范围

DOM2级在Document类型中定义了createRange()方法

var support = document.implementation.hasFeature("Range","2.0");  
var alsoSupport = typeof document.createRange === "function";//书上写的是var alsoSupport = （typeof document.createRange == "function"）

测试一下浏览器是否兼容，除了IE以外，alert一下都返回true，IE有自己实现的方法，一会写。

创建范围

var range1 = document.createRange();

与节点类似，新创建的范围也直接和创建它的文档关联在一起，不能用于其他文档。

每个范围由一个Range类型的实例表示，这个实例拥有许多方法和属性，以下属性提供了当前范围在文档中的位置信息。

startContainer:包含范围起点的节点（即选区中第一个节点的父节点）。

startOffset:范围在startContainer中起点的偏移量。如果startContainer是文本节点、注释节点或CDATA节点，那么startOffset就是范围起点之前跳过的字符数量。否则，startOffset就是范围中第一个子节点的索引（注意索引是从0开始算的）。

endContainer:包含范围终点的节点（即选区中最后一个节点的父节点）。

endOffset: 范围在endContainer中终点的偏移量。（与startOffset遵循同样的取值规则）。

commonAncestorContainer:startContainer和endContainer共同的祖先节点在文档树种位置最深的那个。（p是深，body是浅）

用DOM范围实现简单选择

要使用范围来选择文档的一部分，最简单的方法就是使用selectNode()或selectNodeContents()。这两个方法都接受一个参数，即一个DOM节点，然后使用该节点中的信息来填充范围。

selectNode()选择整个节点，包括其子节点。

selectNodeContexts()方法，只选择节点的子节点

<**body**>  
<**p** id=**"p1"**><**b**>**Hello**</**b**> **world!**</**p**>  
</**body**>

var range1 = document.createRange();  
var range2 = document.createRange();  
var p1 = document.getElementById("p1");  
var r1 = range1.selectNode(p1);  
var r2 = range2.selectNodeContents(p1);

range1包含了P元素及其所有的子元素，而range2包括了b元素及其子节点（Hello）以及文本节点world!。

在调用selectNode()时，startContainer，endContainer ，commonAncestorContainer:startContainer都等于传入节点的父节点，也就是整个例子中的document.body，而startOffset属性等于给定节点在其父节点childNodes集合中的索引（在和这个例子中是1，因为空格节点也是文本节点，它是0索引，所以p节点就是1索引），endOffset等于startOffset加1（个人理解，可以把endOffset的值理解为子节点的数量，那么一个空格文本节点+一个p节点就是2了）

在调用selectNodeContents()时，startContainer，endContainer ，commonAncestorContainer:startContainer都等于传入节点的父节点，也就是整个例子中的p元素，而startOffset属性始终等于0，因为b元素是p元素的第一个子节点，索引为0。endOffset等于子节点的数量(node.childNodes.length)，在这个例子中是2（一个元素节点b,一个文本节点world!,Hello是个孙元素，不算的）

alert(range1.startOffset);//1  
alert(range1.endOffset)//2  
alert(range2.startOffset);//0  
alert(range2.endOffset)//2