object Objekt des Internet Explorer

OBJECT dient zum Ersatz von APPLET, da ab HTML 40.1 APPLET unerwünscht ist.
Container eines per OBJECT-Tag eingebetteten Objektes aus dem HEAD und /oder BODY-Teil des Dokumentes (z.B. ActiveX-Control), also nicht eines per new-Methode erzeugten Objektes.

Das Objekt ist **nicht** das JScript-Objekt object.

Dieser Container ist notwendig, um private Methoden und Eigenschaften des Objektes von den weiter unten genannten Methoden und Eigenschaften des OBJECT-Tags im DOM beim Zugriff unterscheiden zu können.

Bsp: Methode . click() ist eine im DOM implementierte Methode zum OBJECT-Tag
Wenn dem Objekt eine privaten Methode .click() implementiert ist, so erfolgt der Zugriff auf diese private Methode wie folgt:

document.all.objectID.**object**.click()

oder var ZeigerAufObjekt = **document.all.**objectID;
ZeigerAufObjekt**.object**.click();

wobei objectID z.B. aus dem ID-Attribut des OBJECT-Tags stammt.

Dieser Container verhindert das Überschreiben von im DOM gleichnamigen Methoden und Eigenschaften.
Er sollte immer verwendet werden, vorallem dann, wenn der DOM des Browsers sich geändert hat und der Programmierer davon nichts weiss.

Wenn Objekt nicht instanziert werden kann, so werden trotzdem alle innerhalb <OBJECT> .... </OBJECT> renderbaren Elemente angezeigt.

Events werden immer direkt zum Objekt gesendet !
Events sind per Script programmierbar.

siehe auch document.embeds Collection

Hinweis zu Direct Animation per ActiveX-Control:
Für das STYLE-Attribut des OBJECT-Tags ist **dringend anzuraten**, es **nicht** zu kodieren, sondern **alle** Style-Angaben ausschliesslich durch Referenz per zeiger_auf_da_objekt.style.style_wert = ...; zu erzeugen. Grund: Je nach DA-Objekt-Implementierung wird das STYLE-Attribut im OBJECT-Tag teilweise oder vollständig ignoriert und damit wirkungslos. Der Zeiger auf das DA-Objekt ist der Wert des ID-Attributes im OBJECT-Tag.

Beispiel:

```
<OBJECT CLASSID=" ..... " .....>
    <SPAN STYLE="color:red">
        Das Objekt ist nicht vom Browser instanzierbar !
    </SPAN>
</OBJECT>
```

Beispiel 1

```
<SCRIPT FOR= objectID EVENT= eEvent>
    :
</SCRIPT>
```

Beispiel 2

```
<OBJECT ID="objectID" CLASSID="xyz.abc">
</OBJECT>
```

Eigenschaften:

.accessKey	Tastaturzugriff auf ein Objekt per Alt + Taste bei Ausführung der Tastenkombination wird das Sprungziel wird focussiert die Sprungquelle defocussiert das Focus-Ereignis ausgelöst vor IE 5.x funktioniert Tastenkombination nicht mit jedem Objekt
.align	Ausrichtung
.alt	Alternativer Text als Tooltip
.altHTML	HTML-Script, der ausgeführt wird, wenn Objekt nicht geladen werden kann
.archive	Zeichenkette für Archive-Funktionalität eines Objektes
.BaseHref	Url des Dokumentes, das <OBJECT> .... </OBJECT> enthält (auch frame und iframe) ist meisten die Url des Dokumentes selbst
.border	Rahmendicke in Pixel
.canHaveChildren	prüfen ob Kind möglich ist, also ob Objekt Parent sein kann
.canHaveHTML	prüfen ob Objekt HTML-Tags enthalten darf
.classid	Klassenbezeichner (ID) eines damit dem Objekt zugeordneten ActiveX-Controls von Microsoft Windows (Microsoft eigene oder Fremdhersteller) dient als Ersatz für die browserspezifischen MIME-Typen

.className	Klassenreferenz, Klassenname
.clientHeight	Objekthöhe in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.clientLeft	Abstand in Pixel zum linken Rand des Fensters
.clientTop	Abstand in Pixel zum oberen Rand des Fensters
.clientWidth	Objekt-Breite in Pixel ohne Abstand zum Umgebungsobjekt ohne Rahmen ohne Scrollbalken
.code	Url der *.class-Datei (kompilierter Javacode)
.codeBase	Url der Komponente für Dowload der Komponente vom Server auf den Client optionale Versionsprüfung möglich (nicht bei Applet-Komponente) um unnötigen Download zu ersparen: vorheriger Ableich der Version der Komponente auf Server und Client auch wenn der Versionsabgleich keinen Download ergab, wird immer HTTP Header-Transaktion ausgelöst
.codeType	Internet Media Type (Mimetyp) des Codes zum Objekt, das per OBJECT-Tag eingebunden wurde
.data	Url der Daten des Objektes
.declare	Zeichenkette füir Declare-Funktionalität des Objektes, das per OBJECT-Tag eingebunden wurde
.dir	Umflussrichtung
.disabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.form	Zeiger auf das Formular (Formular als Container) ab IE 6.x für Elemente fieldSet, label, legend
.height	Höhe des Objektes in Pixel
.hideFocus	Focussierbarkeit
.hspace	horizontaler Abstand in Pixel zum Elternobjekt
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) Hinweis: Browser erzeugt pro Objekt ein internes ID, das per Eigenschaft .uniqueID ermittelt und anstelle der Eigenschaft .id verwendet werden kann (falls Browser und betroffenes Objekt die Eigenschaft .uniqueID kennen). Zeiger aus ID bilden var Zeiger = eval(object.id);
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat) Content = Beziehung des Objektes zum Umfeld z.B. bezüglich Layout etc.
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.language	Sprache für Script festlegen
.name	Name des Objektes (nicht ID !!!) muss beim Formular für alle zu sendenden Felder kodiert sein !! Element darf nicht per Methode .createElement() erzeugt worden sein
.nextSibling	Zeiger auf das NACHFOLGENDE Kind laut childNodes collection eines Objektes
.nodeName	String als Name des Kindes (Knoten, Node, Element) also TAG-Bezeichner, Attribut-Name; #text für Anker
.nodeType	Knotentyp laut attributes Collection
.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.object	Zeiger auf das Objekt laut Applet bzw. laut Objekt object
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .offsetHeigt, .offserLeft, .offsetTop und .offsetWidth
.offsetTop	Y-Koordinate der linken oberen Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offestParent)
.offsetWidth	X-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offestParent)
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar
.outerText	Referenz auf den gesamten Plain-Text im Objekt nur nach kompletten einlesen des Dokumentes nutzbar
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten
.recordset	Zeiger des Standard-Record-Set in einem Datasource-Objekt (DSO) Methode . namedRecordset() liefert den Zeiger für Datanquellen-Objekt mit Namen also eines beliebige Mitgliedes im Datasource-Objekt (DSO)

.scopeName	Namensraum laut XMLNS-Attribut
.scrollHeight	Höhe des vertikalen Scrollbereiches, also Abstand von oberen und unteren sichtbaren Rand des Umgebungsobjektes
.scrollLeft	Abstand von linken sichtbaren Randes des Umgebungsobjektes zum linken Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes Hinweis: Eigenschaften .scrollLeft und .scrollTop sind **nicht** beschreibbar, solange eine Scrollaktion aktiv ist.
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes Hinweis: Eigenschaften .scrollLeft und .scrollTop sind **nicht** beschreibbar, solange eine Scrollaktion aktiv ist.
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
.standby	Zeichenkette für Standy-Funktionalität des per OBJECT-Tag eingebundenen Objektes
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.tabIndex	Index des Elementes in der Tab-Tasten-Folge für Anspringen des Dokumentes Anspringen verbunden mit Focus erhalten --> Ereignisse werden ausgelöst !! unter IE 5.x onblur, onfocus ab IE 5.x onblur, onfocus, onkeydown, onkeypress, onkeyup Anspringen default per TAB-Taste für A, BODY, BUTTON, FRAME, IFRAME, IMG, INPUT, SELECT, TEXTAREA Anspringen default nicht per TAB-Taste für APPLET, DIV, FRAMESET, SPAN, TABLE, TD
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.title	Tooltip-Text bei Mouse over über Objekt
.type	MIME-Typ des Objektes (Multipurpose Internet Mail Extension) wenn die Eigenschaft .classid nicht kodiert wird, dann immer .type kodieren bzw. die im Browserstandard enthaltenenen MIME verwenden Hinweis: MIME wird von Simple Mail Transfer Protocol (SMTP) benutzt z.B. bei Audio, Images, Video, Texten
.uniqueID	durch den Browser automatisch-genriertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiede ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
UNSELECTABLE	Selektionsfähigkeit eines Objektes
.useMap	Url oder Anker für client-seitige Image-Map
.vspace	vertikaler Abstand in Pixel zum Elternobjekt
.width	Breite des Objektes in Pixel

Methoden:

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

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

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

.fireEvent() ein Event auslösen

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

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

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

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

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

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

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

.insertAdjacentElement() Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann
wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM
nur nach dem kompletten Laden des Dokumentes möglich
DOM wird geändert

.insertBefore() Objekt als Kindknoten VOR dem einem anderen Kind-Objekt einfügen und Zeiger liefern
einzufügendes Objekt muss mit Methode createElement() erzeugt worden sein
Achtung: NICHT anwenden für einfügen VON bzw. VOR obersten Kindknoten
Sichtbarkeit erst wenn Ende-Tag geparst wurde
DOM wird geändert

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

.namedRecordset() Zeiger desjenigen Datanquellen-Record-Objektes mit Namen, das den Standard-Record-Set enthält
Hinweis: Namen zeigt die Mitgliedschaft in einem Datasource-Objekt (DSO) an
Eigenschaft .recordset liefert den Zeiger für den Standard-Record-Set in einem Datasource-Objekt (DSO)

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

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

.removeAttribute() entfernen eines per HTML erzeugten Attributes
Achtung: Der Browser unterscheidet zwischen HTML-erzeugte oder mit dieser Methode erzeugte Attribute!
per Methode .createAttribute() erzeugte Attribute werden nicht erfasst
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
.removeNode()	Knoten entfernen aus DOM und Referenz auf den entfernten Knoten liefern
	Sichtbarkeit erst wenn Ende-Tag geparst wurde
	DOM wird geändert
.replaceAdjacentText()	Plain-Text (ohne HTML und Script) eines Elementes durch anderen Text ersetzen und Referenz auf den zu ersetzenden Text liefern
	DOM wird nicht geändert
.replaceNode()	Objekt durch anderes Objekt komplett ersetzen und Referenz auf das komplett ersetzte Objekt liefern
	sichtbar erst mit parsen des Endetags
	DOM wird geändert
.scrollIntoView()	Objekt derart scrollen, dass es im Fenster für User sichtbar wird
	Objekt muss an sich schon renderbar sein
.setActive()	Objekt für die Eventdurchreichung aktivieren aber ohne es zu fokussieren und ohne es scrollbar zu machen
.setAttribute()	Wert von vorhandenem Attribut setzen
	wenn Attribut nicht vorhanden, so wird es automatisch erzeugt und mit dem Wert gefüllt
	DOM wird nur bei Erzeugung geändert
.setAttributeNode()	Attribut einem Knoten zuweisen und Referenz liefern
	DOM wird geändert
.setCapture()	Maus-Überwachung einschalten für ein Objekt
	Maus-Events sind : onmousedown, onmouseup, onmousemove, onclick, ondblclick, onmouseover und onmouseout.
	ab IE 5.5
	Hinweis: ausschalten per Methode .releaseCapture()
.setExpression()	Wert definieren, der als Ausdruck für die Methode .getExpression() zur Berechnung einer Style-Eigenschaft als Objektreferenz der Form objekt.style.eigenschaft. dient
	Ausdruck nur als Script kodierbar
	DOM wird nicht geändert
.swapNode()	Positionen von 2 Knoten im DOM tauschen (Zeigertausch)
	nur sichtbar wenn Endetag geparst
	DOM wird geändert