<script type="text/javascript">

document.write("开启JS之旅!");

</script>

<script src="script.js"></script>

定义变量：

<script type="text/javascript">

var mynum=8;

</script>

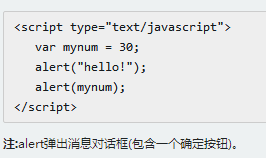
定义函数，和使用函数



输出内容



警告（alert 消息对话框）



确认（confirm 消息对话框）



提问（prompt 消息对话框）



打开新窗口（window.open）



关闭窗口（window.close）



认识DOM

文档对象模型DOM（Document Object Model）定义访问和处理HTML文档的标准方法。DOM 将HTML文档呈现为带有元素、属性和文本的树结构（节点树）。

通过ID获取元素

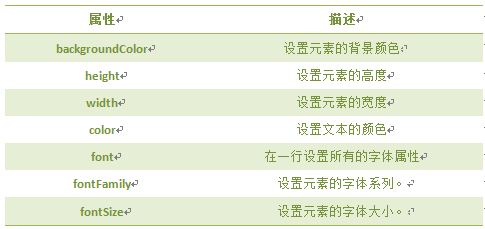
document. ElementById(“id”)

innerHTML 属性

innerHTML 属性用于获取或替换 HTML 元素的内容。



改变 HTML 样式

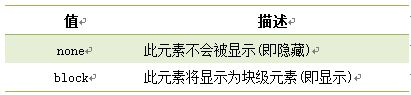


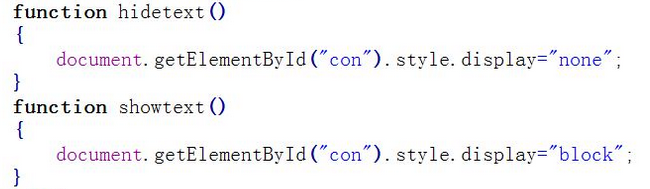
HTML DOM 允许 JavaScript 改变 HTML 元素的样式。如何改变 HTML 元素的样式呢？



显示和隐藏（display属性）

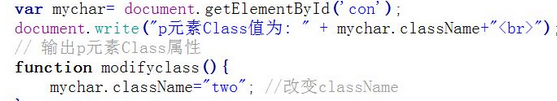
网页中经常会看到显示和隐藏的效果，可通过display属性来设置。





控制类名（className 属性）

className 属性设置或返回元素的class 属性。



主要事件表:



鼠标单击事件( onclick ）

<input name="button" type="button" value="点击提交" onclick="add2()" />

鼠标经过事件（onmouseover）



鼠标移开事件（onmouseout）



光标聚焦事件（onfocus）



失焦事件（onblur）



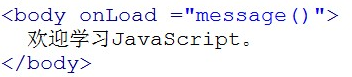
内容选中事件（onselect）



文本框内容改变事件（onchange）



加载事件（onload）



卸载事件（onunload）



对象

Date 日期对象

ng 字符串对象

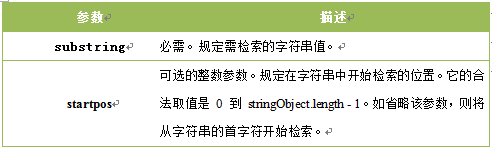


返回指定位置的字符

stringObject.charAt(index)

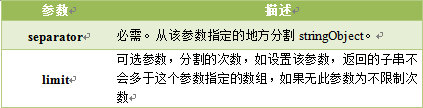
返回指定的字符串首次出现的位置

stringObject.indexOf(substring, startpos)



字符串分割split()

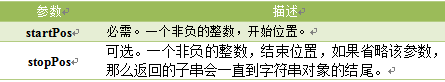
stringObject.split(separator,limit)





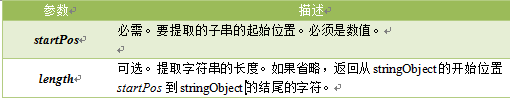
提取字符串substring()

stringObject.substring(starPos,stopPos)



提取指定数目的字符substr()

stringObject.substr(startPos,length)



注意：如果参数startPos是负数，从字符串的尾部开始算起的位置。也就是说，-1 指字符串中最后一个字符，-2 指倒数第二个字符，以此类推。

如果startPos为负数且绝对值大于字符串长度，startPos为0。

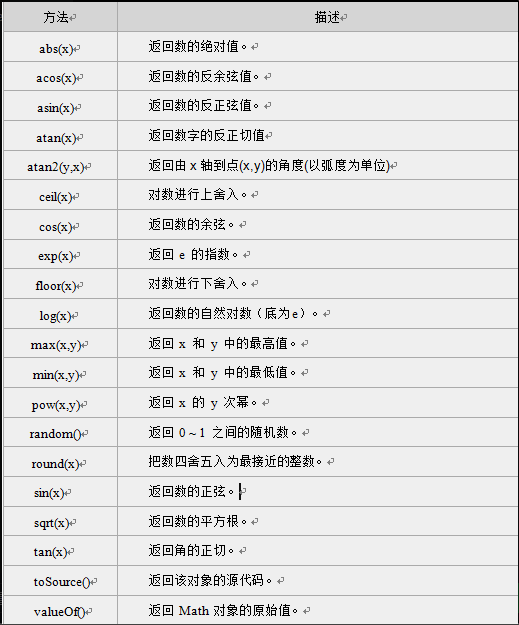
Math对象



Math 对象属性



Math 对象方法



随机数 random()

random() 方法可返回介于 0 ~ 1（大于或等于 0 但小于 1 )之间的一个随机数。>=0&&<1

Array 数组对象

数组方法：



**join()函数不会改变原来数组，会产生一个新的数组**

**颠倒数组元素顺序reverse()**

**reverse() 方法用于颠倒数组中元素的顺序。**

**语法：**

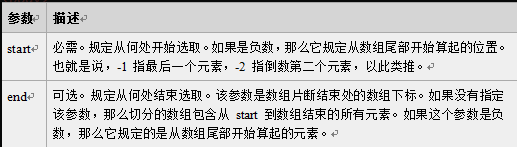
**arrayObject.reverse()**

**注意：该方法会改变原来的数组，而不会创建新的数组。**

**选定元素slice()**

**slice() 方法可从已有的数组中返回选定的元素。**

**arrayObject.slice(start,end)**



**1.返回一个新的数组，包含从 start 到 end （不包括该元素）的 arrayObject 中的元素。**

**2. 该方法并不会修改数组，而是返回一个子数组。**

**注意：**

**1. 可使用负值从数组的尾部选取元素。**

**2.如果 end 未被规定，那么 slice() 方法会选取从 start 到数组结尾的所有元素。**

**3. String.slice() 与 Array.slice() 相似。**

**数组排序sort()**

**arrayObject.sort(方法函数)**

**1.如果不指定<方法函数>，则按unicode码顺序排列。**

**2.如果指定<方法函数>，则按<方法函数>所指定的排序方法排序。**

**myArray.sort(sortMethod);**

**注意: 该函数要比较两个值，然后返回一个用于说明这两个值的相对顺序的数字。比较函数应该具有两个参数 a 和 b，其返回值如下：**

**若返回值<=-1，则表示 A 在排序后的序列中出现在 B 之前。**

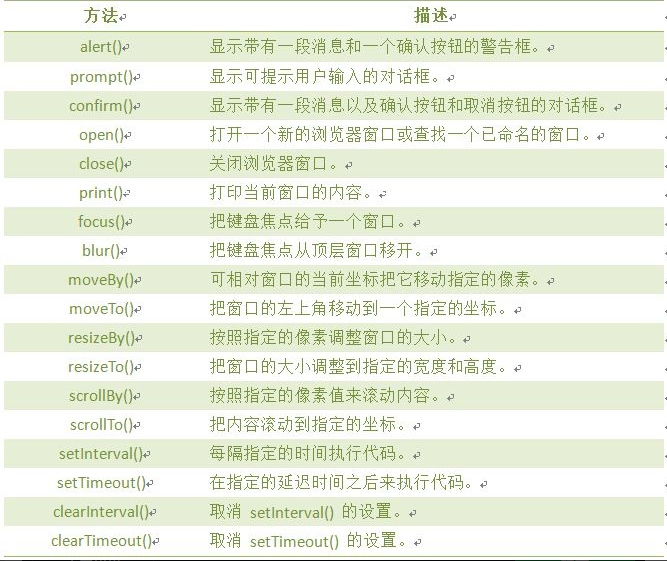
**若返回值>-1 && <1，则表示 A 和 B 具有相同的排序顺序。**

**若返回值>=1，则表示 A 在排序后的序列中出现在 B 之后。**

**window对象**

**window对象是BOM的核心，window对象指当前的浏览器窗口。**

**window对象方法:**



**JavaScript 计时器**

**在JavaScript中，我们可以在设定的时间间隔之后来执行代码，而不是在函数被调用后立即执行。**

**计时器类型：**

**一次性计时器：仅在指定的延迟时间之后触发一次。**

**间隔性触发计时器：每隔一定的时间间隔就触发一次。**



**计时器setInterval()**

**在执行时,从载入页面后每隔指定的时间执行代码。**

**setInterval(代码,交互时间);**

参数说明：

1. 代码：要调用的函数或要执行的代码串。

2. 交互时间：周期性执行或调用表达式之间的时间间隔，以毫秒计（1s=1000ms）。

返回值:

一个可以传递给 clearInterval() 从而取消对"代码"的周期性执行的值。

History 对象

history对象记录了用户曾经浏览过的页面(URL)，并可以实现浏览器前进与后退相似导航的功能。

注意:从窗口被打开的那一刻开始记录，每个浏览器窗口、每个标签页乃至每个框架，都有自己的history对象与特定的window对象关联。

语法：

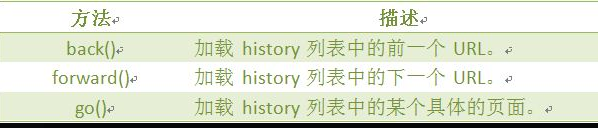
window.history.[属性|方法]

注意：window可以省略。

History 对象属性



History 对象方法



返回前一个浏览的页面



返回浏览历史中的其他页面



Location对象

location.[属性|方法]

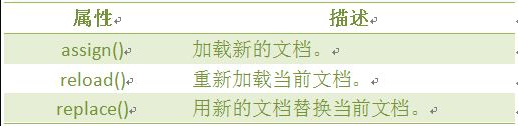
location对象属性图示:



location 对象属性：

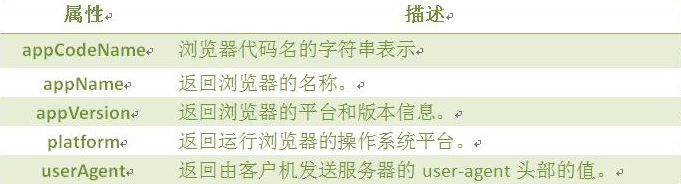


location 对象方法:



Navigator对象

Navigator 对象包含有关浏览器的信息，通常用于检测浏览器与操作系统的版本。



<script type="text/javascript">

var browser=navigator.appName;

var b\_version=navigator.appVersion;

document.write("Browser name"+browser);

document.write("<br>");

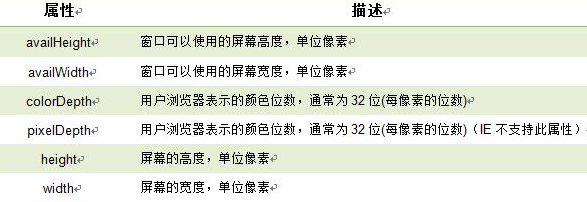
document.write("Browser version"+b\_version);

</script>

screen对象

screen对象用于获取用户的屏幕信息。

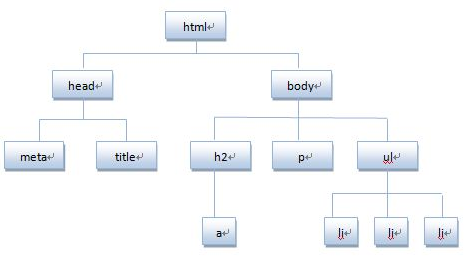
window.screen.属性



认识DOM

文档对象模型DOM（Document Object Model）定义访问和处理HTML文档的标准方法。DOM 将HTML文档呈现为带有元素、属性和文本的树结构（节点树）。

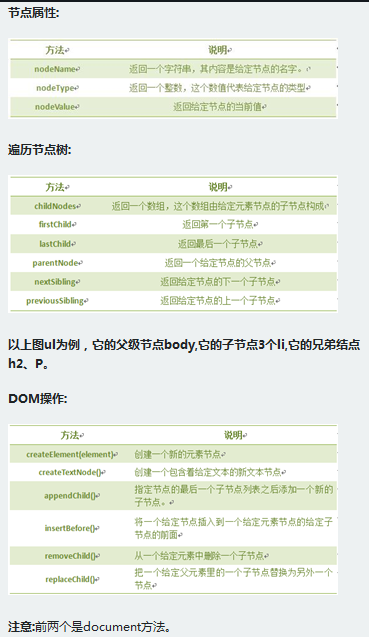
将HTML代码分解为DOM节点层次图:



1. 元素节点：上图中<html>、<body>、<p>等都是元素节点，即标签。

2. 文本节点:向用户展示的内容，如<li>...</li>中的JavaScript、DOM、CSS等文本。

3. 属性节点:元素属性，如<a>标签的链接属性href="http://www.imooc.com"。



getElementsByName()方法

返回带有指定名称的节点对象的集合。

document.getElementsByName(name)

与getElementById() 方法不同的是，通过元素的 name 属性查询元素，而不是通过 id 属性。

注意:

1. 因为文档中的 name 属性可能不唯一，所有 getElementsByName() 方法返回的是元素的数组，而不是一个元素。

2. 和数组类似也有length属性，可以和访问数组一样的方法来访问，从0开始。

getElementsByTagName()方法

返回带有指定标签名的节点对象的集合。返回元素的顺序是它们在文档中的顺序。

getElementsByTagName(Tagname)

Tagname是标签的名称，如p、a、img等标签名。

区别getElementByID,getElementsByName,getElementsByTagName

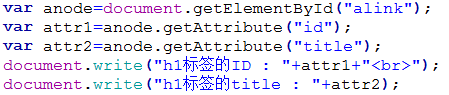
以人来举例说明，人有能标识身份的身份证，有姓名，有类别(大人、小孩、老人)等。

getAttribute()方法

通过元素节点的属性名称获取属性的值。

elementNode.getAttribute(name)

elementNode：使用getElementById()、getElementsByTagName()等方法，获取到的元素节点。



setAttribute()方法

setAttribute() 方法增加一个指定名称和值的新属性，或者把一个现有的属性设定为指定的值。

elementNode.setAttribute(name,value)

1. 把指定的属性设置为指定的值。如果不存在具有指定名称的属性，该方法将创建一个新属性。

节点属性

在文档对象模型 (DOM) 中，每个节点都是一个对象。DOM 节点有三个重要的属性 ：

1. nodeName : 节点的名称

2. nodeValue ：节点的值

3. nodeType ：节点的类型

一、nodeName 属性: 节点的名称，是只读的。

1. 元素节点的 nodeName 与标签名相同

2. 属性节点的 nodeName 是属性的名称

3. 文本节点的 nodeName 永远是 #text

4. 文档节点的 nodeName 永远是 #document

二、nodeValue 属性：节点的值

1. 元素节点的 nodeValue 是 undefined 或 null

2. 文本节点的 nodeValue 是文本自身

3. 属性节点的 nodeValue 是属性的值

三、nodeType 属性: 节点的类型，是只读的。以下常用的几种结点类型:

元素类型 节点类型

元素 1

属性 2

文本 3

注释 8

文档 9

访问子结点childNodes

访问选定元素节点下的所有子节点的列表，返回的值可以看作是一个数组，他具有length属性。

elementNode.childNodes

如果选定的节点没有子节点，则该属性返回不包含节点的 NodeList。



访问子结点的第一和最后项 elementNode.firstNode elementNode.lastNode

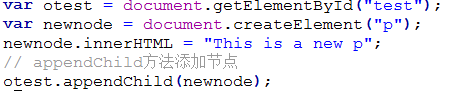
访问父节点parentNode elementNode.parentNode

访问兄弟节点 elementNode. nextSibling

插入节点appendChild()

在指定节点的最后一个子节点列表之后添加一个新的子节点。

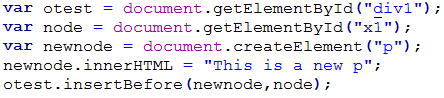
appendChild(newnode)



插入节点insertBefore()

insertBefore() 方法可在已有的子节点前插入一个新的子节点。

insertBefore(newnode,node);



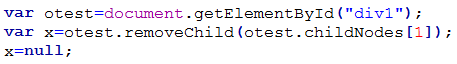
删除节点removeChild()

removeChild() 方法从子节点列表中删除某个节点。如删除成功，此方法可返回被删除的节点，如失败，则返回 NULL。

nodeObject.removeChild(node)

注意: 把删除的子节点赋值给 x，这个子节点不在DOM树中，但是还存在内存中，可通过 x 操作。

如果要完全删除对象，给 x 赋 null 值，代码如下:



替换元素节点replaceChild()

replaceChild 实现子节点(对象)的替换。返回被替换对象的引用。

node.replaceChild (newnode,oldnew )

注意:

1. 当 oldnode 被替换时，所有与之相关的属性内容都将被移除。

2. newnode 必须先被建立。

创建元素节点createElement

createElement()方法可创建元素节点。此方法可返回一个 Element 对象。

document.createElement(tagName)

tagName：字符串值，这个字符串用来指明创建元素的类型。

创建文本节点createTextNode

createTextNode() 方法创建新的文本节点，返回新创建的 Text 节点。

document.createTextNode(data)

<script type="text/javascript">

var element = document.createElement("p");

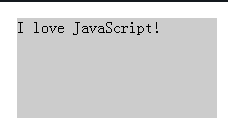
element.className = "message";

var textNode = document.createTextNode("I love JavaScript!");

element.appendChild(textNode);

document.body.appendChild(element);

</script>



浏览器窗口可视区域大小

获得浏览器窗口的尺寸（浏览器的视口，不包括工具栏和滚动条）的方法:

一、对于IE9+、Chrome、Firefox、Opera 以及 Safari：

• window.innerHeight - 浏览器窗口的内部高度

• window.innerWidth - 浏览器窗口的内部宽度

二、对于 Internet Explorer 8、7、6、5：

• document.documentElement.clientHeight表示HTML文档所在窗口的当前高度。

• document.documentElement.clientWidth表示HTML文档所在窗口的当前宽度。

或者

Document对象的body属性对应HTML文档的<body>标签

• document.body.clientHeight

• document.body.clientWidth

在不同浏览器都实用的 JavaScript 方案：

var w= document.documentElement.clientWidth

|| document.body.clientWidth;

var h= document.documentElement.clientHeight

|| document.body.clientHeight;

网页尺寸scrollHeight

scrollHeight和scrollWidth，获取网页内容高度和宽度。

一、针对IE、Opera:

scrollHeight 是网页内容实际高度，可以小于 clientHeight。

二、针对NS、FF:

scrollHeight 是网页内容高度，不过最小值是 clientHeight。也就是说网页内容实际高度小于 clientHeight 时，scrollHeight 返回 clientHeight 。

三、浏览器兼容性

var w=document.documentElement.scrollWidth

|| document.body.scrollWidth;

var h=document.documentElement.scrollHeight

|| document.body.scrollHeight;

注意:区分大小写

scrollHeight和scrollWidth还可获取Dom元素中内容实际占用的高度和宽度。

网页尺寸offsetHeight

offsetHeight和offsetWidth，获取网页内容高度和宽度(包括滚动条等边线，会随窗口的显示大小改变)。

一、值

offsetHeight = clientHeight + 滚动条 + 边框。

二、浏览器兼容性

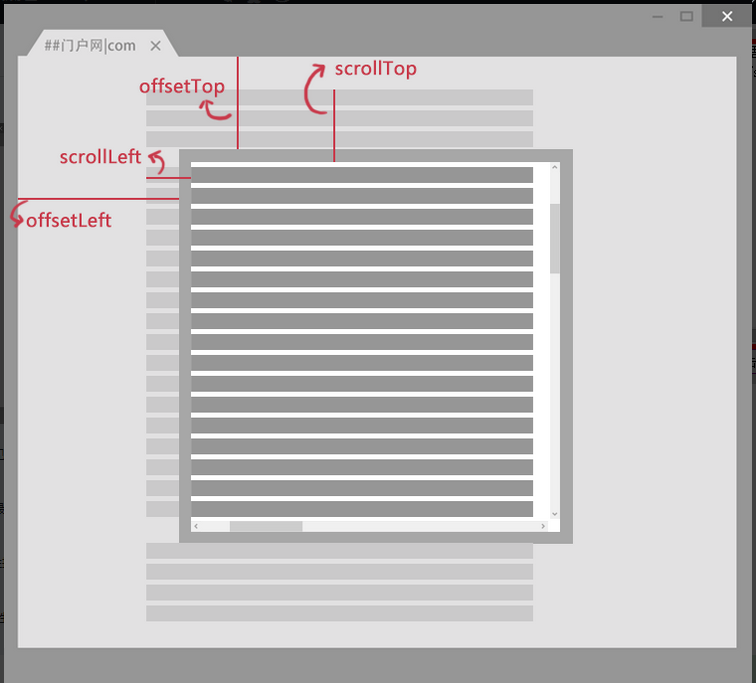
var w= document.documentElement.offsetWidth

|| document.body.offsetWidth;

var h= document.documentElement.offsetHeight

|| document.body.offsetHeight;

网页卷去的距离与偏移量



scrollLeft:设置或获取位于给定对象左边界与窗口中目前可见内容的最左端之间的距离 ，即左边灰色的内容。

scrollTop:设置或获取位于对象最顶端与窗口中可见内容的最顶端之间的距离 ，即上边灰色的内容。

offsetLeft:获取指定对象相对于版面或由 offsetParent 属性指定的父坐标的计算左侧位置 。

offsetTop:获取指定对象相对于版面或由 offsetParent 属性指定的父坐标的计算顶端位置 。

注意:

1. 区分大小写

2. offsetParent：布局中设置postion属性(Relative、Absolute、fixed)的父容器，从最近的父节点开始，一层层向上找，直到HTML的body。

练习题：

<tr>

<td>xh001</td>

<td>王小明</td>

<td><a href="javascript:;" onclick="rem(this)">删除</a></td> <!--在删除按钮上添加点击事件 -->

</tr>

其中onclick的rem()函数中用到了this指针，传入的是当前标签的对象地址