

table.colGroup Objekt des Internet Explorer

Objekt zur Gruppierung von Spalten einer Tabelle

COL kann innerhalb von COLGROUP kodiert sein:

COL **erbt keine** gleichnamige Eigenschaften von COLGROUP
überschreibt gleichnamige Eigenschaften von COLGROUP

Bsp: Es sollte das SPAN-Attribut innerhalb von COL kodiert werden, wenn SPAN in COLGROUP mit einem anderen Wert hat als dem Standardwert des SPAN-Attributes von COL kodiert wurde.

Beispiel 1:

```
<TABLE BORDER="2" RULES="groups">
  <COLGROUP SPAN="2" STYLE="color:red">
</COLGROUP>
  <COLGROUP STYLE="color:blue">
</COLGROUP>
  ....
</TABLE>
```

Beispiel 2:

```
<TABLE BORDER="2">
  <COLGROUP SPAN="3" STYLE="color:green;background:black">
    <COL SPAN="2" STYLE="color:red">
  </COLGROUP>
  ....
</TABLE>
```

Eigenschaften:

.align	Ausrichtung
ATOMICSELECTION	Selektierbarkeit des Objektes einstellen
.bgColor	deprecated und durch STYLE-Attribut zu ersetzen
.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
.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
.dir	Umflussrichtung
.firstChild	Zeiger auf das ERSTE Kind laut childNodes-Collection eines Objektes
.id	Bezeichner des Objektes für Referenzierung des Objektes (Zeiger, ID, logischer Objektname) ID-Attribut in HTML: Wert ist String alphanumerisch muss mit Buchstaben beginnen Unterstrich _ verwendbar kann in " " bzw. ' ' kodiert werden, muss aber nicht
	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);
	Die Verwendung des ID-Attributes erhöht die Performance des Browsers und ist zu empfehlen.
.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
	Plain-Text HTML-Elemente je nach Objekt möglich Script: muss mit DEFER-Attribut kodiert sein schreiben: wenn Bereich nicht leer, so komplett überschreiben wenn Bereich leer so einfügen
	nur lesen bei COL, COLGROUP, FRAMESET, HTML, STYLE, TABLE, TBODY, TFOOT, THEAD, TITLE, TR.
.isContentEditable	Editierbarkeit des Objekt-Content (auch wenn kein Layout hat)
.isDisabled	Interaktionsfähigkeit nur wenn sichtbar so User-Interaktion möglich
.isMultiLine	Mehrzeiligkeit des Objektinhaltes
.isTextEdit	Erzeugbarkeit eines Textbereiches
.lang	Sprache für Anzeige von Sonderzeichen etc.
.lastChild	Zeiger auf das LETZTE Kind laut childNodes collection eines Objektes
.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 1 für Element-Knoten. 3 für Textknoten. oder null (nicht 0 !!) wenn Knoten nicht vorhanden



.nodeValue	Knotenwert (Wert des Kindes, Node, Elementes) nur für Text- und Attribut-Elemente nicht für Element-Knoten (Knotentyp 1)
.offsetHeight	Y-Koordinate der rechten unteren Ecke des Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetLeft	X-Koordinate der linken oberen Ecke Objektes bezüglich Koordinatensystem des Elternobjektes (.offsetParent)
.offsetParent	Referenz der Eltern für Nutzung von .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 des Elternobjektes (.offsetParent)
.onOffBehavior	deprecated ab IE 5.x Unterstützung von DirectAnimation z.B. für 2D, 3D, Sound
.outerHTML	Referenz auf Bereich ab inklusive HTML-Start bis hinter -Ende-Tag wirksam mit parsen des Ende-Tag nur nach kompletten Einlesen des Dokumentes nutzbar schreiben: wenn Bereich gefüllt so komplett überschreiben Plain-Text HTML-Elemente möglich nur lesen bei CAPTION, COL, COLGROUP, FRAMESET, HTML, TBODY, TD, TFOOT, TH, THEAD, TR.
.ownerDocument	Referenz auf das document Objekt zu dem der Knoten gehört, also in dem der Knoten erzeugt wurde
.parentElement	Referenz auf das Elternobjekt, also nicht Elternknoten innerhalb DOM
.parentNode	Referenz auf Elternknoten innerhalb der DOM-Hierarchie
.parentTextEdit	Textbereich des Elternobjektes referenzieren
.previousSibling	Referenz auf das Vorgängerkind
.readyState	aktueller Status des Objektes beim Füllen des Objektes mit Daten "uninitialized" Objekt ist nicht initialisiert "loading" Objekt ist nicht initialisiert aber lädt gerade Daten zur Initialisierung "loaded" Objekt hat alle Daten komplett geladen und ist komplett initialisiert "interactive" Objekt kann vom User interaktiv verwendet werden zum Füllen mit Daten "complete" Objekt hat alle Daten geladen und ist mit diesen komplett initialisiert
.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 immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollTop	Abstand von oberen sichtbaren Randes des Umgebungsobjektes zum oberen Rand des Objektes nur nutzbar nach dem kompletten Laden des Dokumentes immer 0 wenn Scrollbars im Objekt nicht vorhanden bzw. nicht erlaubt sind
.scrollWidth	Breite des horizontalen Scrollbereiches, also Abstand von linken und rechten sichtbaren Rand des Umgebungsobjektes
.sourceIndex	Index des Objektes in der Collection document.all
.span	Anzahl der Spalten innerhalb einer Spaltengruppe einer Tabelle
Beispiel:	<pre><TABLE BORDER="2"> <COLGROUP SPAN="3" STYLE="color:green;background:black"> <COL SPAN="2" STYLE="color:red"> </COLGROUP> </TABLE></pre>
STYLE	direkt im HTML-Element kodierter Style (Inline-Style) Hinweis: für Scripting ist das Style-Objekt zu nutzen
.tagName	Tag-Bezeichner des Objektes
.tagUrn	Uniform Resource Name (URN) laut Namensraum laut XMLNS-Attribut
.uniqueID	durch den Browser automatisch-generiertes ID des Objektes Browser generiert zu verschiedenen Zeitpunkten auch verschiedene ID, wenn Objekt mehrmals geladen wurde kann anstelle eines privat vergebenen ID als ID-Attributwert weiterverwendet werden
.vAlign	vertikale Ausrichtung des Inhaltes z.B. Text eines Objektes "middle" zentriert, Standard "baseline" Basislinie der 1. Zeile eines Textes im Objekt ausrichten an der Basislinie des benachbarten Objektes "bottom" am Fuss "top" am Kopf
.width	Breite des Objektes in Pixel Integer in Pixels für absolute Breite, >=0 bei Tabelle-Zelle und Spalte als Elternobjekt: maximal 32750 Pixel String Ziffernfolge eines Integer mit nachfolgendem % für Breite als Anteil der Breite des Elternobjektes z.B. 10%

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 angehängen Anhängen wird danach nur sichtbar, wenn zusätzlich dem BODY-Objekt angehängen 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 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)
.clearAttributes()	alle HTML-Attribute eines Objektes entfernen außer ID, STYLE und per Script definierte Attribute Script-erzeugte Attribute nicht entfernbar DOM wird geändert
.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_bezeichnet zu, das nicht behandelt werden soll, falls es für das Window-Objekt auftritt (also Standardbehandlung aktiv)
.fireEvent()	ein Event auslösen
.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 .getElementsByName()
.getExpression()	DOM nicht geändert Wert einer Style-Eigenschaft anhand des Ausdruckes berechnen und liefern Style-Eigenschaft ist per Methoden expression() oder setExpression() zu definieren DOM wird nicht verändert (nur Werteveränderung), aber das Dokument-Layout (nach dem eventuellen expliziten Dokument-Refresh)



.hasChildNodes()	prüfen auf Existenz von Kinder(n) als HTML-Elemente oder Textknoten (Textelemente) in einem Objekt DOM nicht geändert
.insertAdjacentElement()	Objekt in eine Objekt einfügen und Referenz liefern, wobei die Lage definiert werden kann wenn Element bereits eingefügt vorhanden, so wird dieses nur verschoben laut Lage des Objektes im DOM nur nach dem kompletten Laden des Dokumentes möglich DOM wird geändert
.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 geparkt 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
.normalize()	Normalisierung des DOM zur Erreichung einer konsistenten Struktur Achtung: CDATA-Sections dürfen nicht enthalten sein, da diese immer Inkonsistenz erzeugen
.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
.removeChild()	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 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 geparkt 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
.replaceChild()	Kind-Objekt ersetzen durch ein Objekt ersetzende Objekt muss per Methode .createElement() erzeugt worden sein Sichtbarkeit erst wenn Ende-Tag geparkt wurde DOM wird 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
.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
.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

