JavaScript

千锋教育
QIAN FENG EDUCATION

做真实的自己，用良心做教育

张骏
H5教学部

BOM

课程概要:

BOM与DOM概述
BOM的主要属性
window对象
window子对象
核心DOM
节点概念
节点操作
元素操作
HTML DOM

BOM与DOM

- BOM：Brower Object Model浏览器对象模型，用来访问和操作浏览器窗口，使JS有能力与浏览器交互
  - 通过使用BOM、可移动窗口、更改状态栏文本、执行其它不与页面内容发生直接联系的操作
  - 没有相关标准，但被广泛支持
- DOM：Document Object Model文档对象模型，用来操作当前HTML文档的内容
  - 定义了访问和操作HTML文档的标准方法
  - 通过对DOM树的操作，实现对HTML文档树据的操作
  - W3C指定了相关标准

BOM模型

- window对象是BOM的根对象，其它对象其实都是window对象的属性，window对象的属性和方法都可以省略“window.”

对象名	说明
window	表示浏览器中打开的窗口
navigator	包含有关浏览器的信息
screen	包含有关客户端显示屏幕的信息
history	包含用户（在浏览器窗口中）访问过的URL
location	包含有关URL的信息
document	包含当前浏览器加载的文档信息
event	包含当前所触发的事件对象

Window对象概述

- window对象常用属性

属性名	含义
length	返回窗口中的框架数量（HTML5标准中被删除）
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
name	设置或返回窗口名称
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
outerHeight	返回窗口的外部高度
outerWidth	返回窗口的外部宽度

Window对象

- window对象常用方法

方法名	含义
alert()	显示带有一段消息和一个确认按钮的警告框
confirm()	显示带有一段消息以及确认和取消按钮的对话框。
prompt()	显示可提示用户输入的对话框
close()	关闭浏览器窗口
open()	打开一个新的浏览器窗口或查找一个已命名的窗口
blur()	把键盘焦点从顶层窗口移开
focus()	把键盘焦点给予一个窗口
print()	打印当前窗口的内容

Window对象

- window对象常用方法

方法	含义
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。
clearInterval	取消由setInterval()设置的timeout。
clearTimeout	取消由setTimeout()设置的timeout。

窗口的打开和关闭

- 关闭一个窗口：
  - close()
- 可以使用如下方法打开一个新的窗口：
  - window.open(URL,name,features,replace)

参数名称	参数说明
URL	可选，声明了要显示的文档的URL
name	可选，生命了新窗口的名称，这个名称可以用作标记<a>和<form>的属性target的值
features	可选，声明了新窗口要显示的标准浏览器的特征

窗口的打开和关闭（续）

```
var config ="left=200,top=300,width=500,height=300";

    //打开窗口

    var openurl = "http://www.baidu.com";

    var newWin = window.open(openurl,'popwin',config);
```

对话框

- window对象提供了三种对话框：
  - window.alert(msg)　　//弹出一个警告框
  - window.primpt(msg)　　//弹出一个输入提示框
  - window.confirm(msg)　　//弹出一个确认框

定时器

- 多用于网页动态时钟、制作倒计时效果等
- 周期性时钟
  - 以一定的间隔执行代码，循环往复

- 一次性时钟
  - 在一个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

周期性定时器

- setInterval(exp,time):周期性触发代码exp
  - exp：执行语句
  - time：时间周期，单位为毫秒
  - 返回已经启动的定时器ID

- clearInterval(tID):停止启动的定时器
  - tID：启动的定时器对象

一次性定时器

- setTimeout(exp,time):一次性触发代码exp
  - exp：执行语句
  - time：时间周期，单位为毫秒
  - 返回已经启动的定时器ID

- clearTimeout(tID):停止启动的定时器
  - tID：启动的定时器对象

Window常用子对象

- navigator对象
- location对象
- history对象
- screen对象

navigator对象的作用

- navigator对象包含浏览器的信息
  - 常用于获取客户端浏览器和操作系统信息

```
for(var  i in navigator){
    console.log(i+":"+navigator[i]);
        //遍历navigator对象的所有属性
}
```

常用属性

属性名	含义
appCodeName	返回浏览器的代码名。
appMinorVersion	返回浏览器的次级版本。
appName	返回浏览器的名称。
appVersion	返回浏览器的平台和版本信息。
borwserLanguage	返回当前浏览器的语言。
cookieEnabled	返回指明浏览器中是否启用cookie的布尔值。
cpuClass	返回浏览器的CPU等级。
onLine	返回指明系统是否出于脱机模式的布尔值。
platform	返回运行浏览器的操作系统平台。
systemLanguage	返回OS使用的默认语言。
userAgent	返回由客户机发送服务器的user-agent头部的值。
userLanguage	返回OS的自然语言设置。

location对象的作用

- location对象包含有关当前URL的信息
  - 常用于获取和改变当前浏览的网址

```
//获取当前显示的URL
    var url = location.href;
    consolle.log(url);
    //更改要显示的页面URL--页面跳转
    location.href = 'http://www.baidu.com';
```

常用属性和方法

属性名	含义
hash	设置或返回从井号开始的URL（锚）
host	设置或返回主机名和当前URL的端口号
hostname	设置或返回当前的URL的主机名
href	设置或返回完整的URL
pathname	设置或返回当前的URL的路径部分
port	设置或返回当前URL的端口号
protocol	设置或返回当前的URL的协议
search	设置或返回从问号开始的URL（查询部分）

常用属性和方法（续）

方法名	含义
assign()	加载新的文档
reload()	重新加载当前文档
replace()	用新的文档替换当前文档

history对象

- history对象包含用户（在浏览器窗口中）访问过的URL的历史记录

常用属性和方法

属性名	含义
length	返回浏览器历史列表中的URL数量

方法名	含义
back()	加载history列表中的前一个URL
forward()	加载history列表中的下一个URL
go()	加载history列表中的某一个具体页面

screen对象

- Screen对象包含有关客户端显示屏幕的信息，方用于获取屏幕的分辨率和色彩

- JavaScript程序将利用这些信息来优化它们的输出，以达到需要显示要求。

常用属性和方法

属性名	含义
availHeight	返回显示屏幕的高度（除Windows任务栏之外）
availWidth	返回显示屏幕的宽度（除Windows任务栏之外）
bufferDepth	设置或返回调色板的比特深度
colorDepth	返回目标设备或缓冲器上的调色板的比特深度
deviceXDPI	返回显示的每英寸水平点数
deviceYDPI	返回显示屏幕的每英寸垂直点数
fontSmoothingEnabled	返回用户是否在显示控制面板中启用了字体平滑
height	返回显示屏幕的高度
logicalXDPI	返回显示屏幕每英寸的水平方向的常规点数
logicalYDPI	返回显示屏幕每英寸的垂直方向的常规点数
pixelDepth	返回显示屏幕的颜色分辨率
updateInterval	设置或返回刷新屏幕的刷新率
width	返回显示器屏幕的宽度

DOM

DOM概述、文档结构和遍历、读取和修改节点信息

DOM概述

- DOM是W3C（万维网联盟）的标准，是中立于平台和语言的接口，它允许程序和脚本动态的访问和更新文档的内容、结构和样式。
- W3C DOM标准被分为3个不同的部分：
  - 核心DOM 针对任何结构化文档的标准模型
  - XML DOM 针对XML文档的标准模型
  - HTML DOM针对HTML文档的标准模型

DOM概述

- HTML DOM：针对HTML文档的对象模型
  - 当网页被加载时，浏览器会创建页面的文档对象模型
- 通过DOM，可以访问所有的HTML元素，连同它们所包含的文本和属性
  - 可以对其中的内容进行修改和删除，同时也可以创建新的元素
- 文档中的所有节点组成了一个文档树
  - document对象是一颗文档树的根

document对象

- 浏览器内置的JS解释器会为载入的每个HTML文档创建一个对应的document对象

- 通过使用document对象，可以从脚本中对HTML页面中的所有元素进行访问

DOM操作

- 通过可编程的对象模型，JavaScript获得了足够的能力来创建动态的HTML
  - 查找节点
  - 读取节点信息
  - 修改节点信息
  - 创建新节点
  - 删除节点

常用DOM方法		常用DOM属性
getElementById()	createTextNode()	innerHTML
getElementsByTagName()	getAttribute()	parentNode
getElementsByClassNam e()	setAttribute()	childNodes
appendChild()		attributes
removeChild()		
replaceChild()		
insertBefore()		
createAttribute()		
createElement()		

节点树

- 什么是节点树
- HTML DOM将HTML文档视作树结构。
- 文档中的元素、属性、文本、注释等都被看作一个节点。

上下层节点

- 节点树中的节点彼此拥有层级关系，DOM使用如下属性遍历整个节点树：

属性	含义
parentNode	获取父节点
childNodes	获取子节点集合
firstChild	获取第一个子节点
lastChild	获取最后一个子节点

平行的节点

- 节点树中使用如下方法访问平行的兄弟节点：

属性	含义
previousSibling	获取上一个兄弟节点
nextSibling	获取下一个兄弟节点

节点名称 nodeName

- nodeName：节点的名称，String类型属性
  - nodeName 是只读的

节点类型	nodeName
元素节点	标签名
属性节点	属性名
文本节点	始终是#text
注释节点	始终是#comment
文档节点	始终是#document

节点类型 nodeType

- nodeType：节点类型，Number类型属性

节点类型	node Type
元素节点	1
属性节点	2
文本节点	3
注释节点	8
文档节点	9
文档类型声明	10

节点值 nodeValue

- nodeValue：节点的值，String类型属性

节点类型	node Value
元素节点	undefined或null
属性节点	属性值
文本节点	文本本身
注释节点	注释文本本身
文档节点	undefined或null

HTML内容

- 元素节点对象的innerHTML属性读取或设置元素节点中的HTML内容

```
<div id="div1">JavaScript</div>
var div=document.getElementById('div1');
 console.log(div.innerHTML);//读取
    div.innerHTML = 'jQuery';//设置
```

文本内容

- 元素节点对象的textContent属性用于读取或设置元素节点中的文本内容

注：有争议的innerText

标准DOM操作中，并没有innerText属性；

IE8及之前的IE浏览器不支持标准的textContent属性

使用innerText实现类似的功能，目前此属性已被大多数浏览器所兼容，但Firefox仍不支持此属性

属性集合

- 元素节点的attributes属性返回节点的属性集合，即一个类数组对象

```
<div id="div1" onclick="up()" class="add"></div>


  var div=document.getElementById('div1');
    console.log(div.attributes);//类数组对象
    console.log(div.attributes.length);//3
    console.log(div.attributes[1]);//属性对象
```

读取属性

- 可以使用如下几种方式读取某个属性的值:
  （1）element.attributes[下标].value
  （2）element.attributes['属性名'].value
  （3）element.getAttributeNode('属性名').value
  （4）element.getAttribute('属性名')

设置属性

- 可以使用如下两种种方式设置元素的属性:

（1）element.setAttribute(name,value);

（2）element.setAttributeNode(attrNode);

```
<div id="div1" onclick="up()" ></div>
var div=document.getElementById('div1');
    var atr=document.createAttribute('class')
    atr.nodeValue="className";
    div.setAttributeNode(atr);
```

移除属性

- 可以使用如下两种种方式删除一个属性：
  （1）element.removeAttribute('属性名');
  （2）element.removeAttributeNode(attrNode)

判断属性

- 如下方法可用于判定元素是否有指定属性:

（1）element.hasAttribute('属性名');//true或false

（2）element.hasAttributes(); //是否拥有属性、IE8及以下版本不支持此方法

DOM

DOM选取元素、增加、删除和替换节点

选取元素

- document.getElementById('id')可用于当前DOM树中根据ID选择某一个子元素
- Node.getElementsByTagName('标签名')可根据标签名返回所有具有指定标签名的元素集合
- document.getElementsByName('name属性值')可以返回DOM数中具有指定那么属性值的所有子元素集合
- node.getElementsByClassName('className')可以根据class名称选取元素的方法（IE9+、Firfox3+、Safari3.1+、Chrome和Opera9.5+）

选取元素

- 通过css选择器选取元素

（1）node.querySelector('selector')//返回第一个匹配的

（2）node.querySelectorAll('selector')//返回全部匹配的

```
<div id="div1">
        <p class="p1">中国</p>
        <p class="p2">澳门</p>
        <p class="p2">台湾</p>
        <p class="p2">香港</p>
    </div>
var div= document.getElementById('div1');
    var p= div.querySelectorAll('.p2');
    console.log(p);
```

其它选取

- document.documentElement返回整个HTML文档的根元素（即<html>元素）
- document.head返回HTML文档中<head>元素
- document.body返回HTML文档中<body>元素

创建节点

- 创建节点使用如下方法创建一个新的元素
  - document.createElement('元素名');

- 创建文本节点使用如下方法可以创建一个新的文本节点
  - document.createTextNode('text');

插入节点

- parentNode.appendChild(childNode)可用于将一个父元素追加最后一个子节点

- parentNode.insertBefore(newChild,existingChild)方法用于在父元素中指定子节点之前添加一个新的子节点

删除节点

- 可以使用parentNode.removeChild(childNode);
  - 此方法返回被删除的节点的引用

替换节点

- 可以使用如下方法替换一个已经存在的子节点:

parentNode.replaceChild(newNode,oldNode);

节点树VS元素树

节点树	元素树
节点有若干类型：文档、文档类型、元素、文本、注释、属性....	节点只有两个类型：元素(标签)、文档
parentNode	parentElementNode
childNodes	children
firstChild	firstElementChild
lastChild	lastElementChild
previousSibling	previousElementSibling
nextSibling	nextElementSibling

HTML DOM、BOM

HTML DOM概述

- HTML DOM定义了用于HTML的一系列标准的对象，以及访问的处理HTML文档的标准方法
- HTML标签对象化
  - 网页中的每个元素都看作一个对象

常用HTML DOM对象

```
var div = document.getElementById('div');

    //创建一个新的图片对象，加入到div中
    var newNode = new Image();

    newNode.src="a.jpg";

    div.appendChild(newNode);//将HTML对象化
```

标准DOM与HTML DOM

- 标准DOM提供了统一的操作接口
  - createElement
  - appendChild
  - setAttribute
  - removeAttribute
  - nodeName...
- HTML DOM提供了封装好的各种对象
  - Image
  - Select
  - Option...

标准DOM与HTML DOM（续）

- 标准DOM的实现方式

```
var newNode = document.createElement('img');
```

- HTML DOM的实现方式

```
var newNode = new Image();
```

标准DOM与HTML DOM

- 标准DOM操作适合于：
  操作节点，创建，删除，查找等

- HTML DOM操作适合于：
  操作属性，如读取或者修改属性的值

CSS属性

- 获取元素宽高
- offsetWidth/offsetHeight
- 获取定位值
- offsetTop/offsetLeft

- 获得计算后的CSS样式
- getComputedStyle(elm)[‘attr’]　// 标准
- elm.currentStyle[‘attr’]　// 低版本ie

常用HTML DOM对象

- Image对象
- Image对象代表嵌入的图像
- <img>标签每出现一次，一个Image对象就会被创建
- 也可以使用new Image()创建一个新的对象
- 常用属性：
  - src
  - height
  - width

Table对象

- Table对象代表一个HTML表格
  - <table>标签标示一个Table对象
- 常用属性
  - rows
- 常用方法
  - insertRow(index)：返回TableRow对象
  - deleteRow(index)

TableRow对象

- TableRow对象代表一个HTML表格行
- <tr>标签标示一个TableRow对象
- 常用属性
  - cells、innerHTML、rowIndex
- 常用方法
  - insertCell(index)：返回TableCell对象
  - deleteCell(index)

TableCell对象

- TableCell对象代表一个Html表格单元格
  - <td>标签标示一个TableCell对象
- 常用属性
  - cellIndex、innerHTML、colSpan、rowSpan

文档碎片(了解)

- document.createDocumentFragment()
- 用于创建文档碎片 优化DOM操作的频率

THANK YOU

千锋教育
QIAN FENG EDUCATION

做真实的自己，用良心做教育