**元素选择器：**

[**js中的原生元素选择器**](http://www.cnblogs.com/rongy/p/6698815.html)

当我们想要选择dom中的元素时，第一个想到的就是document.getElementById()和document.getElementsByTagName()，还有针对表单的document.getElementsByName();

在h5中我们又有了类选择器：getElementsByClassName(),但这个方法不兼容IE8及以下,我么在使用时还是需要写一个兼容的函数，因为现在还是很多IE8的，至于为什么不考虑IE8以下，呵呵，我给你一个眼神，自己体会。

但在前几天我突然看到书上的两个选择器querySelect()和querySelectAll(),发现他们竟然兼容IE8，这个真的是太棒了。

他们的用法和getElement用法是一样的,但是选择符适合JQuery一样的！：

[复制代码](javascript:void(0);)

<ul>

<li class="aa"></li>

<li class ="a"></li>

<li class="b aa"></li>

<li class="aa"></li>  
　　 <li id="a"></li>

</ul>

[复制代码](javascript:void(0);)

我们使用querySelect()

console.log(document.querySelector('#a'));

console.log(document.querySelector('.a'));

console.log(document.querySelector('.aa'));

我们发现输出的是 <li id="a"></li>,<li class = 'a'></li>,<li class="aa"></li>

这说明querySelector()选择的只是所有中的第一个元素，而且返回的是元素对象，不是数组。

使用querySelectAll()

console.log(document.querySelectorAll('#a'));

console.log(document.querySelectorAll('.a'));

console.log(document.querySelectorAll('.aa'));

这时输出的是[li#a],[li.a],[li.aa,li.b.aa,li.aa];

这说明querySelector()选择的是所有符合条件的元素，并返回数组。

因为这两个方法兼容IE8所以可以大胆的使用

**JS获取子节点、父节点和兄弟节点的若干种方式**

2017年07月22日 16:50:49

阅读数：24302

**一、js获取子节点的方式**

**1.通过获取dom方式直接获取子节点**

其中test的父标签id的值，div为标签的名字。getElementsByTagName是一个方法。返回的是一个数组。在访问的时候要按数组的形式访问。

var a = document.getElementById("test").getElementsByTagName("div");

* 1
* 2

**2.通过childNodes获取子节点**

使用childNodes获取子节点的时候，childNodes返回的是子节点的集合，是一个数组的格式。他会把换行和空格也当成是节点信息。

var b =document.getElementById("test").childNodes;

* 1
* 2

为了不显示不必须的换行的空格，我们如果要使用childNodes就必须进行必要的过滤。通过正则表达式式取掉不必要的信息。下面是过滤掉

//去掉换行的空格

for(var i=0; i<b.length;i++){

if(b[i].nodeName == "#text" && !/\s/.test(b.nodeValue)){

document.getElementById("test").removeChild(b[i]);

}

}

//打印测试

for(var i=0;i<b.length;i++){

console.log(i+"---------")

console.log(b[i]);

}

//补充 document.getElementById("test").childElementCount; 可以直接获取长度 同length

**4.通过children来获取子节点**

利用children来获取子元素是最方便的，他也会返回出一个数组。对其获取子元素的访问只需按数组的访问形式即可。

var getFirstChild = document.getElementById("test").children[0];

* 1
* 2

**5.获取第一个子节点**

firstChild来获取第一个子元素，但是在有些情况下我们打印的时候会显示undefined，这是什么情况呢？？其实firstChild和childNodes是一样的，在浏览器解析的时候会把他当换行和空格一起解析，其实你获取的是第一个子节点，只是这个子节点是一个换行或者是一个空格而已。那么不要忘记和childNodes一样处理呀。

var getFirstChild = document.getElementById("test").firstChild;

**6.firstElementChild获取第一个子节点**

使用firstElementChild来获取第一个子元素的时候，这就没有firstChild的那种情况了。会获取到父元素第一个子元素的节点 这样就能直接显示出来文本信息了。他并不会匹配换行和空格信息。

var getFirstChild = document.getElementById("test").firstElementChild;

* 1
* 2

**7.获取最后一个子节点**

lastChild获取最后一个子节点的方式其实和firstChild是类似的。同样的lastElementChild和firstElementChild也是一样的。不再赘余。

var getLastChildA = document.getElementById("test").lastChild;

var getLastChildB = document.getElementById("test").lastElementChild;

* 1
* 2
* 3

**二、js获取父节点的方式**

**1.parentNode获取父节点**

获取的是当前元素的直接父元素。parentNode是w3c的标准。

var p = document.getElementById("test").parentNode;

* 1
* 2

**2.parentElement获取父节点**

parentElement和parentNode一样，只是parentElement是ie的标准。

var p1 = document.getElementById("test").parentElement;

* 1
* 2

**3.offsetParent获取所有父节点**

一看offset我们就知道是偏移量 其实这个是于位置有关的上下级 ，直接能够获取到所有父亲节点， 这个对应的值是body下的所有节点信息。

var p2 = document.getElementById("test").offsetParent;

* 1
* 2

**三、js获取兄弟节点的方式**

**1.通过获取父亲节点再获取子节点来获取兄弟节点**

var brother1 = document.getElementById("test").parentNode.children[1];

* 1
* 2

**2.获取上一个兄弟节点**

在获取前一个兄弟节点的时候可以使用previousSibling和previousElementSibling。他们的区别是previousSibling会匹配字符，包括换行和空格，而不是节点。previousElementSibling则直接匹配节点。

var brother2 = document.getElementById("test").previousElementSibling;

var brother3 = document.getElementById("test").previousSibling;

* 1
* 2
* 3

**3.获取下一个兄弟节点**

同previousSibling和previousElementSibling，nextSibling和nextElementSibling也是类似的。

var brother4 = document.getElementById("test").nextElementSibling;

var brother5 = document.getElementById("test").nextSibling;