DOM

概念

DOM是一个内置对象,是API的核心对象,并赋予了js操作文档的能力.

Dom节点

根据w3c的html DOM标准,html文档找那个的所有内容都是节点

整个文档是一个文档节点

每个html元素是元素节点.

html元素内的文本是文本节点

每个html属性都是属性节点

注释也是节点,叫注释节点

DOM树

DOM树有DOM文档树和DOM元素树两种,

DOM文档树则包括DOM文档中所有的内容.

获取元素

用id获取元素

getElementById()的方法,接受一个参数:获取元素的id,如果找到相应元素

则返回该元素的,否则返回null.

方法var d=document.getElementById('d'); x

用标签名获取元素

getElementsByTagName()获取该元素名称下所有的元素,返回一个伪数组,

或者说是节点列表.

方法 var lis=document.getElementsByTagName('li');

在某一个元素中,并不是在全局文档中利用标签获取元素,这样更加精准

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

var list=ul.getElementsByTagName('li');

当元素只有一个的时候,返回的也依然是一个列表,而不是一个元素.可以 通过数组的下标找到该元素.

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

用类名获取元素

方法 var a=document.getElementsByClassName('a');

用name属性值获取元素

getElementsByName()方法可以获取相同名称的name元素,返回一个伪数组对象.

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

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

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

console.log(sex);

用选择器获取元素

querySelector()和querySelectorAll()可以依靠选择器找到元素,但是前者只能找到元素列表的第一个元素

,而后者可以全部找到.注意该方法性能没有直接利用标签寻找高.

var ul = document.getElementById('ul')

var x = document.querySelector('#div>div a');

var lis = ul.querySelectorAll('li');

获取元素名

当我们使用id获取元素的时候，该属性只能获取不能设置.

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

获取元素节点里的内容

元素里的内容可能包括:文本或元素

需要用innerHTML属性.

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

console.log(div.innerHTML);

innerHTML属性除了可以获取标签内的内容外,还可以设置标签内的内容.

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

div.innerHTML='你好';

获取元素节点中的文本

利用innerHTML属性获取的只能是文本节点,也可以设置元素中的文本

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

console.log(div.innerHTML);

div.innerText='hello';

获取元素的类名

利用className属性获取元素的类名,以字符串的形式返回.同时可以设置新的class类名,需要注意的是,返回值是一个字符串.

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

获取元素的属性

getAttrbute('')可以获取元素的某个属性,

设置元素属性

setAttribute()可以设置元素的某个属性,第一个参数是属性名,第二个参数是属性值

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

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

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

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

删除元素属性

使用removeAttribute(),可以删除某个属性,括号中放入需要删除的属性名,用引号包裹.

var div = document.getElementById('mydiv')

创建元素节点

利用js创建一个元素的方法,先在文档中创建一个标签document.createElement()括号

中写标签名,当然要用字符串形式,创建之后不是就存在了.而是需要将这个已经创建的元素放到

你想放到的元素(父级元素)中去.

//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);

创建文本节点

利用js创建元素的文本节点,先在文档中创建文本节点createTextNode('添加一段文本'),在括号中将要添加的字符串放入,

最后插入到需要的元素中.

var text = document.createTextNode('添加一段文本')

var p = document.getElementById('p')

p.appendChild(text)

元素的替换

元素的替换是在父元素中一个子元素需要被另一个新的子元素所替代,使用replaceChild()方法,

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

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

document.body.replaceChild(myp,mydiv);

元素的删除

元素的删除是在父元素的其中一个子元素需要删除,使用removeChild()方法,将需要删除的元素放进括号里.

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

document.body.removeChild(mydiv);

查找节点(node)

常用查找元素,文本,属性,注释.

取所有子节点

使用 childNodes属性拿到的是该元素下的所有节点

var myul=document.getElementById('myul'); console.log(myul.childNodes);

获取第一个和最后一个子节点

用firstChild和lastChild可以拿到该父元素下得第一个和最后一个子节点,注意不一定是元素

<ul id="myul"><li>111</li><li>222</li><li>333</li></ul>

<script type="text/javascript">

console.log(myul.firstChild)

console.log(myul.lastChild)

获取父节点

使用parentNode属性可以拿到该元素的父节点,注意父节点只有一个

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

<script type="text/javascript">

var myul = document.getElementById('myul') console.log(myul.parentNode)

获取兄弟节点

使用previousSibling和nextElementSibling可以获取该元素的前一个和后一个兄弟节点,只能获取一个

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

console.log(mydiv.previousSibling);

console.log(mydiv.nextElementSibling); offsetWidth和offsetHight

需要获取一个元素的宽度和高度,使用style是无法获取的,所以我们选择使用offsetWidth和offsetHight属性

该属性可以获取元素的站位宽高,它也含内边距和边框

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

console.log(mydiv.style)//这种方法只能拿到内联样式

console.log(mydiv.offsetWidth);//含内边距 不含外边距

clientWidth和clientHight

clientWidth和clientHight属性也可以获取元素的宽高,但与offsetWidth不同的是不包含边框

console.log(mydiv.clientWidth);//不包括内边框

子元素与父元素的距离

offsetLeft和offsetTop是距离body左边界和上边距的距离,但如果盖子元素使用了定位属性

则 offsetLeft和offsetTop参照的就不再是body而是距离它最近的父级元素

<div id="baba">

<div id="erzi">

</div>

</div>

<script>

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

console.log(erzi.offsetLeft);

console.log(erzi.offsetTop);