# DOM 与 BOM

## 1 事件

JS是以事件驱动为核心的一门语言。

事件的三要素是：事件源、事件、事件驱动程序。

三句话：获取事件源、绑定事件、书写事件驱动程序。

获取事件源：document.getElementById(“box”);

绑定事件： box.onclick = function(){ 程序 };

书写事件驱动程序：以后要学习的关于DOM的操作。

常见事件：



## 2 DOM

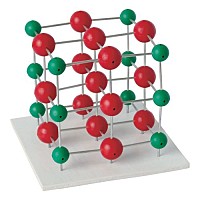
### 2.1 解析过程

HTML加载完毕，渲染引擎会在内存中把HTML文档，生成一个DOM树，getElementById是获取内中DOM上的元素节点。然后操作的时候修改的是该元素的属性。

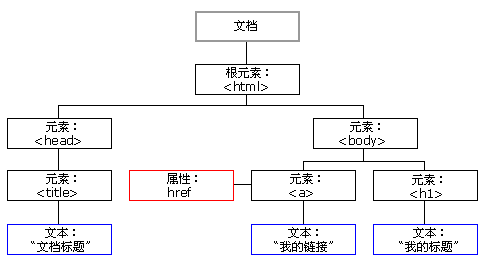
### 2.2 DOM (文档对象模型)

document是文档对象模型的一部分。

DOM是一个复合的数据类型。



### 2.3 DOM的数据结构（树状）



### 2.4 节点

HTML的组成部分为节点（ Node ），在HTML当中一切都是节点……

由结构图中我们可以看到，整个文档就是一个文档节点。

每一个HMTL标签都是一个元素节点（标签）。

标签中的文字则是文字节点。（文本）

标签的属性是属性节点。（属性）

### 2.5 DOM节点的获得

操作节点，必须首先找到该元素。有三种方法来做这件事：

通过 id 找到 HTML 元素 document.getElementById("demo");

通过标签名找到 HTML 元素 document.getElementsByTagName("div");

通过类名找到 HTML 元素 document.getElementsByClassName("a");

注意：通过类名查找 HTML 元素在 IE 5,6,7,8 中无效

标签数组= document.getElementsByTagName("div");

标签数组= document.getElementsByClassName("a");

他们两个的返回值是标签数组，习惯性是遍历之后再使用。

特殊情况：数组中的值只有1个。

document.getElementsByTagName("div")[0];取数组中第一个元素

document.getElementsByClassName("a")[0];取数组中第一个元素

### 2.6 DOM 访问关系

节点的访问关系，是以属性的方式存在的。DOM的节点并不是孤立的，因此可以通过DOM节点之间的相对关系对它们进行访问。

##### 2.6.1获取父节点

节点.parentNode （一个节点只有一个父节点）

##### 2.6.2 获取下一个兄弟节点

节点.nextSibling

IE678中指下一个元素节点（标签）。

火狐谷歌IE9+以后都指的是下一个节点（包括空文档和换行节点）。

节点.nextElementSibling

在火狐谷歌IE9都指的是下一个元素节点。

总结：在IE678中用nextSibling，标准浏览器用nextElementSibling

##### 2.6.3 获取前一个兄弟节点

节点.previousSibling

IE678中指前一个元素节点（标签）。

在标准浏览器指的是前一个节点（包括空文档和换行节点）

节点.previousElementSibling

在火狐谷歌IE9都指的是前一个元素节点。

总结：在IE678中用previousSibling，标准浏览器用previousElementSibling。

##### 2.6.4 获取第一个子节点

父节点.firstChild

IE678中指第一个子元素节点（标签）。

在标准浏览器中指第一个节点（包括空文档和换行节点）。

父节点.firstElementChild

在火狐谷歌IE9都指的第一个元素节点。

同理用法还有获取最后一个子节点的方法：

lastChild lastElementChild

##### 2.6.5 获取所有子节点

子节点数组 = 父节点.childNodes; 获取所有节点。

childNodes是标准属性，返回包括HTML节点，所 有属性，文本节点。

注意：火狐 谷歌等高本版会把换行也看做是子节点

nodeType==1时才是元素节点(标签)

nodeType == 1 表示的是元素节点 （元素就是标签）

nodeType == 2 表示是属性节点

nodeType == 3 是文本节点

子节点数组 = 父节点.children; 用的最多。

children是非标准属性，只返回HTML节点，甚至不返回文本节点，

注意：虽然不是标准的DOM属性，但它和innerHTML方法一样，得到了几乎所有浏览器的支持。children在IE6/7/8中包含注释节点 ，在IE678中，注释节点不要写在里面。

##### 2.6.6 随意得到兄弟节点

节点自己.parentNode.children[index];随意得到兄弟节点。

##### 2.6.7 其他

定义一个函数。必须传递自己。定义一个数组，获得自己的父亲，在获得自己父亲的所有儿子（包括自己）。遍历所有的儿子，只要不是自己就放进数组中。

function siblings(elm) {

var a = [];

var p = elm.parentNode.children;

for(var i =0,pl= p.length;i<pl;i++) {

if(p[i] !== elm) a.push(p[i]);

}

return a;

}

### 3 节点属性

获取：getAttribute(名称)

设置：setAttribute(名称, 值)

删除：removeAttribute(名称)

注意：IE6、7不支持。

调用者：节点。 有参数。 没有返回值。

### 4 节点使用

##### 4.1 创建节点

使用方法是这样的document.createElement();

新的标签（节点） = document.createElement(“标签名”);

##### 4.2 插入节点

使用方法： 父节点.appendChild();

父节点.appendChild(新节点); 父节点的最后插入一个新节点

使用方法：父节点.insertBefore(要插入的节点，参考节点)；

父节点.insertBefore(新节点,参考节点)在参考节点前插入;

如果参考节点为null，那么他将在节点最后插入一个节点。

##### 4.3 删除节点

用法：用父节点删除子节点。

父节点.removeChild（子节点）；必须制定要删除的子节点

节点自己删除自己：

不知道父级的情况下，可以这么写：node.parentNode.removeChild(node)

##### 4.4 复制节点

想要复制的节点调用这个函数cloneNode()，得到一个新节点。 方法内部可以传参，入股是true，深层复制，如果是false，只复制节点本身。

新节点=要复制的节点.cloneNode(参数) ; 参数可选复制节点

用于复制节点， 接受一个布尔值参数， true 表示深复制（复制节点及其所有子节点）， false 表示浅复制（复制节点本身，不复制子节点）

## 3 BOM

### 3.1简介

Browser Object Model，浏览器对象模型，定义了JS操作浏览器的方法和属性。BOM中的方法属性大部分都属于window对象。

open(url,打开方式)

打开新的一个窗口，url为空，打开一个空白页，打开方式为空，新窗口方式打开。

close()

用法同上

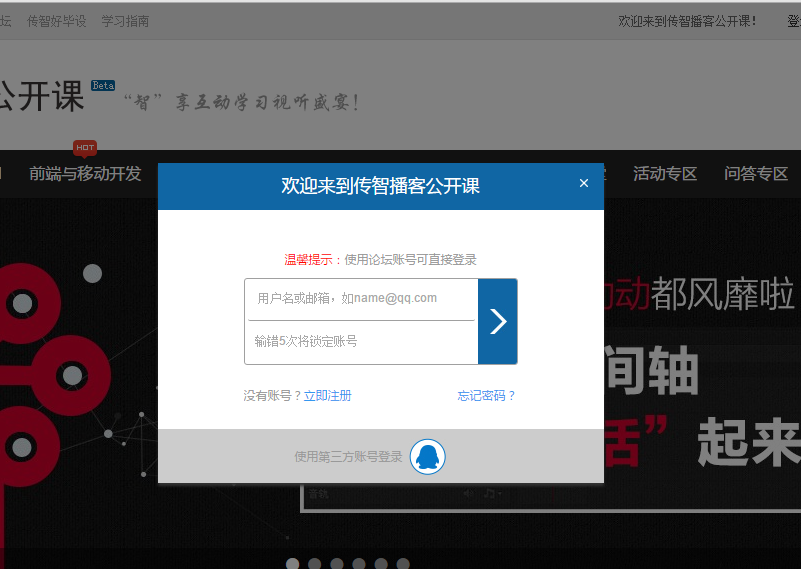
|  |
| --- |
| **window**.**onload** = **function** () {  **var** btns = **document**.getElementsByTagName(**'input'**);  btns[0].onclick = **function** () {  **opener** = **window**.open(); *//另外一个窗口的window对象* **opener**.**document**.**body**.**style**.**backgroundColor** = **'red'**; *//如果打开地址出现了跨域，是无法修改的* }  btns[1].onclick = **function** () {  **opener**.close(); *//可以在当前页关闭新开的窗口，注意：opener不是用var定义的* } } |

|  |
| --- |
| **if**(**window**.**navigator**.**userAgent**.indexOf(**'MSIE'**) == -1){  alert(**'我不是IE'**); }**else** {  alert(**'我是IE'**); } alert(**window**.**location**.**href**); *//url地址* alert(**window**.**location**.**search**); *//参数* alert(**window**.**location**.**hash**); *//#* |

|  |
| --- |
| 可视区域：即窗口中展示给用户的实际界面大小  document.documentElement.clientWidth 同理有Height  滚动距离：文档滚动距离  document.documentElement.scrollTop 同理有Left  document.body.scrollTop Chrome认为滚动不属于文档，滚动属于body  兼容写法：*//因为chrome的document.documentElement.scrollTop是0* **var *scrollTop*** = **document**.**documentElement**.**scrollTop** || **document**.**body**.**scrollTop**;  内容高度：除开边框盒模型内部的高度  element对象.scrollHeight  文档高度：  document.documentElement.offsetHeight  document.body.offsetHeight |

|  |
| --- |
| **var *i*** = 0; **window**.onscroll = **function** () { *//窗口滚动条滚动触发事件* **document**.**title** = ***i***++; }; **window**.onresize = **function** () { *//窗口改变大小触发事件* **document**.**title** = ***i***++; }; |

### 3.2 案例-模态框



<script type="text/javascript">  
 **window**.onload = function(){  
 //事件源: 登录  
 var login = **document**.getElementById("login");  
 var mask = **document**.getElementById("mask");  
 var box = **document**.getElementById("box");  
 login.onclick = function(){  
 // 当我们点击登录，会弹出灰色的大盒子和白色的小盒子  
 mask.**style**.**display** = "block";  
 box.**style**.**display** = "block";  
 }  
 // 事件源 span x  
 var close\_all = **document**.getElementById("close\_all");  
 close\_all.onclick = function(){  
 mask.**style**.**display** = 'none';  
 box.**style**.**display** = "none";  
 }  
 }  
 </script>  
</head>