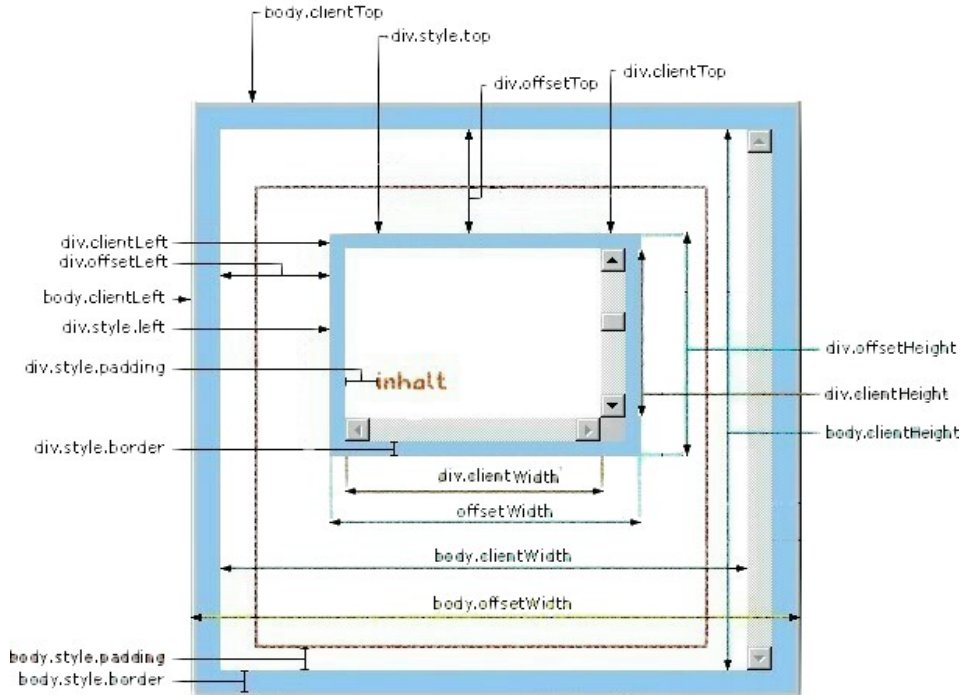
Der NS verlangt immer einen BODY-Teil und ignoriert o.g. Script-Anweisungen im HEAD des Dokumentes, solange der BODY des Dokumentes nicht komplett geparkt wurde (</BODY> noch nicht erkannt wurde). Es können also nur nachträglich neue HTML-Elemente per Script im HEAD des Dokumentes erzeugt werden. Scriptaufrufe aus dem BODY-Teil werden mit dem BODY geparkt, egal ob die Scripte im HEAD oder BODY liegen.

Im NS unter der Version 6.x wird anstelle von DIV- bzw. SPAN-Objekt das LAYER-Objekt favorisiert. Ab NS 6.x wird das Layer-Objekt radikal nicht mehr unterstützt: Es sind für die Ebenendarstellung (überlappende Objekte) keine Layer mehr sondern z.B. DIV oder SPAN verwendbar.

Bezüglich zeitsteuernder Eigenschaften/Methoden werden ab IE 6.x auch Behavior style.time2 und Objekt currTimeState benutzt.

Wichtige Eigenschaften im IE:





Hinweis Margin: Abstand des Aussenrandes eines Objektes zur Umgebung
 Padding: Abstand des Objektinhaltes zum Aussenrand

Hinweis zur Pars-Reihenfolge des Internet Explorer innerhalb der HTML-Kodierung bezüglich dem Tag-Name und den Element-Attributen CLASS, ID, STYLE

1. Element-Bezeichner
2. CLASS-Attribut mit Bezeichner aus Klassendeklaration im HEAD
3. ID-Attribut
4. STYLE-Attribut mit Style-Werten (nicht mit Bezeichner aus Style-Deklaration im HEAD)

Es gilt: Wenn gleiche Bezeichner verwendet, so nur Werte des **zuletzt** geparsten Bezeichners verwendet !
 Die CLASS-Deklaration aus dem HEAD des Dokumentes wird wertmäßig durch die Style-Deklaration per Attribut des HTML-Elementes überschrieben, wenn gleiche Style-Eigenschaften betroffen sind (ansonsten hinzufügen).

Hinweis zu dynamischen Eigenschaftenveränderung zur Laufzeit:

Die zuletzt während der Laufzeit getätigte Definition ersetzt wertmäßig den aktuellen Attributwert, wenn gleiche Attributnamen / Eigenschaften betroffen sind (sonst hinzufügen).

Beispiele:



Beispiel 1:

```

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

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

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

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

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

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

```

Beispiel 2 für Sound mit Sekundenanzeige:

```

<HTML>
<HEAD>
<SCRIPT>
    // ++++++ globale Variablen, die verändert werden können
    var SoundUrl = "56sec.mid";
    var SoundDauerInSekunden = 56; // Dauer muss exakt stimmen !

    var PixelBreiteProBalkenErweiterung = 10;

    // ++++++ Browser-Typ ermitteln
    // Dieser Quellcode muss VOR allen anderen Routinen codiert sein, damit zuerst abgearbeitet
    var ns = document.layers ? true : false;
    var ie = document.all ? true : false;

    // ++++++ Routinen der Sekundenzählung
    var SekundenZahler = 0;
    var SekundenZahlerTimeoutID = null;

    function SekundenZaehlen() // wird durch Rekursion alle Sekunde neu gestartet
    {
        // Zähler erhöhen

```



```

        SekundenZahler++;

        // visuelle Anzeige im Dokument auffrischen, also alle Audrucke der Style-Werte
        // neu berechnen und damit alle DIV's neu visualisieren
        document.recalc();
    }

function SekundenZaehlen_Start()
{
    // prüfen ob Sekunden zählen nicht bereits läuft
    if (SekundenZahlerTimeoutID == null)
    {
        // nicht aktiv

        // Rekursion starten: SekundenZaehlen() wird permanent alle Sekunde aktiviert
        SekundenZahlerTimeoutID = setInterval("SekundenZaehlen()", 1000);
    }
}

function SekundenZaehlen_Stop()
{
    // prüfen ob Sekunden zählen aktiv ist
    if (SekundenZahlerTimeoutID != null)
    {
        // aktiv, also stoppen
        clearInterval(SekundenZahlerTimeoutID);
        SekundenZahlerTimeoutID = null;
    }
}

// ++++++ Routine zur Erzeugung Sound-Objekt
function SoundObjektErzeugen(SoundFileUrl, SoundFileDauerInSekunden)
{
    this.SoundFileUrl = SoundFileUrl;
    this.SoundFileDauerInMillisekundeSekunden = SoundFileDauerInSekunden * 1000;
    // Timerzeit für Rekursion
    this.SoundFileBeendet = true; // kein Sound aktiv
}

// ++++++ Routinen zur Wiedergabe Sound-Objekt
var SoundTimeoutID=0;

function SoundAbspielen()
{
    // prüfen ob Sound nicht bereits aktiv ist
    if (SoundObjekt.SoundFileBeendet)
    {
        // Anzeige initialisieren
        SekundenAnzeigeInit();

        // Sekundenzähler starten, wobei das Zählen eigenständig und parallel erfolgt
        SekundenZaehlen_Start();

        // Sound erzeugen und sofort starten durch Url-Zuweisung
        ID_BGSound.src=SoundObjekt.SoundFileUrl ;
        SoundObjekt.SoundFileBeendet=false;

        // Millisekunden warten und danach die Funktion SoundAbspielen() neu aufrufen
        SoundTimeoutID = setTimeout(
            "SoundAbspielen()",
            SoundObjekt.SoundFileDauerInMillisekundeSekunden
        );
    }
    else
    {
        // Dieser Zweig wird erst mit dem 2. Aufruf der Funktion abgearbeitet

        // Sound zu Ende
        SoundObjekt.SoundFileBeendet=true;

        // Sekundenzähler stoppen
        SekundenZaehlen_Stop();

        // und Meldung
    }
}

```



```

        var TimerUngenauigkeit1 = SoundDauerInSekunden - SekundenZahler;
        var TimerUngenauigkeit2 = TimerUngenauigkeit1 / SoundDauerInSekunden;
        alert(      "Wiedergabe beendet\nUngenauigkeit des Timers = "
            + TimerUngenauigkeit1.toString()+ " Sekunden\n"
            + "also " + TimerUngenauigkeit2.toString() + " Ticks pro Sekunde"
        );
    }
}

// ++++++ Sekunden-Anzeige initialisieren
function SekundenAnzeigeInit()
{
    // ---- Variablen init
    SekundenZahler      = 0;
    SekundenZahlerTimeoutID = null;

    // ---- visuelle Anzeige erzeugen
    // - - - Sekundenbalken und Sekundenzähler dynamisch visualisieren
    //      Es wird jedem DIV als Style-Wert ein Ausdruck hinterlegt, also kein Wert.
    //      Der Ausdruck liefert den Wert , welcher sofort das Layout der
    //      DIV's beeinflusst.
    //      Jeder Ausdruck besitzt den SekundenZahler als Komponente.
    //      Damit ändert sich der Wert des Ausdruckes.
    //      Für die Neuberechnung des Ausdruckes ist der Aufruf von
    //      document.recal()
    //      nötig.
    //      Dieser Aufruf erfolgt in SekundenZaehlen(), also permanent pro Sekunde.
    //      Damit wird der Style-Wert permanent neu berechnet.
    //      Damit visualisieren sich die DIV's permanent neu.

    // Sekundenbalken in der Style-Eigenschaft width (Breite) mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_Balken.style.setExpression( "width",
        "SekundenZahler * PixelBreiteProBalkenErweiterung"
    );

    // Sekundenzähler in der Eigenschaft .innerText mit Ausdruck belegen,
    //      also dynamisch anzeigen
    ID_DIV_SekundenZahler.setExpression("innerText","SekundenZahler.toString()");

    // - - - Messlatte statisch anzeigen
    ID_DIV_MessLatte.style.width = SoundDauerInSekunden * PixelBreiteProBalkenErweiterung;
    ID_DIV_MessLatte.innerText = "Der Sound dauert "
        + SoundDauerInSekunden.toString()
        + " Sekunden";
}

// ##### Dieser Teil wird mit dem Laden des Dokumentes abgearbeitet #####
if (ie)
{
    document.write('<BODY></BODY>');

    document.write(      '<BGSOUND ID="ID_BGSound" LOOP="0">'

    document.write(      '<DIV ID="ID_DIV_Balken"'
        + 'STYLE="background-color:lightblue"'
        + '>'
        + '</DIV>'
        + '<BR>'
    );

    document.write(      '<DIV ID="ID_DIV_SekundenZahler"'
        + 'STYLE="color:hotpink;font-weight:bold"'
        + '>'
        + '</DIV>'
        + '<BR>'
    );

    document.write(      '<DIV ID="ID_DIV_MessLatte"'
        + 'STYLE="color:white;background-color:gray"'
        + '>'
        + '</DIV>'
    );
}

```



```

// ++++++ Sound initialisieren und starten mit Laden des Dokumentes

// ---- Sound-Objekt erzeugen anhand globaler Variablen
SoundObjekt = new SoundObjektErzeugen(SoundUrl, SoundDauerInSekunden);

// ---- Sound-Objekt wiedergeben
SoundAbspielen(); // meldet wenn Wiedergabe beendet ist
}
</SCRIPT>
</HEAD>
<BODY>
<!-- BODY-Teil muss leer bleiben -->
</BODY>
</HTML>

```

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
.align	Ausrichtung
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.begin	Wartezeit einstellen und dann Objekt aktivieren laut Eigenschaft .timeAction siehe Objekt currTimeState und Behavior .style.time2
.blockDirection	Umfluss um ein Objekt
.className	Klassenreferenz
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.contentEditable	Objekt-Content-Editierbarkeit (auch wenn Objekt kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.dataFld	Datenquelle-Name vergeben (ID)
.dataFormatAs	Datenquelle-Anzeigart
.dataSrc	Datenquelle als Anker festlegen
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.dur	Dauer Objektaktivitäten laut Eigenschaft .timeAction alternativ: Eigenschaft .end siehe Objekt currTimeState und Behavior .style.time2
.end	Objektaktivitäten laut Eigenschaft .timeAction beenden ab IE 6.x alternativ: Eigenschaft .dur siehe Objekt currTimeState und Behavior .style.time2
.hasMedia	Objekt ist HTML-Media-Objekt siehe Objekt currTimeState und Behavior .style.time2
.hideFocus	Focussierbarkeit
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID)
.innerHTML	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.innerText	Referenz auf den Bereich zwischen HTML-Start und -Ende-Tag ein Tag ohne Ende-Tag kann kein innerHTML haben, z.B. dient zur Veränderung dieses Bereiches zur Laufzeit des Dokumentes, also nach dem Laden des Objektes aber wirksam erst mit parsen des HTML-Endetag
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.noWrap	Wortumbruch einstellen
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem



.offsetLeft	des Elternobjektes (.offsetParent) X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern
.offsetTop	für Nutzung von .offsetHeigt, .offserLeft, .offsetTop und .offsetWidth Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.onOffBehavior	deprectated 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	nur nach kompletten Einlesen des Dokumentes nutzbar Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.recordNumber	Datenquelle-Satznummer eines Datenfeldes
.scroll	Scrollbar-Anzeige ein/aus vor IE 6.x nur BODY-Objekt ab IE 6.x alle HTML-Elemente wenn DOCTYPE im Kopf des Dokumentes kodiert
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
SYSTEMLANGUAGE	Sprache festlegen für das Objekt
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.timeAction	Aktion des Objektes in der Timeline Aktion nur ausführbar wenn Objekt aktiv ist UND Timeline aktiv ist siehe Objekt currTimeState und Behavior .style.time2
.timeContainer	Typ der Timeline des Objektes siehe Objekt currTimeState und Behavior .style.time2
.title	Tooltip-Text bei Mouse over über Objekt
UNSELECTABLE	Selektionsfähigkeit eines Objektes
Methoden:	
.addBehavior()	DHTML-Verhaltenseigenschaft einem Element hinzufügen Empfehlung: Standard-IE-Eigenschaften nutzen, da diese mit "#default#behaviorName" komplett erfasst werden und bereits im Browser implementiert sind (keine HTC-Datei nötig). ab IE 5.x bis unter IE 5.5
.appendChild()	Knoten als Kind an die DOM-Hierarchie anhängen und danach den Zeiger laut DOM liefern DOM wird geändert Zeiger wird zugleich immer am Ende der Collection childNodes angehangen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehangen wird UND Ende-Tag (falls vorhanden) des Knoten geparkt wurde
.applyElement()	Elementeigenschaft "Kind sein" oder "Eltern sein" festlegen, also die Lage im DOM, und danach Referenz laut DOM liefern DOM wird geändert Element kann selbst Kinder haben Element erst sichtbar, wenn Endetag (falls vorhanden) des Elementes geparkt wurde Achtung: Wenn Element per Methode .createElement() erzeugt wurde, aber nicht im im Dokumentenbaum eingebunden ist, so wird die Eigenschaft .innerHTML gelöscht !
.attachEvent()	Einschalten des Registrieren eines Events durch Eventhandler Hinweis: Abschalten mit Methode .detachEvent() Achtung: Wenn mehrere Eventhandler zum Event, so Aufruf der Handler leider



	NICHT verkettet sondern in Zufallsfolge , es sei denn die Handler prüfen ihre Aufruffolge (muss programmiert werden)
.blur()	Element den Focus wegnehmen und Event onblur auslösen Der Focus wird nicht automatisch auf irgend ein anderes Element gesetzt ! vor IE 5.0 TABINDEX-Attribut muss kodiert sein ab IE 5.0 TABINDEX-Attribut muss nicht kodiert sein
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.click()	simuliert einen Klick auf das Element und löst onclick-Event aus manipuliert nicht den Focus
.cloneNode()	Objekt klonen und Referenz des erzeugten Klone liefern DOM wird nicht geändert, da Klone nicht in DOM eingebunden wird (reines Neu-Instanzieren eines DOM-Objektes im Hauptspeicher außerhalb des DOM)
.componentFromPoint()	Layout-Komponente eines Objektes ermitteln, die an einer Koordinate liegt auch für CSS-Layout onmouseover-Event hat nicht die Pixelgenauigkeit wie die Angaben der Methode .componentFromPoint() also wenn Event erzeugt, muss die Maus noch lange nicht genau die obengenannte Pixelpos erreicht haben Overbereich der Maus ist mehr als 1 Pixel gross beachte Einstellungen der Maus zur Cursorgeschwindigkeit etc.
.contains()	prüfen ob Element innerhalb eines Elementes liegt, also ob das innere, eingeschlossene Element Eltern (Eltern-Objekt, Container) hat und somit ein Kind-Objekt ist DOM nicht geändert
.detachEvent()	Abschalten des Registrieren eines Events durch Eventhandler wobei Registrierung mit Methode .attachEvent() aktiviert wurde Abschalten = ordnet dem Window-Objekt das Event laut Parameter event_bezeichner zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.doScroll()	Click auf Scrollbar simulieren Achtung: Scrollelemente müssen bereits vorhanden sein !
.dragDrop()	prüfen des Status der letzten Drag-Manipulation (anklicken, ziehen, ablegen) auf Element
.fireEvent()	ein Event auslösen
.focus()	Focus setzen und Focus-Event auslösen nur nach dem kompletten Laden des Dokumentes vor IE 5.x: Objekt muss TABINDEX-Attribut besitzen
.getAdjacentText()	Text eines Objektes liefern, wobei Textlage im Objekt definiert werden kann Text kann HTML-Tags enthalten, muss aber nicht DOM nicht geändert
.getAttribute()	Wert eines per HTML erzeugten Attributes liefern DOM nicht geändert
.getAttributeNode()	Referenz auf Eigenschaft des attribute-Objektes liefern, also Zeiger auf attribute.name Eigenschaft. Eigenschaft kann mit HTML-Anweisung erzeugt worden sein, muss aber nicht Eigenschaft ist selbst ein Knoten in der Attribute-Objekt-Hierarchie zum Objekt Wert des Attributes wird somit über die Referenz laut DOM erreichbar DOM nicht geändert
.getBoundingClientRect()	Referenz auf TextRectangle-Objekt im Element holen
.getClientRects()	Referenz auf Feld der Zeiger auf TextRectangle-Objekte im Fenster Feld mit Index als Integer ab 0 pro Eintrag ein Rectangle
.getElementsByTagName()	Referenz auf ein Feld (Collection) aller im Objekt befindlichen Kinder-Objekte mit gemeinsamen Tagnamen liefern, inklusive aller Kinder und Unterkinder etc. Hinweis: Natürlich kann auch das document-Objekt so verarbeitet werden (beachte dabei document.all Collection) Achtung: Kinder-Objekte, die keinen Tag-Name besitzen, werden nicht erfasst ! Für Verwaltung per ID (analog zum ID-Attribut): siehe Methode getElementById() Für Verwaltung per NAME (analog zum NAME-Attribut): siehe Methode .getElementByName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren
.hasChildNodes()	DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh) prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.hasFocus()	Objekt im Focus-Zustand
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich



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



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

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

