DOM

1.DOM的概念

DOM文档对象模型,DOM对象不仅仅是一个普通的内置对象,它还是一个巨大的API的核心对象,它将文档的内容呈现在js面前,并赋予了js操作文档能力.

2.DOM的节点

概念：加载html页面是,web浏览器生成一个树形结构,用来表示页面内部结构,DOM将这种结构理解为节点组成.根据w3c的html.DOM标准,html文档找那个的所有内容都是节点.

2.1查找节点

2.1.0节点属性

<div id="mydiv">这是一个div</div>

<script>

var mydiv = document.getElementById('mydiv');

console.log(mydiv.nodeName);//查找元素

console.log(mydiv.nodeValue);//查找value值

console.log(mydiv.nodeType);

document.body.nodeName

</script>

2.1.1获取第一个节点和最后一个节点

<ulid="myul"><li>11</li><li>22</li><li>33</li></>

<script>

var myul = document.getElementById('myul');

//获取最后一个节点和第一个节点

console.log(myul.firstChild);//第一个节点

console.log(myul.lastChild);//最后一节点

</script>

2.1.2获取父节点

<ul id="myul"></ul>

<script>

var myul = document.getElementById('myul');

//childNodes是所有节点

console.log(myul.parentNode);//获取上一级的元素

</script>

2.1.3获取子节点

<ul id="myul"><li></li><li></li><li></li> </ul>

<script>

var myul = document.getElementById('myul');

//childNodes是所有节点

console.log(myul.childNodes);//获取所有节点

</script>

2.1.4获取兄弟节点

<h2>h2</h2><div id="mydiv"></div><p>p</p>

<script>

var mydiv = document.getElementById('mydiv');

console.log(mydiv.previousSibling);//前一个元素

console.log(mydiv.nextSibling);//后一个元素

</script>

2.2获取元素和设置元素

2.2.1用id获取元素

var mytag = document.getElementById('mytag');

console.log(mytag.tagName);

2.2.2name获取元素

<input type="radio" name="sex" value="0">男

<input type="radio" name="sex" value="1">女

<br>

<input type="checkbox" name="ball">足球

<input type="checkbox" name="ball">篮球

<input type="checkbox" name="ball">网球

<script>

var ball = document.getElementsByName('ball');

console.log(ball);

</script>

2.2.3获取元素节点中的内容

var ul = document.getElementById('ul');

console.log(ul.innerHTML);

2.2.4获取元素节点中文本

var ul = document.getElementById('ul');

console.log(ul.innerText);

ul.innerText = 'helloe'

2.2.5获取元素类名

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

console.log(div.className);

2.2.6获取元素样式

var div=document.getElementsByTagName('div');

//js拿到的是内联

console.log(div[1]);

2.2.7获取元素属性

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

var a = document.getElementsByTagName('a');

console.log(div.getAttribute('id'));

console.log(a[0].getAttribute('href'));

2.2.8设置元素属性

Vardiv=document.getElementsByTagName('div');

var a = document.getElementsByTagName('a');

div[0].setAttribute('id','mydiv');

a[0].setAttribute('href','www.baidu.com');

2.2.9删除元素属性

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

div.removeAttribute('id');

2.3增加元素

2.3.1创建元素节点

//用一个变量承接在文档中“创建”的一个元素

var div = document.createElement('div');

//将已经创建的元素放在想放的位置

document.body.appendChild(div);

2.3.2创建元素文本

var p = document.getElementById('p');

var text = document.createTextNode('一段文');

p.appendChild(text);

2.3.3 css样式赋值

div.style.cssText='width:100px;height:100px;background-color:red;'

2.3.3 在后元素插入创建的新元素

<ul id="ul">

<li id="pg">苹果</li>

<li id="jz">橘子</li>

<li>香蕉</li>

</ul>

<script type="text/javascript">

var ul = document.getElementById('ul');

//获取苹果

var pg = document.getElementById('pg');

//获取橘子

var jz= document.getElementById('jz');

var li = document.createElement('li');

//在父级中插入一个元素（插入元素，在谁前面）

ul.insertBefore(li,jz);

var t = document.createTextNode('火龙果');

li.appendChild(t);

</script>

2.4删除和替换元素

2.4.1元素替换

var mydiv = document.getElementById('mydiv');

var myp = document.createElement('p');

//在父元素中替换子元素第一个是新的第二个是旧的

document.body.replaceChild(myp,mydiv);

2.4.2元素的删除

var mydiv = document.getElementById('mydiv');

document.body.removeChild(mydiv);

2.5元素的位置属性

2.5.1 offsetWidth和offsetHeight

var erzi = document.getElementById('erzi');

console.log(erzi.offsetWidth)

console.log(erzi.offsetHeight)

2.5.2 clientWidth和clientHeight

var mydiv = document.getElementById('mydiv')

console.log(mydiv.clientWidth);

2.5.3子元素和父元素的距离

var erzi = document.getElementById('erzi');

console.log(erzi.offsetLeft);

console.log(erzi.offsetTop);