**DOM参考手册**

**Anchor 对象**

Anchor 对象代表一个 HTML 超链接。在 HTML 文档中 <a> 标签每出现一次，Anchor 对象就会被创建。锚可用于创建到另一个文档的链接（通过 href 属性），或者创建文档内的书签。

**Anchor 对象的属性**

属性描述：

* accessKey 设置或返回访问一个链接的快捷键。
* charset 设置或返回被链接资源的字符集。
* coords 设置或返回逗号分隔列表，其中包含了图像映射中一个链接的坐标。
* href 设置或返回被链接资源的 URL。
* hreflang 设置或返回被链接资源的语言代码。
* id 设置或返回一个链接的 id。
* innerHTML 设置或返回一个链接的内容。
* name 设置或返回一个链接的名称。
* rel 设置或返回当前文档与目标 URL 之间的关系。
* rev 设置或返回目标 URL 与之间当前文档的关系。
* shape 设置或返回图像映射中某个链接的形状。
* tabIndex 设置或返回某个链接的 Tab 键控制次序。
* target 设置或返回在何处打开链接。
* type 设置或返回被链接资源的 MIME 类型。

标准属性：

* className 设置或返回元素的 class 属性。
* dir 设置或返回文本的方向。
* lang 设置或返回元素的语言代码。
* title 设置或返回元素的 title。

属性说明：

1、accessKey 属性可设置或返回访问一个链接的键盘按键。请使用 Alt + accessKey 为拥有指定快捷键的元素赋予焦点。

语法：

anchorObject.accessKey=accessKey

实例： document.getElementById('w3').accessKey="w"

2、tabIndex 属性可为链接设置或返回 tab 键控制次序。

语法：

anchorObject.tabIndex=tabIndex

实例：下面的例子可更改三个链接的 tab 键控制次序：

function changeTabIndex() {

document.getElementById('1').tabIndex="3"

document.getElementById('2').tabIndex="2"

document.getElementById('3').tabIndex="1"

}

3、target 属性可设置或返回在何处打开链接。

语法：

anchorObject.target=\_blank|\_parent|\_self|\_top 四个保留的目标名称：

* \_blank - 在一个新的未命名的窗口载入文档
* \_self - 在相同的框架或窗口中载入目标文档
* \_parent - 把文档载入父窗口或包含了超链接引用的框架的框架集
* \_top - 把文档载入包含该超链接的窗口,取代任何当前正在窗口中显示的框架

实例：document.getElementById('myAnchor').target="\_blank";

**Anchor 对象的方法**

方法描述：

* blur() 把焦点从链接上移开
* focus() 给链接应用焦点

**Body 对象**

Body 对象代表文档的主体 (HTML body) 。

**Body 对象的属性**

属性描述：

* className 设置或返回元素的 class 属性
* dir 设置或返回文本的方向
* id 设置或返回 body 的 id
* lang 设置或返回元素的语言代码
* title 设置或返回元素的咨询性的标题

**Button 对象**

Button 对象代表一个按钮。在 HTML 文档中 <button> 标签每出现一次，Button 对象就会被创建。

**Button 对象的属性**

属性描述：

* accessKey 设置或返回访问某个按钮的快捷键。
* alt 设置或返回当浏览器无法显示按钮时供显示的替代文本。
* disabled 设置或返回是否禁用按钮。
* form 返回对包含按钮的表单的引用。
* id 设置或返回按钮的 id。
* name 设置或返回按钮的名称。
* tabIndex 设置或返回按钮的 Tab 键控制次序。
* type 返回按钮的表单类型。
* value 设置或返回显示在按钮上的文本。

属性说明：

1、disabled 属性可设置或返回是否禁用按钮。

语法：

buttonObject.disabled=true|false

实例：document.getElementById("myButton").disabled=true

2、form 属性可返回对包含该按钮的表单的引用。如果包含该按钮，则返回一个 form 对象，如果未包含该按钮，则返回 null。

语法：

buttonObject.form

实例：下面的例子可返回该按钮所属的表单的 id：

<p>The form containing the button is:

<script type="text/javascript">

x=document.getElementById('button1');

document.write(x.form.id);

</script></p>

3、type 属性可返回表单元素的类型。对于一个 Buttom 对象，则永远是 "button" 类型。

语法：

buttonObject.type

实例：alert(document.getElementById("myButton").type)

**Button 对象的方法**

方法描述：

* blur() 把焦点从元素上移开。
* click() 在某个按钮上模拟一次鼠标单击。
* focus() 为某个按钮赋予焦点。

**Checkbox 对象**

Checkbox 对象代表一个 HTML 表单中的 一个选择框。在 HTML 文档中 <input type="checkbox"> 每出现一次，Checkbox 对象就会被创建。

**Checkbox 对象的属性**

属性描述：

* accessKey 设置或返回访问 checkbox 的快捷键。
* alt 设置或返回当浏览器不支持 checkbox 时供显示的替代文本。
* checked 设置或返回 checkbox 是否应被选中。
* defaultChecked 返回 checked 属性的默认值。
* disabled 设置或返回 checkbox 是否应被禁用。
* form 返回对包含 checkbox 的表单的引用。
* id 设置或返回 checkbox 的 id。
* name 设置或返回 checkbox 的名称。
* tabIndex 设置或返回 checkbox 的 tab 键控制次序。
* type 返回 checkbox 的表单元素类型。
* value 设置或返回 checkbox 的 value 属性的值

**Checkbox 对象的方法**

方法描述：

* blur() 从 checkbox 上移开焦点。
* click() 模拟在 checkbox 中的一次鼠标点击。
* focus() 为 checkbox 赋予焦点。

**Document 对象**

Document 对象代表整个 HTML 文档，可用来访问页面中的所有元素。Document 对象是 Window 对象的一个部分，可通过 window.document 属性来访问。

**Document 对象的集合**

集合说明：

* all[] 提供对文档中所有 HTML 元素的访问。
* anchors[] 返回对文档中所有 Anchor 对象的引用。
* applets[] 返回对文档中所有 Applet 对象的引用。
* forms[] 返回对文档中所有 Form 对象引用。
* images[] 返回对文档中所有 Image 对象引用。
* links[] 返回对文档中所有 Area 和 Link 对象引用。

**Document 对象的属性**

属性描述：

* body 提供对 <body> 元素的直接访问。对于定义了框架集的文档，该属性引用最外层的 <frameset>。
* cookie 设置或返回与当前文档有关的所有 cookie。
* domain 返回当前文档的域名。
* lastModified 返回文档被最后修改的日期和时间。
* referrer 返回载入当前文档的文档的 URL。
* title 返回当前文档的标题。
* URL 返回当前文档的 URL。
* **Document 对象的方法**

方法描述：

* close() 关闭用 document.open() 方法打开的输出流，并显示选定的数据。
* getElementById() 返回对拥有指定 id 的第一个对象的引用。
* getElementsByName() 返回带有指定名称的对象集合。
* getElementsByTagName() 返回带有指定标签名的对象集合。
* open() 打开一个流，以收集来自任何 document.write() 或 document.writeln() 方法的输出。
* write() 向文档写 HTML 表达式 或 JavaScript 代码。
* writeln() 等同于 write() 方法，不同的是在每个表达式之后写一个换行符。

方法说明：

1、close() ：关闭一个由 document.open 方法打开的输出流，并显示选定的数据。

语法：

document.close()

说明：

该方法将关闭 open() 方法打开的文档流，并强制地显示出所有缓存的输出内容。如果您使用 write() 方法动态地输出一个文档，必须记住当你这么做的时候要调用 close() 方法，以确保所有文档内容都能显示。一旦调用了 close()，就不应该再次调用 write()，因为这会隐式地调用 open() 来擦除当前文档并开始一个新的文档。

实例：

<html>

<head>

<script type="text/javascript">

function createNewDoc() {

var newDoc=document.open("text/html","replace");

var txt="<html><body>Learning about the DOM is FUN!</body></html>";

newDoc.write(txt);

newDoc.close();

}

</script>

</head>

<body>

<input type="button" value="Write to a new document"

onclick="createNewDoc()">

</body>

</html>

2、open() ：打开一个新文档，并擦除当前文档的内容。

语法：

document.open(mimetype,replace)

参数描述：

mimetype 可选。规定正在写的文档的类型。默认值是 "text/html"。

replace 可选。当此参数设置后，可引起新文档从父文档继承历史条目。

说明：

该方法将擦除当前 HTML 文档的内容，开始一个新的文档，新文档用 write() 方法或 writeln() 方法编写。调用 open() 方法打开一个新文档并且用 write() 方法设置文档内容后，必须记住用 close 方法关闭文档，并迫使其内容显示出来。属于被覆盖的文档的一部分的脚本或事件句柄不能调用该方法，因为脚本或事件句柄自身也会被覆盖。

3、getElementById() ：返回对拥有指定 ID 的第一个对象的引用。

语法：

document.getElementById(id)

说明：

getElementById() 是一个重要的方法，在 DOM 程序设计中，它的使用非常常见。我们为您定义了一个工具函数，这样您就可以通过一个较短的名字来使用 getElementById() 方法了：

function id(x) {

if (typeof x == "string")

return document.getElementById(x);

return x;

}

这个函数接受元素 ID 作为它们的参数。对于每个这样的参数，您只要在使用前编写 x = id(x) 就可以了。

4、getElementsByName() ：返回带有指定名称的对象的集合。

语法：

document.getElementsByName(name)

说明：

因为一个文档中的 name 属性可能不唯一（如 HTML 表单中的单选按钮通常具有相同的 name 属性），所有 getElementsByName() 方法返回的是元素的数组，而不是一个元素。

5：getElementsByTagName() ：返回带有指定标签名的对象的集合。

语法：

document.getElementsByTagName(tagname)

说明：传递给 getElementsByTagName() 方法的字符串可以不区分大小写。

**Link 对象**

Link 对象代表某个 HTML 的 <link> 元素。<link> 元素可定义两个链接文档之间的关系。<link> 元素被定义于 HTML 文档的 head 部分。

**Link 对象的属性**

属性描述：

* charset 设置或返回目标 URL 的字符编码
* disabled 设置或返回目标 URL 是否当被禁用
* href 设置或返回被链接资源的 URL
* hreflang 设置或返回目标 URL 的基准语言
* id 设置或返回某个 <link> 元素的 id
* media 设置或返回文档显示的设备类型
* name 设置或返回 <link> 元素的名称
* rel 设置或返回当前文档与目标 URL之间的关系
* rev 设置或返回目标 URL 与当前文档之间的关系
* type 设置或返回目标 URL 的 MIME 类型
* dir 设置或返回文本的方向。
* lang 设置或返回元素的语言代码。

**Option 对象**

Option 对象代表 HTML 表单中下拉列表中的一个选项。在 HTML 表单中 <option> 标签每出现一次，一个 Option 对象就会被创建。

**Option 对象的属性**

属性描述：

* defaultSelected 返回 selected 属性的默认值。
* disabled 设置或返回选项是否应被禁用。
* form 返回对包含该元素的 <form> 元素的引用。
* id 设置或返回选项的 id。
* index 返回下拉列表中某个选项的索引位置。
* label 设置或返回选项的标记(仅用于选项组option-groups)。
* selected 设置或返回 selected 属性的值。
* text 设置或返回某个选项的纯文本值。
* value 设置或返回被送往服务器的值。

**Radio 对象**

Radio 对象代表 HTML 表单中的单选按钮。在 HTML 表单中 <input type="radio"> 每出现一次，一个 Radio 对象就会被创建。单选按钮是表示一组互斥选项按钮中的一个。当一个按钮被选中，之前选中的按钮就变为非选中的。当单选按钮被选中或不选中时，该按钮就会触发 onclick 事件句柄。

**Radio 对象的属性**

属性描述：

* accessKey 设置或返回访问单选按钮的快捷键。
* alt 设置或返回在浏览器不支持单选按钮时供显示的替代文本。
* checked 设置或返回单选按钮的状态。
* defaultChecked 返回单选按钮的默认状态。
* disabled 设置或返回是否禁用单选按钮。
* form 返回一个对包含此单选按钮的表单的引用。
* id 设置或返回单选按钮的 id。
* name 设置或返回单选按钮的名称。
* tabIndex 设置或返回单选按钮的 tab 键控制次序。
* type 返回单选按钮的表单类型。
* value 设置或返回单选按钮的 value 属性的值。

**Radio 对象的方法**

方法描述：

* blur() 从单选按钮移开焦点。
* click() 在单选按钮上模拟一次鼠标点击。
* focus() 为单选按钮赋予焦点。

**Select 对象**

Select 对象代表 HTML 表单中的一个下拉列表。在 HTML 表单中，<select> 标签每出现一次，一个 Select 对象就会被创建。

**Select 对象的集合**

集合说明：

* options[] 返回包含下拉列表中的所有选项的一个数组。

**Select 对象的属性**

属性描述：

* disabled 设置或返回是否应禁用下拉列表。
* form 返回对包含下拉列表的表单的引用。
* id 设置或返回下拉列表的 id。
* length 返回下拉列表中的选项数目。
* multiple 设置或返回是否选择多个项目。
* name 设置或返回下拉列表的名称。
* selectedIndex 设置或返回下拉列表中被选项目的索引号。
* size 设置或返回下拉列表中的可见行数。
* tabIndex 设置或返回下拉列表的 tab 键控制次序。
* type 返回下拉列表的表单类型。

**Select 对象的方法**

方法描述：

* add() 向下拉列表添加一个选项。
* blur() 从下拉列表移开焦点。
* focus() 在下拉列表上设置焦点。
* remove() 从下拉列表中删除一个选项。

方法说明：

1、add() ：向 <select> 添加一个 <option> 元素。

语法：

selectObject.add(option,before)

参数描述：

* option 必需。要添加选项元素。必需是 option 或 optgroup 元素。
* before 必需。在选项数组的该元素之前增加新的元素。如果该参数是null，元素添加到选项数组的末尾。

实例：下面的例子可向下拉列表的末尾添加一个 "kiwi" 选项：

function insertOption(){

var y=document.createElement('option');

y.text='Kiwi'

var x=document.getElementById("mySelect");

try {

x.add(y,null); // standards compliant

}

catch(ex) {

x.add(y); // IE only

}

}

2、remove() ：从下拉列表删除选项。

语法：

selectObject.remove(index)

参数描述：

index 必需。规定要删除的选项的索引号。

说明：

该方法从选项数组的指定位置移除 <option> 元素。如果指定的下标比 0 小，或者大于或等于选项的数目，remove() 方法会忽略它并什么也不做。

**Style 对象**

Style 对象代表一个单独的样式声明。可从应用样式的文档或元素访问Style 对象。使用Style 对象属性的语法：：document.getElementById("id").style.property="值"。

**Style 对象的属性：**

**background属性**

属性描述：

* background 在一行中设置所有的背景属性
* backgroundAttachment 设置背景图像是否固定或随页面滚动
* backgroundColor 设置元素的背景颜色
* backgroundImage 设置元素的背景图像
* backgroundPosition 设置背景图像的起始位置
* backgroundPositionX 设置backgroundPosition属性的X坐标
* backgroundPositionY 设置backgroundPosition属性的Y坐标
* backgroundRepeat 设置是否及如何重复背景图像

**Border 和 Margin 属性**

属性描述：

* border 在一行设置四个边框的所有属性
* borderBottom 在一行设置底边框的所有属性
* borderBottomColor 设置底边框的颜色
* borderBottomStyle 设置底边框的样式
* borderBottomWidth 设置底边框的宽度
* borderColor 设置所有四个边框的颜色 (可设置四种颜色)
* borderLeft 在一行设置左边框的所有属性
* borderLeftColor 设置左边框的颜色
* borderLeftStyle 设置左边框的样式
* borderLeftWidth 设置左边框的宽度
* borderRight 在一行设置右边框的所有属性
* borderRightColor 设置右边框的颜色
* borderRightStyle 设置右边框的样式
* borderRightWidth 设置右边框的宽度
* borderStyle 设置所有四个边框的样式 (可设置四种样式)
* borderTop 在一行设置顶边框的所有属性
* borderTopColor 设置顶边框的颜色
* borderTopStyle 设置顶边框的样式
* borderTopWidth 设置顶边框的宽度
* borderWidth 设置所有四条边框的宽度 (可设置四种宽度)
* margin 设置元素的边距 (可设置四个值)
* marginBottom 设置元素的底边距
* marginLeft 设置元素的左边距
* marginRight 设置元素的右边据
* marginTop 设置元素的顶边距
* outline 在一行设置所有的outline属性
* outlineColor 设置围绕元素的轮廓颜色
* outlineStyle 设置围绕元素的轮廓样式
* outlineWidth 设置围绕元素的轮廓宽度
* padding 设置元素的填充 (可设置四个值)
* paddingBottom 设置元素的下填充
* paddingLeft 设置元素的左填充
* paddingRight 设置元素的右填充
* paddingTop 设置元素的顶填充

**Layout 属性**

属性描述：

* clear 设置在元素的哪边不允许其他的浮动元素
* clip 设置元素的形状
* content 设置元信息
* counterIncrement 设置其后是正数的计数器名称的列表。其中整数指示每当元素出现时计数器的增量。默认是1。
* counterReset 设置其后是正数的计数器名称的列表。其中整数指示每当元素出现时计数器被设置的值。默认是0。
* cssFloat 设置图像或文本将出现（浮动）在另一元素中的何处。
* cursor 设置显示的指针类型
* direction 设置元素的文本方向
* display 设置元素如何被显示
* height 设置元素的高度
* markerOffset 设置marker box的principal box距离其最近的边框边缘的距离
* marks 设置是否cross marks或crop marks应仅仅被呈现于page box边缘之外
* maxHeight 设置元素的最大高度
* maxWidth 设置元素的最大宽度
* minHeight 设置元素的最小高度
* minWidth 设置元素的最小宽度
* overflow 规定如何处理不适合元素盒的内容
* verticalAlign 设置对元素中的内容进行垂直排列
* visibility 设置元素是否可见
* width 设置元素的宽度

**List 属性**

属性描述：

* listStyle 在一行设置列表的所有属性
* listStyleImage 把图像设置为列表项标记
* listStylePosition 改变列表项标记的位置
* listStyleType 设置列表项标记的类型

**Positioning 属性**

属性描述：

* bottom 设置元素的底边缘距离父元素底边缘的之上或之下的距离
* left 置元素的左边缘距离父元素左边缘的左边或右边的距离
* position 把元素放置在static, relative, absolute 或 fixed 的位置
* right 置元素的右边缘距离父元素右边缘的左边或右边的距离
* top 设置元素的顶边缘距离父元素顶边缘的之上或之下的距离
* zIndex 设置元素的堆叠次序

**Printing 属性**

属性描述：

* orphans 设置段落留到页面底部的最小行数
* page 设置显示某元素时使用的页面类型
* pageBreakAfter 设置某元素之后的分页行为
* pageBreakBefore 设置某元素之前的分页行为
* pageBreakInside 设置某元素内部的分页行为
* size 设置页面的方向和尺寸
* widows 设置段落必须留到页面顶部的最小行数

**Scrollbar 属性 (IE-only)**

属性描述：

* scrollbar3dLightColor 设置箭头和滚动条左侧和顶边的颜色
* scrollbarArrowColor 设置滚动条上的箭头颜色
* scrollbarBaseColor 设置滚动条的底色
* scrollbarDarkShadowColor 设置箭头和滚动条右侧和底边的颜色
* scrollbarFaceColor 设置滚动条的表色
* scrollbarHighlightColor 设置箭头和滚动条左侧和顶边的颜色，以及滚动条的背景
* scrollbarShadowColor 设置箭头和滚动条右侧和底边的颜色
* scrollbarTrackColor 设置滚动条的背景色

**Table 属性**

属性描述：

* borderCollapse 设置表格边框是否合并为单边框，或者像在标准的HTML中那样分离。
* borderSpacing 设置分隔单元格边框的距离
* captionSide 设置表格标题的位置
* emptyCells 设置是否显示表格中的空单元格
* tableLayout 设置用来显示表格单元格、行以及列的算法

**Text 属性**

属性描述：

* color 设置文本的颜色
* font 在一行设置所有的字体属性
* fontFamily 设置元素的字体系列。
* fontSize 设置元素的字体大小。
* fontSizeAdjust 设置/调整文本的尺寸
* fontStretch 设置如何紧缩或伸展字体
* fontStyle 设置元素的字体样式
* fontVariant 用小型大写字母字体来显示文本
* fontWeight 设置字体的粗细
* letterSpacing 设置字符间距
* lineHeight 设置行间距
* quotes 设置在文本中使用哪种引号
* textAlign 排列文本
* textDecoration 设置文本的修饰
* textIndent 缩紧首行的文本
* textShadow 设置文本的阴影效果
* textTransform 对文本设置大写效果
* unicodeBidi
* whiteSpace 设置如何设置文本中的折行和空白符
* wordSpacing 设置文本中的词间距

**Standard 属性**

属性描述：

* dir 设置或返回文本的方向
* lang 设置或返回元素的语言代码
* title 设置或返回元素的咨询性的标题

**其他属性**

* cssText 属性 它是一组样式属性及其值的文本表示。这个文本格式化为一个 CSS 样式表，去掉了包围属性和值的元素选择器的花括号。将这一属性设置为非法的值将会抛出一个代码为 SYNTAX\_ERR 的 DOMException 异常。当 CSS2Properties 对象是只读的时候，试图设置这一属性将会抛出一个代码为 NO\_MODIFICATION\_ALLOWED\_ERR 的 DOMException 异常。

**Table 对象**

Table 对象代表一个 HTML 表格。在 HTML 文档中 <table> 标签每出现一次，一个 Table 对象就会被创建。

**Table 对象的集合**

集合描述：

* cells[] 返回包含表格中所有单元格的一个数组。
* rows[] 返回包含表格中所有行的一个数组。
* tBodies[] 返回包含表格中所有 tbody 的一个数组。

**Table 对象的属性**

属性描述：

* align 表在文档中的水平对齐方式。（已废弃）
* bgColor 表在文档中的水平对齐方式。（已废弃）
* border 设置或返回表格边框的宽度。
* caption 对表格的 <caption> 元素的引用。
* cellPadding 设置或返回单元格内容和单元格边框之间的空白量。
* cellSpacing 设置或返回在表格中的单元格之间的空白量。
* frame 设置或返回表格的外部边框。
* id 设置或返回表格的 id。
* rules 设置或返回表格的内部边框（行线）。
* summary 设置或返回对表格的描述（概述）。
* tFoot 返回表格的 TFoot 对象。如果不存在该元素，则为 null。
* tHead 返回表格的 THead 对象。如果不存在该元素，则为 null。
* width 设置或返回表格的宽度。

**Table 对象的方法**

方法描述：

* createCaption() 为表格创建一个 caption 元素。
* createTFoot() 在表格中创建一个空的 tFoot 元素。
* createTHead() 在表格中创建一个空的 tHead 元素。
* deleteCaption() 从表格删除 caption 元素以及其内容。
* deleteRow() 从表格删除一行。
* deleteTFoot() 从表格删除 tFoot 元素及其内容。
* deleteTHead() 从表格删除 tHead 元素及其内容。
* insertRow() 在表格中插入一个新行。

方法说明：

1、createCaption() ：在表格中获取或创建 <caption> 元素。

语法：

tableObject.createCaption()

返回值：

返回一个 HTMLElement 对象，表示该表的 <caption> 元素。如果该表格已经有了标题，则返回它。如果该表没有 <caption> 元素，则创建一个新的空 <caption> 元素，把它插入表格，并返回它。

实例：下面的例子为表格创建了一个标题：

function createCaption(){

var x=document.getElementById('myTable').createCaption()

x.innerHTML="My table caption"

}

2、createTFoot() ：在表格中获取或创建 <tfoot> 元素。

语法：

tableObject.createTFoot()

返回值：

返回一个 TableSection，表示该表的 <tfoot> 元素。如果该表格已经有了脚注，则返回它。如果该表没有脚注，则创建一个新的空 <tfoot> 元素，把它插入表格，并返回它。

实例：下面的例子为表格创建了一个脚注：

function createCaption(){

var x=document.getElementById('myTable').createTFoot()

x.innerHTML="My table foot"

}

3、createTHead() 方法用于在表格中获取或创建 <thead> 元素。

语法：

tableObject.createTHead()

返回值：

返回一个 TableSection，表示该表的 <thead> 元素。如果该表格已经有了表头，则返回它。如果该表没有表头，则创建一个新的空 <thead> 元素，把它插入表格，并返回它。

实例：下面的例子为表格创建了一个脚注：

function createCaption(){

var x=document.getElementById('myTable').createTHead()

x.innerHTML="My table head"

}

4、deleteCaption() ：删除表格的 caption 元素及其内容。

语法：

tableObject.deleteCaption()

说明：如果该表有 <caption> 元素，则从文档树种删除它。否则，什么也不做。

实例：下面的例子删除表格的标题：

function deleteCaption() {

document.getElementById('myTable').deleteCaption()

}

5、deleteRow() ：从表格删除指定位置的行。

语法：

tableObject.deleteRow(index)

参数说明：

参数 index 指定了要删除的行在表中的位置。行的编码顺序就是他们在文档源代码中出现的顺序。<thead> 和 <tfoot> 中的行与表中其它行一起编码。

实例：

function delRow() {

document.getElementById('myTable').deleteRow(0)

}

6、insertRow()：在表格中的指定位置插入一个新行。

语法：

tableObject.insertRow(index)

返回值：返回一个 TableRow，表示新插入的行。

说明：

该方法创建一个新的 TableRow 对象，表示一个新的 <tr> 标记，并把它插入表中的指定位置。新行将被插入 index 所在行之前。若 index 等于表中的行数，则新行将被附加到表的末尾。如果表是空的，则新行将被插入到一个新的 <tbody> 段，该段自身会被插入表中。

抛出：

若参数 index 小于 0 或大于等于表中的行数，该方法将抛出代码为 INDEX\_SIZE\_ERR 的 DOMException 异常。

实例：下面的例子在表格的开头插入一个新行：

function insRow(){

document.getElementById('myTable').insertRow(0)

}

**TableCell 对象**

TableCell 对象代表一个 HTML 表格单元格。在一个 HTML 文档中 <td> 标签每出现一次，一个 TableCell 对象就会被创建。

**TableCell 对象的属性**

属性描述：

* abbr 设置或返回单元格中内容的缩写版本。
* align 设置或返回单元格内部数据的水平排列方式。
* axis 设置或返回相关单元格的一个逗号分隔的列表。
* cellIndex 返回单元格在某行的单元格集合中的位置。
* ch 设置或返回单元格的对齐字符。
* chOff 设置或返回单元格的对齐字符的偏移量。
* colSpan 单元格横跨的列数。
* headers 设置或返回 header-cell 的 id 值。
* id 设置或返回单元格的 id。
* innerHTML 设置或返回单元格的开始标签和结束标签之间的 HTML。
* rowSpan 设置或返回单元格可横跨的行数。
* scope 设置或返回此单元格是否可提供标签信息。
* vAlign 设置或返回表格单元格内数据的垂直排列方式。
* width 设置或返回单元格的宽度。

**TableRow 对象**

TableRow 对象代表一个 HTML 表格行。在 HTML 文档中 <tr> 标签每出现一次，一个 TableRow 对象就会被创建。

**TableRow 对象的集合**

集合描述：

* cells[] 返回包含行中所有单元格的一个数组。

**TableRow 对象的属性**

属性描述：

* align 设置或返回在行中数据的水平排列。
* ch 设置或返回在行中单元格的对齐字符。
* chOff 设置或返回在行中单元格的对齐字符的偏移量。
* id 设置或返回行的 id。
* innerHTML 设置或返回行的开始标签和结束标签之间的 HTML。
* rowIndex 返回该行在表中的位置。
* sectionRowIndex 返回在 tBody 、tHead 或 tFoot 中，行的位置。
* vAlign 设置或返回在行中的数据的垂直排列方式。

**TableRow 对象的方法**

方法描述：

* deleteCell() 删除行中的指定的单元格。
* insertCell() 在一行中的指定位置插入一个空的 <td> 元素。

方法说明：

1、deleteCell() ：删除表格行中指定位置的单元格（<td> 元素）。

语法：

tablerowObject.deleteCell(index)

参数说明：

参数 index 是要删除的表元在行中的位置。

抛出：

若参数 index 小于 0 或大于等于行中的的表元数，该方法将抛出代码为 INDEX\_SIZE\_ERR 的 DOMException 异常。

实例：下面的例子从行中删除了一个单元格：

function delCell() {

document.getElementById('tr2').deleteCell(0)

}

2、insertCell() 方法用于在 HTML 表的一行的指定位置插入一个空的 <td> 元素。

语法：

tablerowObject.insertCell(index)

返回值：一个 TableCell 对象，表示新创建并被插入的 <td> 元素。

说明：

该方法将创建一个新的 <td> 元素，把它插入行中指定的位置。新单元格将被插入当前位于 index 指定位置的表元之前。如果 index 等于行中的单元格数，则新单元格被附加在行的末尾。请注意，该方法只能插入 <td> 数据表元。若需要给行添加头表元，必须用 Document.createElement() 方法和 Node.insertBefore() 方法（或相关的方法）创建并插入一个 <th> 元素。

抛出：

若参数 index 小于 0 或大于等于行中的的表元数，该方法将抛出代码为 INDEX\_SIZE\_ERR 的 DOMException 异常。

实例：下面的例子在表格行中插入了一个单元格：

function insCell() {

var x=document.getElementById('tr2').insertCell(0)

x.innerHTML="John"

}

**Text 对象**

Text 对象代表 HTML 表单中的文本输入域。在 HTML 表单中 <input type="text"> 每出现一次，Text 对象就会被创建。该元素可创建一个单行的文本输入字段。当用户编辑显示的文本并随后把输入焦点转移到其他元素的时候，会触发 onchange 事件句柄。

**Text 对象的属性**

属性描述：

* accessKey 设置或返回访问文本域的快捷键。
* alt 设置或返回当浏览器不支持文本域时供显示的替代文本。
* defaultValue 设置或返回文本域的默认值。
* disabled 设置或返回文本域是否应被禁用。
* form 返回一个对包含文本域的表单对象的引用。
* id 设置或返回文本域的 id。
* maxLength 设置或返回文本域中的最大字符数。
* name 设置或返回文本域的名称。
* readOnly 设置或返回文本域是否应是只读的。
* size 设置或返回文本域的尺寸。
* tabIndex 设置或返回文本域的 tab 键控制次序。
* type 返回文本域的表单元素类型。
* value 设置或返回文本域的 value 属性的值。

**Text 对象的方法**

方法描述：

* blur() 从文本域上移开焦点。
* focus() 在文本域上设置焦点。
* select() 选取文本域中的内容。

**Textarea 对象**

Textarea 对象代表 HTML 表单中的一个文本区 (text-area)。在表单中 <textarea> 标签每出现一次，一个 Textarea 对象就会被创建。

**Textarea 对象的属性**

属性描述：

* accessKey 设置或返回访问 textarea 的键盘快捷键。
* cols 设置或返回 textarea 的宽度。
* defaultValue 设置或返回文本框中的初始内容。
* disabled 设置或返回 textarea 是否应当被禁用。
* form 返回对包含该 textarea 的表单对象的引用。
* id 设置或返回某个 textarea 的 id
* name 设置或返回 textarea 的名称。
* readOnly 设置或返回 textarea 是否应当是只读的。
* rows 设置或返回 textarea 的高度。
* tabIndex 设置或返回 textarea 的 tab 键控制次序。
* type 返回该文本框的表单类型。
* value 设置或返回在 textarea 中的文本。

**Textarea 对象的方法**

方法描述：

* blur() 从 textarea 移开焦点。
* focus() 在 textarea 上设置焦点。
* select() 选择 textarea 中的文本。

**Window 对象**

Window 对象是 JavaScript 层级中的顶层对象。Window 对象代表一个浏览器窗口或一个框架。Window 对象会在 <body> 或 <frameset> 每次出现时被自动创建。

**Window 对象的集合**

集合描述：

* frames[] 返回窗口中所有命名的框架。该集合是 Window 对象的数组，每个 Window 对象在窗口中含有一个框架或 <iframe>。属性 frames.length 存放数组 frames[] 中含有的元素个数。注意，frames[] 数组中引用的框架可能还包括框架，它们自己也具有 frames[] 数组。

**Window 对象的属性**

属性描述：

* closed 返回窗口是否已被关闭。
* defaultStatus 设置或返回窗口状态栏中的默认文本。
* document 对 Document 对象的只读引用。请参阅 Document 对象。
* history 对 History 对象的只读引用。请参数 History 对象。
* innerheight 返回窗口的文档显示区的高度。
* innerwidth 返回窗口的文档显示区的宽度。
* length 设置或返回窗口中的框架数量。
* location 用于窗口或框架的 Location 对象。请参阅 Location 对象。
* name 设置或返回窗口的名称。
* Navigator 对 Navigator 对象的只读引用。请参考 Navigator 对象。
* opener 返回对创建此窗口的窗口的引用。
* outerheight 返回窗口的外部高度。
* outerwidth 返回窗口的外部宽度。
* pageXOffset 设置或返回当前页面相对于窗口显示区左上角的 X 位置。
* pageYOffset 设置或返回当前页面相对于窗口显示区左上角的 Y 位置。
* parent 返回父窗口。
* Screen 对 Screen 对象的只读引用。请参考 Screen 对象。
* self 返回对当前窗口的引用。等价于 Window 属性。
* status 设置窗口状态栏的文本。
* top 返回最顶层的先辈窗口。
* window window 属性等价于 self 属性，它包含了对窗口自身的引用。
* screenLeft
* screenTop
* screenX
* screenY

只读整数。声明了窗口的左上角在屏幕上的的 x 坐标和 y 坐标。IE、Safari 和 Opera 支持 screenLeft 和 screenTop，而 Firefox 和 Safari 支持 screenX 和 screenY。

属性说明：

1、opener：是一个可读可写的属性，可返回对创建该窗口的 Window 对象的引用。opener 属性非常有用，创建的窗口可以引用创建它的窗口所定义的属性和函数。

语法：

window.opener

说明：

只有表示顶层窗口的 Window 对象的 operner 属性才有效，表示框架的 Window 对象的 operner 属性无效。

实例：下面的例子可向 opener 窗口写文本（父窗口）：

<script type="text/javascript">

myWindow=window.open('','MyName','width=200,height=100')

myWindow.document.write("This is 'myWindow'")

myWindow.focus()

myWindow.opener.document.write("This is the parent window")

</script>

2、name ：设置或返回存放窗口的名称的一个字符串。该名称是在 open() 方法创建窗口时指定的或者使用一个 <frame> 标记的 name 属性指定的。窗口的名称可以用作一个 <a> 或者 <form> 标记的 target 属性的值。以这种方式使用 target 属性声明了超链接文档或表单提交结果应该显示于指定的窗口或框架中。

语法：window.name=name

**Window 对象的方法**

方法描述：

* alert() 显示带有一段消息和一个确认按钮的警告框。
* blur() 把键盘焦点从顶层窗口移开。
* clearInterval() 取消由 setInterval() 设置的 timeout。
* clearTimeout() 取消由 setTimeout() 方法设置的 timeout。
* close() 关闭浏览器窗口。
* confirm() 显示带有一段消息以及确认按钮和取消按钮的对话框。
* createPopup() 创建一个 pop-up 窗口。
* focus() 把键盘焦点给予一个窗口。
* moveBy() 可相对窗口的当前坐标把它移动指定的像素。
* moveTo() 把窗口的左上角移动到一个指定的坐标。
* open() 打开一个新的浏览器窗口或查找一个已命名的窗口。
* print() 打印当前窗口的内容。
* prompt() 显示可提示用户输入的对话框。
* resizeBy() 按照指定的像素调整窗口的大小。
* resizeTo() 把窗口的大小调整到指定的宽度和高度。
* scrollBy() 按照指定的像素值来滚动内容。
* scrollTo() 把内容滚动到指定的坐标。
* setInterval() 按照指定的周期（以毫秒计）来调用函数或计算表达式。
* setTimeout() 在指定的毫秒数后调用函数或计算表达式。

方法说明：

1、open() ：打开一个新的浏览器窗口或查找一个已命名的窗口。

语法：

window.open(URL,name,features,replace)

参数描述：

* URL 一个可选的字符串，声明了要在新窗口中显示的文档的 URL。如果省略了这个参数，或者它的值是空字符串，那么新窗口就不会显示任何文档。
* name 一个可选的字符串，该字符串是一个由逗号分隔的特征列表，其中包括数字、字母和下划线，该字符声明了新窗口的名称。这个名称可以用作标记 <a> 和 <form> 的属性 target 的值。如果该参数指定了一个已经存在的窗口，那么 open() 方法就不再创建一个新窗口，而只是返回对指定窗口的引用。在这种情况下，features 将被忽略。
* features 一个可选的字符串，声明了新窗口要显示的标准浏览器的特征。如果省略该参数，新窗口将具有所有标准特征。在窗口特征这个表格中，我们对该字符串的格式进行了详细的说明。
* replace 一个可选的布尔值。规定了装载到窗口的 URL 是在窗口的浏览历史中创建一个新条目，还是替换浏览历史中的当前条目。值：true - URL 替换浏览历史中的当前条目。false - URL 在浏览历史中创建新的条目。

窗口特征：（Window Features）

* channelmode=yes|no|1|0 是否使用剧院模式显示窗口。默认为 no。
* directories=yes|no|1|0 是否添加目录按钮。默认为 yes。
* fullscreen=yes|no|1|0 是否使用全屏模式显示浏览器。默认是 no。处于全屏模式的窗口必须同时处于剧院模式。
* height=pixels 窗口文档显示区的高度。以像素计。
* left=pixels 窗口的 x 坐标。以像素计。
* location=yes|no|1|0 是否显示地址字段。默认是 yes。
* menubar=yes|no|1|0 是否显示菜单栏。默认是 yes。
* resizable=yes|no|1|0 窗口是否可调节尺寸。默认是 yes。
* scrollbars=yes|no|1|0 是否显示滚动条。默认是 yes。
* status=yes|no|1|0 是否添加状态栏。默认是 yes。
* titlebar=yes|no|1|0 是否显示标题栏。默认是 yes。
* toolbar=yes|no|1|0 是否显示浏览器的工具栏。默认是 yes。
* top=pixels 窗口的 y 坐标。
* width=pixels 窗口的文档显示区的宽度。以像素计。

重要事项：

请不要混淆方法 Window.open() 与方法 Document.open()，这两者的功能完全不同。为了使您的代码清楚明白，请使用 Window.open()，而不要使用 open()。

实例1：下面的例子讲在新浏览器窗口中打开 www.w3school.com.cn ：

function open\_win(){

window.open("http://www.w3school.com.cn")

}

实例2：下面的例子讲在新浏览器窗口中打开 about:blank 页：

<script type="text/javascript">

myWindow=window.open('','','width=200,height=100')

myWindow.document.write("This is 'myWindow'")

myWindow.focus()

</script>

2、close()：关闭浏览器窗口。

语法：

window.close()

说明：

方法 close() 将关闭有 window 指定的顶层浏览器窗口。某个窗口可以通过调用 self.close() 或只调用 close() 来关闭其自身。只有通过 JavaScript 代码打开的窗口才能够由 JavaScript 代码关闭。这阻止了恶意的脚本终止用户的浏览器。

**Window 对象的描述**

Window 对象表示一个浏览器窗口或一个框架。在客户端 JavaScript 中，Window 对象是全局对象，所有的表达式都在当前的环境中计算。也就是说，要引用当前窗口根本不需要特殊的语法：，可以把那个窗口的属性作为全局变量来使用。例如，可以只写 document，而不必写 window.document。同样，可以把当前窗口对象的方法当作函数来使用，如只写 alert()，而不必写 Window.alert()。

除了上面列出的属性和方法，Windos 对象还实现了核心 JavaScript 所定义的所有全局属性和方法。

Window 对象的 window 属性和 self 属性引用的都是它自己。当你想明确地引用当前窗口，而不仅仅是隐式地引用它时，可以使用这两个属性。除了这两个属性之外，parent 属性、top 属性以及 frame[] 数组都引用了与当前 Window 对象相关的其他 Window 对象。

要引用窗口中的一个框架，可以使用如下语法：：

frame[i] //当前窗口的框架

self.frame[i] //当前窗口的框架

w.frame[i] //窗口 w 的框架

要引用一个框架的父窗口（或父框架），可以使用下面的语法：：

parent //当前窗口的父窗口

self.parent //当前窗口的父窗口

w.parent //窗口 w 的父窗口

要从顶层窗口含有的任何一个框架中引用它，可以使用如下语法：：

top //当前框架的顶层窗口

self.top //当前框架的顶层窗口

f.top //框架 f 的顶层窗口

新的顶层浏览器窗口由方法 Window.open() 创建。当调用该方法时，应把 open() 调用的返回值存储在一个变量中，然后使用那个变量来引用新窗口。新窗口的 opener 属性反过来引用了打开它的那个窗口。

一般来说，Window 对象的方法都是对浏览器窗口或框架进行某种操作。而 alert() 方法、confirm() 方法和 prompt 方法则不同，它们通过简单的对话框与用户进行交互。