**DOM:document object model 文档对象模型:里面提供了操作DOM元素的相关方法和属性**

document:文档对象

BOM:browser object model 浏览器对象模型

window:浏览器对象

var tabUl = **document.getElementById**("tabUl");

tabUl.style.border = "1px solid red";

document是他的上下文,**getElementById这个方法的上下文只能是document**

如果ID重复了，只获取第一个

在IE6、7下会把表单元素的name当做id，所有在项目中我们的name要和id区分开

在IE6、7下不区分ID的大小写

我们可以直接用元素的id代表这元素(仅作了解)

var oLis = tabUl.**getElementsByTagName**("li");

tabUl限定了我们的获取范围，我们把他称之为我们的上下文(context)

console.dir(oLis);//**HTMLCollection元素集合(类数组)**

var zhuFeng = document.**getElementsByName**("zhufeng");//通过元素的name属性的值获取一组元素

获取的是一个**NodeList节点集合(类数组)**

**getElementsByName的上下文只能是document**

**在所有的IE浏览器下，只对表单元素起作用**

var a = **tabUl.getElementsByClassName**("w100");//通过元素的class属性值获取一组元素

HTMLCollection 元素集合(类数组)

**在IE6~8下不兼容**

**document.documentElement** 获取整个HTML

**document.body** 获取整个body

**获取当前屏幕的宽度和高度**

var cw = **document.documentElement.clientWidth || document.body.clientWidth;**

var ch = **document.documentElement.clientHeight || document.body.clientHeight;**

移动端经常用的两个方法：在IE6~8下不兼容

**document.querySelector();**//根据选择器获取一个元素

**document.querySelectorAll();**//根据选择器获取多个元素

document.querySelector("#tabUl");

document.querySelectorAll("li");

document.querySelectorAll(".w100");

**节点：Node 元素标签、文字、注释等都是节点**

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

tabUl.**childNodes;**//获取元素下的所有子节点

tabUl.**children;**//获取元素下的所有元素子节点

tabUl.**parentNode;**//获取元素的父亲节点

tabUl.**previousSibling;**//获取上一个哥哥节点

tabUl.**nextSibling;**//获取下一个弟弟节点

tabUl.**firstChild;**//获取所有子节点中的第一个节点

tabUl.**lastChild;**//获取所有子节点中的最后一个节点

还有一些不兼容的

tabUl.**previousElementSibling;**//获取上一个哥哥元素节点

tabUl.**nextElementSibling;**//获取下一个弟弟元素节点

tabUl.**firstElementChild;**//获取所有元素子节点中的第一个元素节点

tabUl.**lastElementChild;**//获取所有元素子节点中的最后一个元素节点

节点 nodeType(节点类型) nodeName(节点名称) nodeValue

元素节点： 1 大写的标签名 null

文本节点： 3 #text 文本内容

注释节点： 8 #comment 注释内容

document: 9 #document null

**在标准浏览器中(除IE6~8)会把空格、Enter换行都当做文本节点**

我想获取它的上一个哥哥元素节点，发现直接用previousSibling不能直接找到，需要一级级的查找，但是具体查几次不知道,那我们如何的处理?

首先获取一个，判断是不是元素节点，不是的话基于现在的基础在往上找，一直找到元素节点为止(nodeType===1)

**获取上一个哥哥元素节点**

function getPre(curEle) {

var pre = curEle.previousSibling;

while (pre && pre.nodeType !== 1) {

//只要找得到并且找到的nodeType不是1，我们就基于原有的基础继续往上找，直到找到为止，如果找到头了都有没有返回的就是null

pre = pre.previousSibling;

}

return pre;

}

**获取下一个弟弟元素节点**

function getNext(curEle) {

var next = curEle.nextSibling;

while (next && next.nodeType !== 1) {

next = next.nextSibling;

}

return next;

}

增加

**createElement**动态创建元素的标签

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

oDiv.id = "div1";

oDiv.className = "cc";

oDiv.style.backgroundColor = "yellow";

oDiv.innerHTML = "我是动态创建的#div1元素";

//把动态创建的元素添加到我们的页面(body)中

//**appendChild**将元素添加到指定容器的末尾位置

document.body.appendChild(oDiv);

var oDiv2 = document.createElement("div");

oDiv2.id = "div2";

oDiv2.className = "cc";

oDiv2.style.backgroundColor = "red";

//把div2添加到div1之前

//**insertBefore**在指定的容器中，添加到某一个元素之前,新元素newEle ,老元素oldEle

//**oldEle.parentNode.insertBefore(newEle,oldEle);**

document.body.insertBefore(oDiv2, oDiv);

document.body.**removeChild**(oDiv);//删除

**设置自定义属性的两种办法：**

//**oDiv.zhuFeng=i;**

//**setAttribute**可以把这个属性映射到我们的html结构中

//**setAttribute设置class属性不兼容**

oDiv.**setAttribute**("zhuFeng", 100);

oDiv.**getAttribute**("zhuFeng");

oDiv.**removeAttribute**("zhuFeng");

**replaceChild()**//替换

oDiv.**cloneNode()**;//克隆