# 第16天【DOM基础】

## 主要内容

1. DOM介绍
2. Document节点常用属性
3. 选中页面元素
4. 创建页面元素
5. 操作页面元素属性
6. 事件

## 学习目标

|  |  |  |
| --- | --- | --- |
| 节数 | 知识点 | 要求 |
| 第一节 | DOM介绍 | 了解 |
| Document节点常用属性 | 了解 |
| 第二节 | 选中页面元素 | 掌握 |
| 创建页面元素 | 掌握 |
| 第三节 | 操作页面元素 | 了解 |
| 第四节 | 事件 | 掌握 |

1. **DOM介绍**

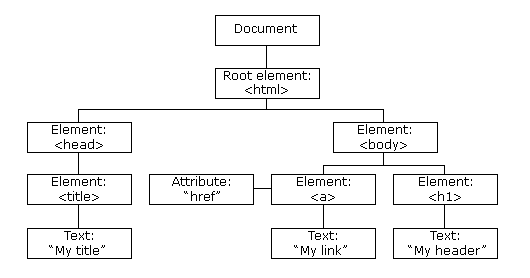
**1.1DOM概念**

DOM全称为Document Object Model，即文档对象模型。它是一套用来管理控制html文档的规则。

**1.2节点**

在 HTML DOM 中，所有事物都是节点。DOM 是被视为节点树的 HTML。

HTML DOM 树：



根据 W3C 的 HTML DOM 标准，HTML 文档中的所有内容都是节点：

1、整个文档是一个文档节点

2、每个 HTML 元素是元素节点

3、HTML 元素内的文本是文本节点

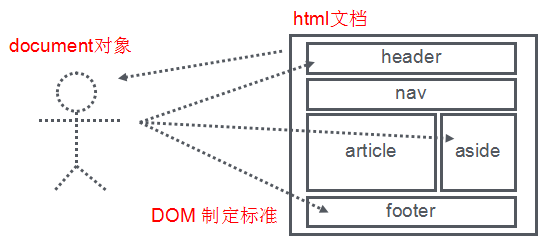
4、每个 HTML 属性是属性节点

5、注释是注释节点

**1.3document节点**

document节点又被叫做document对象。每个载入浏览器的 HTML 文档都会成为 document对象。

document是html文档的根节点，每张网页都有自己的document节点。window.document属性就指向这个节点。也就是说只要浏览器开始载入HTML文档，这个节点对象就存在了，可以直接调用。



1. **Document节点常用属性**
2. **Title**
3. 定义

title属性返回当前文档的标题，该属性是可写的。

2）语法

document.title

3）例子

document.title = '新标题';

1. **Location**

1）定义

返回网页完整的URL

2）语法

document.location.href

3）例子

location.href = 'http://www.baidu.com';

1. **选中页面元素**

**3.1通过id获取页面元素**

|  |
| --- |
| //获取带有指定 id 的节点（元素）  document.getElementById("idName") |

|  |
| --- |
| //id获取元素  var header = document.getElementById("header"); |

**3.2通过标签名获取元素，返回一个数组**

|  |
| --- |
| //返回包含带有指定标签名称的所有元素的节点列表（集合/节点数组）  node.getElementsByTagName("tagName") |

|  |
| --- |
| //标签获取元素  var li = document.getElementsByTagName('li'); |

**3.3通过class名字获取页面元素，返回一个数组**

|  |
| --- |
| //返回包含带有指定类名的所有元素的节点列表  document.getElementsByClassName("className") |

注意：getElementsByClassName() 在 Internet Explorer 5,6,7,8 中无效。

|  |
| --- |
| //返回一个数组，通过下标访问文本  var list = document.getElementsByClassName('list'); |

**3.4通过CSS选择符方式获取页面元素**

|  |
| --- |
| //querySelector：返回符合条件的第一个  document.querySelector("selector")  //返回符合条件的全部  document.querySelectorAll("selector") |

|  |
| --- |
| //querySelector：返回符合条件的第一个  　　var str = document.querySelector('li');  //全部返回，每个li都是一个对象  　　var str1 = document.querySelectorAll('li'); |

1. **创建页面元素**

**4.1创建元素**

1、定义

createElement() 方法通过指定名称创建一个元素

2、语法

|  |
| --- |
| document.createElement(nodename) |

3、例子

创建一个div

|  |
| --- |
| var box=document.createElement("div"); |

**4.2创建文本节点**

1、定义

createTextNode() 可创建文本节点。

2、语法

|  |
| --- |
| document.createTextNode(text) |

3、例子

创建一个文本节点:

|  |
| --- |
| var btn=document.createTextNode("Hello World"); |

HTML元素通常是由元素节点和文本节点组成。

创建一个标题 (H1), 你必须创建 "H1" 元素和文本节点:

|  |
| --- |
| var h=document.createElement("H1")  var t=document.createTextNode("Hello World");  h.appendChild(t); |

**4.3创建属性**

1、定义

createAttribute()方法用于创建一个指定名称的属性，并返回Attr 对象属性。

2、语法

|  |
| --- |
| document.createAttribute(attributename) |

1. 例子

创建一个div

var box=document.createElement("div");

**4.4属性赋值**

|  |
| --- |
| node.value = “属性值” |

**4.5设置属性节点**

1、定义

setAttributeNode() 方法用于添加新的属性节点。

2、语法

|  |
| --- |
| element.setAttributeNode(attributenode) |

3、例子

创建 class 属性, 属性值为 "democlass", 并将clas属性添加到 H1 元素中:

|  |
| --- |
| var att=document.createAttribute("class");  att.value="democlass";  h1.setAttributeNode(att); |

**4.6作为子节点追加到父节点下**

1. 定义

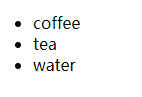
appendChild() 方法可向节点的子节点列表的末尾添加新的子节点。

2、语法

|  |
| --- |
| node.appendChild(node) |

3、例子

|  |
| --- |
| <ul id="ul">  <li>coffee</li>  <li>tea</li>  </ul>  <script>  var ul = document.getElementById("ul");  var li = document.createElement("li");  var text = document.createTextNode("water");  li.appendChild(text);  ul.appendChild(li);  </script> |



**4.7实例1**

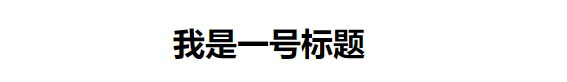
创建一个div元素，内容为我是一个文本，样式为宽300px，高为300px，背景颜色为粉色

|  |
| --- |
| //创建元素  var div = document.createElement("div");  //元素插入到页面中  document.body.appendChild(div);  //创建文本  var text = document.createTextNode("我是文本");  //文本追加到元素中  div.appendChild(text);  //创建属性  var attr = document.createAttribute("style");  //属性赋值  attr.value = "width:200px;height:200px;background-color:pink;"  //设置属性节点  div.setAttributeNode(attr); |



**4.8实例2**

|  |
| --- |
| //创建元素  var h1 = document.createElement("h1");  //元素插入到页面中  document.body.appendChild(h1);  //创建文本  var text = document.createTextNode("我是一号标题");  //文本追加到元素中  h1.appendChild(text);  //创建属性  var alignAttr = document.createAttribute("align");  //属性赋值  alignAttr.value = "center"  //设置属性节点  h1.setAttributeNode(alignAttr);  //创建属性  var titleAttr = document.createAttribute("title");  //属性赋值  titleAttr.value = "我是一号标题";  //设置属性节点  h1.setAttributeNode(titleAttr); |



1. **操作页面元素属性**

CSS与JavaScript是两个有着明确分工的领域。前者负责页面的视觉效果，后者负责与用户的行为互动。但是它们毕竟同属网页开发当中的技术，因此不可避免有着交叉和互相配合。

所以在整体页面布局的过程中我们推荐使用html+css的方式来编写页面的结构和样式。在细节以及交互模块的编写过程中我们推荐使用js的方式来辅助编写。

很显然对于整个html页面来说，css部分所代表的样式就显得更为重要一些。因为这些样式能够决定我们的页面整体显示效果。因此js中也提供了几种操作页面元素属性的方法：

**5.1元素节点的方法**

操作Element节点的CSS样式，最简单的方法之一就是使用节点对象的getAttribute方法、setAttribute方法和removeAttribute方法，读写或删除HTML元素的style属性。

#### getAttribute() 方法

1. 定义

getAttribute() 方法通过名称获取属性的值。

1. 语法

*element*.getAttribute(attributename)

1. 例子

document.getElementsByTagName("a")[0].getAttribute("target");

#### 2、setAttribute() 方法

1）定义

setAttribute() 方法创建或改变某个新属性。

如果指定属性已经存在,则只设置该值。

2）语法

element.setAttribute(attributename,attributevalue)

3）例子

document.getElementsByTagName("INPUT")[0].setAttribute("type","button");

#### 3、removeAttribute() 方法

1）定义

 removeAttribute() 方法删除指定的属性

2）语法

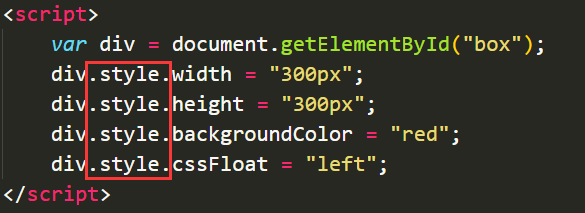
element.removeAttribute(attributename)

3）例子

document.getElementsByTagName("H1")[0].removeAttribute("style");

**5.2元素节点的style属性**

Element节点本身还提供style属性，用来操作CSS样式。style属性指向一个对象，用来读写页面元素的行内CSS样式。



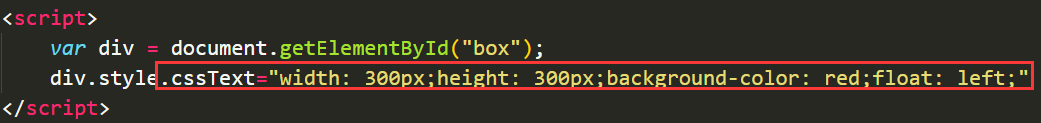
元素的style对象对应元素的style属性，style对象中的样式与元素style属性中的样式名是一一对应的，但是需要一点点改写规则:

a.将横杠从CSS属性名中去除，然后将横杠后的第一个字母大写

b.CSS属性名是JavaScript保留字的，在属性名之前需要加上字符串“css”

c.style对象的属性值都是字符串，而且包括单位。

style对象的cssText属性可以用来读写或删除整个style属性。因此刚才的写法可以改写成：



删除整个style属性可以用(div.cssText='' '';)这种写法。

**5.3className属性**

className 属性设置或返回元素的 class 属性。

|  |
| --- |
| HTMLElementObject.className=classname |

1. **事件**

HTML 事件是发生在 HTML 元素上的事情。

当在 HTML 页面中使用 JavaScript 时， JavaScript 可以触发这些事件。

HTML 事件可以是浏览器行为，也可以是用户行为。

以下是 HTML 事件的实例：

* 1、HTML 页面完成加载
* 2、HTML input 字段改变时

3、HTML 按钮被点击

通常，当事件发生时，你可以做些事情。

在事件触发时 JavaScript 可以执行一些代码。

可以使用多种方法来执行 JavaScript 事件代码：

1. HTML 事件属性可以直接执行 JavaScript 代码

HTML 元素中可以添加事件属性，使用 JavaScript 代码来添加 HTML 元素。

单引号:

<some-HTML-element some-event=**'JavaScript 代码'**>

双引号:

<some-HTML-element some-event=**"JavaScript 代码"**>

在以下实例中，按钮元素中添加了 onclick 属性 (并加上代码):

|  |
| --- |
| <button onclick="alert('恭喜您中了1个亿');">点我有惊喜！！</button> |

1. HTML 事件属性可以调用 JavaScript 函数

|  |
| --- |
| <button onclick="fun();">点我有惊喜！！</button>  <script>  function fun(){  alert("恭喜您中了1个亿");  }  </script> |

1. 你可以为 HTML 元素指定自己的事件处理程序

|  |
| --- |
| <button id="btn">点我有惊喜！！</button>  <script>  var btn = document.getElementById("btn");  btn.onclick = function(){  alert("恭喜您中了1个亿");  }  </script> |

1. **作业**
2. **js实现水平导航条**

|  |
| --- |
| //创建数组 存储a里面的文本  var textArr = ["小米商城","金融","有品","小爱开放平台","企业团购","协议下载"];  //创建ul  var ul = document.createElement("ul");  //追加到body中  document.body.appendChild(ul);  //控制创建li 执行六遍  for(var i = 0; i < textArr.length; i++){  //创建li  var li = document.createElement("li");    //追加到ul中  ul.appendChild(li);  //创建a  var a = document.createElement("a");  //追加到li中  li.appendChild(a);  //设置属性 href = "#"  a.setAttribute("href","#");  //设置a元素的内容  a.innerHTML = textArr[i];  //li设置样式  li.style.cssText = "list-style-type: none;float: left;padding: 20px 40px;background-color: #ff6700;";  //设置a的样式  a.style.cssText = "text-decoration: none;color: #fff;";  } |

1. **轮播图**

HTML部分：

|  |
| --- |
| <div class="banner">  <img src="images/5f8d27f0498561818146b7ef78392f33.webp" alt="" id="img"/>  <div class="left"></div>  <div class="right"></div>  <ul class="circle">  <li onclick="banner(0);"></li>  <li onclick="banner(1);"></li>  <li onclick="banner(2);"></li>  <li onclick="banner(3);"></li>  <li onclick="banner(4);"></li>  </ul>  </div> |

CSS部分

|  |
| --- |
| .banner{  width: 1226px;  height: 460px;  margin: 0 auto;  position: relative;  }  .left,.right{  width: 41px;  height: 69px;  background-color: rgba(0,0,0,.6);  position: absolute;  top: 50%;  margin-top: -34.5px;  cursor: pointer;  }  .left{  left: 0;  }  .right{  right: 0;  }  .banner>img{  width: 100%;  }  .circle>li{  width: 6px;  height: 6px;  border: 2px solid red;  border-radius: 50%;  float: left;  margin: 0 4px;  cursor: pointer;  }  .circle{  position: absolute;  right: 30px;  bottom: 20px;  } |

JavaScript部分

|  |
| --- |
| //存储图片  var imgArr = ["1.webp","2.jpg","3.webp","4.webp","5.webp"];  //获取页面元素  var left = document.querySelector(".left");  var right = document.querySelector(".right");  var img = document.getElementById("img");  var list = document.querySelectorAll(".circle>li");  var i = 0;  //定义右侧按钮的时间  right.onclick = function(){  i++;  if(i > 4){  i = 0;  }  img.src = "images/" + imgArr[i];  };  //定义左侧按钮的事件  left.onclick = function(){  i--;  if(i < 0){  i = 4;  }  img.src = "images/" + imgArr[i];  };  //定义函数用来切换图片  function banner(index){  //循环遍历数组 拿到数组的下标  for(var i = 0; i < imgArr.length; i++){  if(index == i){  img.src = "images/"+ imgArr[i];  list[i].style.backgroundColor = "red";  }else{  list[i].style.backgroundColor = "";  }  }  } |