DOM

用id获取元素

getElementById（）的方法，接受一个参数：获取元素的id，如果找到相应的元素，则返回该元素的，否则返回null。

//用变量接 在文档中找 元素 用id “d”

var d = document.getElementById("p");

获取元素名

当我们使用id获取元素的时候，元素的名称会和整个元素一起显示出来，如果想要拿到该元素的元素名，也就是标签名则需要用

tagName属性，该属性只能获取，不能设置。

getElementsByTagName()：通过节点名称，查找对应节点。

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

用类名获取元素:var a = getElementsByClassName（“a”）；

console.log（a）;

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');

console.log(lis);

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.

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

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

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

console.log(sex);

用标签名获取元素:getElementsByTagName（）可以获取该元素名称下所有的元素，返回一个伪数组，或者说是一个节点列表

var lis = document.getElementsByName("li");

//伪数组

console.log(lis);

获取元素节点里的内容:当获取了元素之后，如果我们需要拿到元素中的内容（所有东西），则需要用另一种方法得到，元素里的内容可能包括：文本和元素。

var div = document.getElementById（"madiv"）；

console.log（div.innerHTML）;

获取元素节点中的文本:利用innerText属性获取的只能是文本节点，不能是其他，当然innerText属性除了获取，也可以设置元素中的文本。

var div = document.getElementById（"madiv"）；

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.

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

console.log(div.className);

获取元素样:style属性可以获取元素内联样式的所有属性，当然如果继续在style中找属性名如：backgroundcoclor，就可以得到该属性的值，以字符串的形式返回，同事也可以设置该属性，从而达到增加或更改样式，需要注意的是，以js形式增加的样式的优先级大于css优先级。

<div style="background-color: brown;"></div>

<script>

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

//js拿到和获取的是内联

console.log(div[0].style.backgroundColor);

</script>

获取元素的属性 :getAttribute（""）可以获取元素的某个属性，要将属性名称放在括号中记得要用引号括起来，返回值就是字符串形式的属性值。

var div document getElementById('mydiv')

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

console. log(div getAttribute('data-1j'))

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

设置元素属性:setAttribute()可以设置元素某个属性,第一个参数是属性名,第二个参数是属性值,都需要用引号括起来以逗号分隔.

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(),可以删除某个元素的某个属性,括号中放入需要删除的属性名,用引号包裹.

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

div.removeAttribute('id');

创建元素节点:利用js创建一个元素的方法是，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要的元素（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);

css样式赋予:style.cssText是一个css的样式集合，它可以把scc层叠样式表的css样式直接放在该属性后，就不需要一条一条去赋予样式了，可以一次性赋予样式。

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

div.style.cssText = "width: 100px;height: 100px;background-color: blue;"

insertBefore()这个函数和appendChild()用法基本一致,都是向父级中插入一个子元素,但是区别是,insertBefore是可以选择插入的位置,它需要插入到某一个子元素之前的位置,因此需要那个子元素,insertBefore()有两个参数,第一个参数是需要插入两个元素,第二个参数是在谁之前插入,两个参数用逗号分隔.

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，该方法有两个参数，第一个参数是将要替换的新元素，第二个是被替换的旧元素，中间用逗号分隔。

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

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

//在父元素替换子元素，第一个是新的元素，第二个是旧的

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入括号中。

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

document.body.removeChild(div);

3查找节点（node）

节点是在dom树上的每一个节点，其中包括：元素，文本，属性，注释等等，常用的有三个，元素，文本，属性。

ar mydiv = document.getElementById("mydiv");

console.log(mydiv.nodeValue);

console.log(mydiv.nodeType);

获取所有子节点

使用childNodes属性拿到的是该元素下的所有节点，包含有节点类型，不单单只是元素

<ul><li>111</li><li>222</li><li>333</li></ul>

var myu1 = document.getElementById("mydiv");

//childNodes是所有节点

console.log(myu1.childNodes);

</script>

获取第一个和最后一个子节点

用firstChild和lastChild可以拿到该父元素的第一个和最后一个子节点，注意不一定是节点。

console.log(myu1.firstChild);

console.log(myu1.lastChild)；

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。

console.log(myu1.parentNode);

获取兄弟节点

使用previousElementSibling和nextSibling可以获得该元素的前一个和后一个兄弟节点，只能获取一个。

offsetWidth和offsetHeight

需要获取一个元素的宽度和高度，使用style是无法获取的，所以我我们选择使用offsetWidth和offsetHeight属性，该属性可以获取元素的占位宽高。他包含了内边距和边框。

clientWidth和clientHeight属性也可以获取元素的宽和高，但与offsetWidth不同的是不包含边框。

offsetLeft和offsetTop是距离body左边界和上边界的距离，但如果该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。