**DOM变化**

1. **Ducument类型变换**

Document.importNode()从一个文档中取得节点，然后将其导入到另一个文档。需要注意每个节点都有一个ownerDocument属性，表示所属的文档。直接调用appendChild时所属文档不一样会导致错误，而调用importNode会返回一个新的节点。

Var newNode = Document.importNode(oldNode, true); //传入true会复制其所有子节点

1. **Node类型变化**

Dom3引入比较节点的方法，isSameNode和isEqualNode方法

isSameNode，2节点引用同一个对象是，返回true

isEqualNode，2节点的属性类型都想同时，返回true

var div1 = document.createElement("div");

alert(div1.isSameNode(div1));//true

var div2 = document.createElement("div");

div1.className = "box";

div2.className = "box";

alert(div1.isEqualNode(div2));//true

**样式**

1. **访问元素样式**

* box.style.height通过特性可以访问可以修改，可以返回复合属性，但是只能得到标签上的style的属性。
* box.style.cssText只能得到标签上的style的属性，返回一个字符串，可以修改这个属性，修改之后等于重写这个属性
* 通过特性遍历标签内联样式

for (var i = 0, len = box.style.length; i < len; i++) {

var prop = box.style[i],

value = box.style.getPropertyValue(prop);

console.log(prop + ':' + value);

}

* box.style.removeProperty(),删除传入样式

1. **计算的样式**

var myStyle = document.defaultView.getComputedStyle(box, null);console.log(myStyle.padding);//能取得视图中显示的元素样式，第二个参数是伪类。可以取到计算后的样式（所以css叠加后的样式），复合样式有些浏览器返回underfined。在IE中用currentStyle代替defaultView.getComputedStyle（）

1. **操作样式表**

* 样式表包括link包含的样式表和style标签中定义的样式表。
* 得到样式表对象

function getStyle(ele) {

return ele.styleSheet || ele.sheet;

}

var myStyle = document.getElementsByTagName('link')[0];

console.log(getStyle(myStyle));

console.log(getStyle(myStyle) === document.styleSheets[0]);// true

得到style标签内的rule

var myStyle = document.styleSheets[1];

rules = myStyle.cssRules || myStyle.rules;

rule = rules[0]; //取得第一个rule

console.log(rule.selectorText)//选择器的名

console.log(rule.style.margin)//复合选择器

创建样式用insertRule()接受2个参数，第一个插入的样式，第二个插入的位置。

var sheet = document.styleSheets[0];

function insRule(sheet, selectorText, cssText, position){

if (sheet.insertRule) {//普通浏览器写法2个参数

sheet.insertRule(selectorText + '{' + cssText + '}',position);

} else if (sheet.addRule){//IE写法3个参数

sheet.addRule(selectorText,cssText,position)

}

}

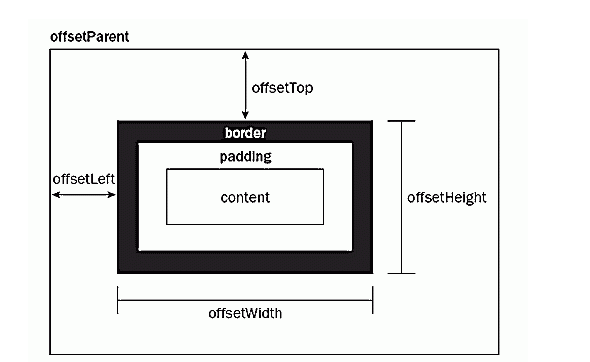
insRule(sheet, '#box', 'padding:10px;', 0);

删除规则

Sheet.deleteRule(position)//一般浏览器

Sheet.removeRule(position)//IE

1. **元素大小**

* ****
* 若要取得相对浏览器顶部的偏移。

function getLeft(element){

var left = element.offsetLeft;

var current = element.offsetParent;

while(current != null){

left += current.offsetLeft;

current = current.offsetParent;

}

return left;

}

* 客户区的大小。

得到浏览器可视区大小：

Function getVP（）{

If(document.compatMode == ‘BackCompat’){

Retuen{

Width: document.body.clientWidth

Height:document.body.clientHeight

} else {

Return{

Width: document. documentElememt.clientWidth

Height:document.documentElememt.clientHeight

}

}

}

}

* 在确定文档的总高度时，必须取得scrollWidth/clientWidth的最大值。
* Math.max(document,documentElement.scrollWidth,document. documentElement.clientWidth)//IE6混杂模式下要使用body

1. **遍历节点**

<html>

<head>

<title>dom中的NodeIterator对象示例</title>

<script>

function makelist()

{

var divnode = document.getElementById("div1");

var

FILTER= new Object();

FILTER.acceptNode = function(node)

{

return (node.tagName == "P") ? NodeFilter.FILTER\_REJECT : NodeFilter.FILTER\_ACCEPT;

}

var iterator = document.createNodeIterator(divnode, NodeFilter.SHOW\_ELEMENT, FILTER, false);

var oput = document.getElementById("textarea1");

var node = iterator.nextNode();

while(node)

{

oput.value += node.tagName +"\n";

node = iterator.nextNode();

}

}

</script>

</head>

<body>

<div id="div1">

<p>你好<b>读者!</b></p>

<ul>

<li>列表项一</li>

<li>列表项二</li>

<li>列表项三</li>

<li>列表项四</li>

</ul>

</div>

<textarea id="textarea1" rows="10"></textarea><br />

<input type="button" value="生成列表" onclick="makelist()" />

</body>

</html>

**createNodeIterator**第一个参数1、root:从树中的哪个节点开始搜索；2、whatToShow:一个数值代码，代表哪些节点需要搜索；3、filter:NodeFilter对象，用来决定需要忽略哪些节点；4、entityReferenceExpansion:布尔值，表示是否需要扩展实体引用；

1. **范围**

DeleteContents直接删除，而extractContents还会返回删除内容，

insertNode在开始插入，surroundContents在range的两端包裹[dome](dome1.html)

collapse折叠，传入true折叠到起点，传入false折叠到终点，collapse判断2个节点是否紧挨。

比较：compareBoundaryPoints()，传入2个参数，第一个Range.START\_TO\_START,Range.START\_TO\_END…..比较调用函数的范围和第二个参数传入的范围的起点或终点等，如果第一个点在第二个点前输出-1，相等输出0，之后输出1；

复制范围，cloneRange

清理范围，range.detach()//从文档中分离

Range=null;//解除引用

**小结**

* 可以用元素的style属性来确定修改行内样式。
* 可以使用getComputedStyle（）方法访问元素的计算样式，IE不支持但是又currentStyle属性返回相同信息。document.defaultView.getComputedStyle(box, null);是方法，而IE中的currentStyle是属性
* 可以通过Document.styleSheets访问样式表，无法访问到link？
* NodeIteration或者TreeWalker可以对DOM执行深度优先的遍历。
* NodeIteration只允许一步一步得移动，后者允许个方向上移动。
* 范围可以在删除文档某些部分同时保持文档的格式良好性。