[DOM+面向对象面试题](http://www.cnblogs.com/liningstyle/p/6646475.html)

1.dom常见的节点类型---------------------------解彬1510-B  
1.nodeType------节点类型，元素节点是1，文本节点是3.  
2.firstChild----该元素节点包含的第一个子节点  
3.lastChild---该元素节点包含的最后一个子节点  
4.nextSibling----该节点的后一个兄弟节点  
5.previousSibling---该节点的前一个兄弟节点  
6.childNodes----子节点列表，可以通过node.childNodes[index]来获取子节点  
7.nodeName----节点名称，对于元素节点，返回tagName，对于文本返回#text

2.dom方法大全---------------------------解彬1510-B  
  
1.document.createElement(TagName) --创建指定元素  
2.\_dom.appendChild(子元素)----添加新节点  
3.\_dom.getAttribute(属性名)----返回元素的属性  
4.\_dom.setAttribute(属性名，属性值)----设置元素属性  
5.\_dom.removeAtrribute(属性名)----删除属性  
6.\_dom.removeChild(子元素)----删除节点  
7.\_dom.replaceChild(新元素，子元素)----替换节点  
8.\_dom.hasChildNodes()-----是否有子元素  
3.什么是dom？---------------------------解彬1510-B

dom其实就是把一个html或者xml等符合w3c标准的文档内容模拟成一个java对象，这样才能给java或者js来操作。  
；  
4.写一个获取非行间样式的函数---------------------------解彬1510-B  
function getStyle(obj,attr,value)  
{  
if(!value)  
{  
if(obj.currentStyle)//ie  
{  
return obj.currentStyle(attr);  
}  
else{//标准浏览器  
obj.getComputedStyle(attr,false);  
}  
}   
else  
{  
obj.style[attr] = value;  
}  
}

5.阻止冒泡函数---------------------------解彬1510-B  
function stopPropagation(e) {   
e = e || window.event;   
if(e.stopPropagation) { //W3C阻止冒泡方法   
e.stopPropagation();   
} else {   
e.cancelBubble = true; //IE阻止冒泡方法   
}   
}   
document.getElementById('need\_hide').onclick = function(e) {   
stopPropagation(e);   
}  
6.JS的继承性---------------------------解彬1510-B  
window.color = 'red';  
var o = {color: 'blue'};  
function sayColor(){  
alert(this.color);  
}  
//1、this的指向  
//2、call的用法  
sayColor(); //red  
sayColor.call(this); //red this指向的是window对象  
sayColor.call(window); //red  
sayColor.call(o); //blue

7.document load 和document ready的区别---------------------------解彬1510-B  
Document.onload 是在结构和样式加载完才执行js  
Document.ready原生种没有这个方法，jquery中有 $().ready(function)  
DOM Ready是在dom加载完成后就可以直接对dom进 行操作，比如一张图片只要<img>标签完成，不用等这 个图片加载完成，  
就可以设置这个图片的宽高的属性或样 式等；Dom Load是在整个document文档（包括了加载图片及 其他信息）加载完成后就  
可以对dom直接进行操作，比如一张图片要等这个图标加载完成后才能设置图片的宽 高的属性或样式等；比如一个图片的浏览效果，  
通常如果 图片尺寸很大的情况下，为了阻止图片把页面撑开通常会 限定图片的宽高或给高度，如果是单张图片或者是多张规 格比较  
同意的图片下我们可以直接在<img>上价格宽度 或者高度的属性  
<img src=“img.jpg” alt=“码头的大 照片”width=“100” height=“90”>，比如（推荐 ）或者可以在css样式中加宽度或者  
高度的属性。但是如 果这些张规格比例不统一的图片要浏览呢？那就有问题，你设置宽高可能造成图片的严重失真。在ie6之后ie7，  
ie8还有其他主流浏览器支持css2.1中的min-width，min-height，max-width，这样我们就可以用min-width，min-height，  
max-width解决这些问题，但是ie6除非是抛弃性能问题有css表达式这个时候ie的做好解决方案就是Dom Ready而不是Dom Load,  
因为通常大图片加载的时候会一点一点的加载。这个在尺寸大，字节多，网速慢的时候表现的非常明显，用Dom Load，通常是  
先把页面撑开，加载完成后再把图片重设宽高，图片加载多少时间，这个一面就会撑开多久，  
用户会非常难受。

8、简述列举文档对象模型DOM里document的常用的查找访问节点的方法并做简单说明---------------------------解彬1510-B  
Document.getElementById 根据元素id查找元素  
Document.getElementByName 根据元素name查找元素  
Document.getElementTagName 根据指定的元素名查找元素

9、有哪项方式可以对一个DOM设置它的CSS样式？　 -----许瑞瑞1510B  
外部样式表，引入一个外部css文件  
内部样式表，将css代码放在 <head> 标签内部  
内联样式，将css样式直接定义在 HTML 元素内部

10、当一个DOM节点被点击时候，我们希望能够执行一个函数，应该怎么做？ -----许瑞瑞1510B  
直接在DOM里绑定事件：<div onclick=”test()”></div>  
在JS里通过onclick绑定：xxx.onclick = test  
通过事件添加进行绑定：addEventListener(xxx, ‘click’, test)  
那么问题来了，Javascript的事件流模型都有什么？  
“事件冒泡”：事件开始由最具体的元素接受，然后逐级向上传播  
“事件捕捉”：事件由最不具体的节点先接收，然后逐级向下，一直到最具体的  
“DOM事件流”：三个阶段：事件捕捉，目标阶段，事件冒泡

11、DOM操作——怎样添加、移除、移动、复制、创建和查找节点。 -----许瑞瑞1510B  
1.) 创建新节点  
createDocumentFragment() // 创建一个DOM片段  
createElement() // 创建一个具体的元素  
createTextNode() // 创建一个文本节点  
2.) 添加、移除、替换、插入  
appendChild()  
removeChild()  
replaceChild()  
insertBefore() // 在已有的子节点前插入一个新的子节点  
3.) 查找  
getElementsByTagName() // 通过标签名称  
getElementsByName() // 通过元素的Name属性的值(IE容错能力较强，会得到一个数组，其中包括id等于name值的)  
getElementById() // 通过元素Id，唯一性

12、CSS中可以通过哪些属性定义，使得一个DOM元素不显示在浏览器可视范围内？ -----许瑞瑞1510B  
最基本的：  
设置display属性为none，或者设置visibility属性为hidden  
技巧性：  
设置宽高为0，设置透明度为0，设置z-index位置在-1000

13.如何显示/隐藏一个dom元素？请用原生的JavaScript方法实现 -----许瑞瑞1510B  
dom.style.display=”none”;  
dom.style.display=””;

14、javascript面向对象中继承实现 ------1510B 杨姣姣

javascript面向对象中的继承实现一般都使用到了构造函数和Prototype原型链，简单的代码如下：

function Animal(name) {  
this.name = name;  
}  
Animal.prototype.getName = function() {alert(this.name)}  
function Dog() {};  
Dog.prototype = new Animal("Buddy");  
Dog.prototype.constructor = Dog;  
var dog = new Dog();

15、构造函数使用this的典型应用 ------1510B 杨姣姣

function Animal(name, color) {  
this.name = name;  
this.color = color;  
}

16、如何扩展prototype? ------1510B 杨姣姣

A.prototype.B = C;

A是某个构造函数的名字

B是这个构造函数的属性

C是想要定义的属性的值

17.、dom的节点类型？ ------1510B 杨姣姣  
元素节点、属性节点 文本节点 注释节点 document

18、对象是怎么样创建的?————————————陈蓉蓉 1510-B  
答：1.用var obj={}就可以创建一个对象   
2.用new Object()也可以创建一个对象  
19、JS创建对象的方法： ————————————陈蓉蓉 1510-B  
　　（a）工厂模式：用函数来封装以特定接口创建对象的细节。   
　　　　　　function createPerson(name, age, job){   
　　　　　　　　　　var o = new Object();   
　　　　　　　　　　o.name = name;   
　　　　　　　　　　o.age = age;   
　　　　　　　　　　o.job = job;   
　　　　　　　　　　o.sayName = function(){   
　　　　　　　　　　alert(this.name);   
　　　　　　　　　　};   
　　　　　　return o;   
　　　　　　}   
　　　　var person1 = createPerson(“Nicholas”, 29, “Software Engineer”);   
　　　　var person2 = createPerson(“Greg”, 27, “Doctor”);   
　　　　缺点：工厂模式虽然解决了创建多个相识对象的问题，但却没有解决对象识别的问题（即怎样知道一个对象的类型）。   
（b）构造函数模式：ECMAScript中的构造函数可以用来创建特定类型的对象。可以创建自定义的构造函数，从而定义自定义对象类型的属性和方法。   
　　　　　　function Person(name, age, job){   
　　　　　　　　this.name = name;   
　　　　　　　　this.age = age;   
　　　　　　　　this.job = job;   
　　　　　　　　this.sayName = function(){   
　　　　　　　　alert(this.name);   
　　　　　　　　};   
　　　　　　}   
　　　　　　var person1 = new Person(“Nicholas”, 29, “Software Engineer”);   
　　　　　　var person2 = new Person(“Greg”, 27, “Doctor”);   
　　　　缺点：使用构造函数的主要问题，就是每个方法都要在每个实例上重新创建一遍。不要忘了——ECMAScript中的函数是对象，因此每定义一个函数， 就是实例化一个对象。   
（c）原型模式：我们创建的每个函数都有一个prototype（原型）属性，这个属性是一个指针，指向一个对象，而这个对象的用途是包含可以由特定类型   
　　　　的所有实例共享的属性和方法。使用原型对象的好处是可以让所有对象共享它包含的属性和方法   
　　　　　　function Person(){   
　　　　　　}   
　　　　　　Person.prototype.name = “Nicholas”;   
　　　　　　Person.prototype.age = 29;   
　　　　　　Person.prototype.job = “Software Engineer”;   
　　　　　　Person.prototype.sayName = function(){   
　　　　　　　　alert(this.name);   
　　　　　　};   
　　　　　　var person1 = new Person();   
　　　　　　person1.sayName(); //”Nicholas”   
　　　　　　var person2 = new Person();   
　　　　　　person2.sayName(); //”Nicholas”   
　　　　　　alert(person1.sayName == person2.sayName); //true   
　　　　缺点：原型中所有属性是被很多实例共享的，这种共享对于函数非常合适。但是对于引用类型值的属性来说，问题就比较突出了。  
（d）组合使用构造函数模式和原型模式：创建自定义类型的最常见方式，就是使用组合使用构造函数模式和原型模式。构造函数模式用于定义实例属性，   
　　　　　　而原型模式用于定义方法和共享的属性。   
　　　　　　function Person(name, age, job){   
　　　　　　　　this.name = name;   
　　　　　　　　this.age = age;   
　　　　　　　　this.job = job;   
　　　　　　　　this.friends = [“Shelby”, “Court”];   
　　　　　　}   
　　　　　　Person.prototype = {   
　　　　　　　　constructor: Person,   
　　　　　　　　sayName : function () {   
　　　　　　　　alert(this.name);   
　　　　　　　　}   
　　　　　　};   
　　　　　　var person1 = new Person(“Nicholas”, 29, “Software Engineer”);   
　　　　　　var person2 = new Person(“Greg”, 27, “Doctor”);   
　　　　　　person1.friends.push(“Van”);   
　　　　　　alert(person1.friends); //”Shelby,Court,Van”   
　　　　　　alert(person2.friends); //”Shelby,Court”   
　　　　　　alert(person1.friends === person2.friends); //false   
　　　　　　alert(person1.sayName === person2.sayName); //true

20、简述创建函数的几种方式————————————陈蓉蓉 1510-B  
答：第一种（函数声明）：   
function sum1(num1,num2){  
return num1+num2;  
}  
第二种（函数表达式）：  
var sum2 = function(num1,num2){  
return num1+num2;  
}  
匿名函数：  
function(){}:只能自己执行自己  
第三种（函数对象方式）：  
var sum3 = new Function("num1","num2","return num1+num2");

21.DOM中有几种类型的节点-----------------------------------王辰艳1510-B  
整个文档是一个文档节点(Document)  
每个 HTML 标签是一个元素节点(Element)  
每一个 HTML 属性是一个属性节点(Attribute)   
包含在 HTML 元素中的文本是文本节点(Text)

22.Dom中target属性：-----------------------------------王辰艳1510-B  
在需要通过一个函数处理多个事件时，可以使用type属性。例如：  
var btn =document.getElementByIdx\_x("myBtn");  
var handler = function (event){ //定义了一个handler函数  
switch (event.type){ //用于处理3个事件  
case "click": //1.当鼠标点击时：弹出Clicked  
alert("Clicked");  
break;  
case "mouseover":  
event.target.style.backgroundColor="red"; // 当鼠标移入时，变为红色  
break;   
case "mouseout":   
event.target.style.backgroundColor=""; //当鼠标移出时，变为默认颜色  
break;   
}  
};  
btn.onclick = handler;  
btn.onmouseover = handler;  
btn.onmouseout = handler;  
这里通过检测event.type属性，让函数能够确定发生了什么事件，并执行相应操作。

23.bind(), live(), delegate()的区别-----------------------------------王辰艳1510-B  
bind： 绑定事件，对新添加的事件不起作用，方法用于将一个处理程序附加到每个匹配元素的事件上并返回jQuery对象。  
live： 方法将一个事件处理程序附加到与当前选择器匹配的所有元素（包含现有的或将来添加的）的指定事件上并返回jQuery对象。  
delegate： 方法基于一组特定的根元素将处理程序附加到匹配选择器的所有元素（现有的或将来的）的一个或多个事件上。

24.JS中的简单继承 call方法！-----------------------------------王辰艳1510-B  
//顶一个父母类，注意：类名都是首字母大写的哦！  
function Parent(name, money){  
this.name = name;  
this.money = money;  
this.info = function(){  
alert('姓名： '+this.name+' 钱： '+ this.money);  
}  
}  
//定义孩子类  
function Children(name){  
Parent.call(this, name); //继承 姓名属性，不要钱。   
this.info = function(){  
alert('姓名： '+this.name);  
}  
}  
//实例化类  
var per = new Parent('parent', 800000000000);  
var chi = new Children('child');  
per.info();  
chi.info();

25、call和apply的区别 -----------------------------------王辰艳1510-B  
call和apply相同点：  
都是为了用一个本不属于一个对象的方法，让这个对象去执行  
toString.call([],1,2,3)  
toString.apply([],[1,2,3])  
Object.call(this,obj1,obj2,obj3)  
Object.apply(this,arguments)  
  
区别：call第二个参数开始接受一个参数列表  
apply第二个参数开始接受一个参数数组

26、new操作符具体干了什么呢?

1、创建一个空对象，并且 this 变量引用该对象，同时还继承了该函数的原型。  
2、属性和方法被加入到 this 引用的对象中。  
3、新创建的对象由 this 所引用，并且最后隐式的返回 this 。

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
1510A李鹏亮、孟科宇、马金良小组的dom面试题

···········1510A孟科宇············  
1. 要动态改变层中内容可以使用的方法有（AB ）  
a)innerHTML  
b)innerText  
c)通过设置层的隐藏和显示来实现  
d)通过设置层的样式属性的display属性

2. 当按键盘A时，使用onKeyDown事件打印event.keyCode的结果是（A ）  
a)65  
b)13  
c)97  
d)37  
3.documen.write和 innerHTML的区别  
document.write只能重绘整个页面  
innerHTML可以重绘页面的一部分  
4. 下列哪一个选项不属于document对象的方法?（D ）  
a)focus()  
b)getElementById()  
c)getElementsByName()  
d)bgColor()  
5、当鼠标离开页面中的所有p标签时让它折叠起来，实现的方法有（ a ）  
A、$(‘p’). mouseleave(function(){$(this).slideUp()})  
B、$(‘p’). mouseleave(function(){$(this).slideDown()})  
C、$(‘p’). mouseleave(function(){this.slideUp()})  
D、$(‘p’). mouseover(function(){$(this).slideUp()})  
6.在jQuery中，函数（c）能够实现元素显示和隐藏的互换。  
hide()  
show()  
toggle()  
fade()  
7.在jQuery中，有以下代码：  
$(“.btn”).click(funtion(){  
var json=[  
{“S\_name”: “周颜”,“S\_sex”: “男”}，  
{“S\_name”: “周颖”,“S\_sex”: “女”}  
];  
$.each(json,function(index,s){  
alert(s.S\_name+ “,”+s.S\_sex)  
})  
})  
以下说法正确的是（C）  
此代码不会正常运行  
语句1会被执行1次  
语句1会被执行2次  
$.each()函数的用法有误  
8.下面哪一个是用来追加到指定元素的末尾?(c)  
A.insertAfter()  
B.Append()  
C.appendTo()  
After()

9.在jQuery中指定一个类，如果存在就执行删除功能  
，如果不存在就执行添加功能，下面哪一个是可以直接完成该功能的？(c)  
A.removeClass()  
B.deleteClass()  
C.toggleClass(class)  
addClass()  
10.在一个表单中，如果想要给输入框添加一个输入验证，可以用下面的哪个事件实现？(D)  
A、hover(over,out)  
B、keypress(fn)  
C、change()  
D.change(fn)  
11. 你如何利用jQuery来向一个元素中添加和移除CSS类? (答案)

通过利用 addClass() 和 removeClass() 这两个 jQuery 方法。动态的改变元素的class属性可以很简单例如. 使用类“.active"来标记它们的未激活和激活状态，等等.

12. jQuery中 detach() 和 remove() 方法的区别是什么? (答案)

尽管 detach() 和 remove() 方法都被用来移除一个DOM元素, 两者之间的主要不同在于 detach() 会保持对过去被解除元素的跟踪, 因此它可以被取消解除, 而 remove() 方法则会保持过去被移除对象的引用. 你也还可以看看 用来向DOM中添加元素的 appendTo() 方法.

13. IE和DOM事件流的区别  
1.执行顺序不一样、  
2.参数不一样  
3.事件加不加on  
4.this指向问题

14.添加 删除 替换 插入到某个接点的方法  
obj.appendChidl()  
obj.innersetBefore  
obj.replaceChild  
obj.removeChild

15. 要动态改变层中内容可以使用的方法有（AB ）  
a)innerHTML  
b)innerText  
c)通过设置层的隐藏和显示来实现  
d)通过设置层的样式属性的display属性  
16. 你是如何将一个 HTML 元素添加到 DOM 树中的？（答案如下）  
你可以用 jQuery 方法 appendTo() 将一个 HTML 元素添加到 DOM 树中。这是 jQuery 提供的众多操控 DOM 的方法中的一个。你可以通过 appendTo() 方法在指定的 DOM 元素末尾添加一个现存的元素或者一个新的 HTML 元素。  
17. 怎么在元素前添加元素。  
insertBefore(node,myNode);   
18. 怎么得到一个该元素的HTML标签内容（包括自身），  
object.outerHTML

19.offsetWidth /offsetHeight返回数值包括哪几部分。

x content  
o border  
o padding  
o width / height  
x margin

20.clinetWidth/ clientHeight返回数值包括哪几部分。

x content  
x margin  
o height/ width  
o padding  
x border  
21.bom和dom的区别  
BOM是浏览器对象模型，用来获取或设置浏览器的属性、行为，例如：新建窗口、获取屏幕分辨率、浏览器版本号等。  
DOM是文档对象模型，用来获取或设置文档中标签的属性，例如获取或者设置input表单的value值。

DOM是Document Object Model文档对象模型的缩写。根据W3C DOM规范（http://www.w3.org/DOM/），DOM是一种与浏览器，平台，语言无关的接口，使得你可以访问页面其他的标准组件。简单理解，DOM解决了Netscape的Javascript和Microsoft的Jscript之间的冲突，给予web设计师和开发者一个标准的方法，让他们来访问他们站点中的数据、脚本和表现层对像。  
DOM是以层次结构组织的节点或信息片断的集合。这个层次结构允许开发人员在树中导航寻找特定信息。分析该结构通常需要加载整个文档和构造层次结构，然后才能做任何工作。由于它是基于信息层次的，因而 DOM 被认为是基于树或基于对象的。  
对于特别大的文档，解析和加载整个文档可能很慢且很耗资源，因此使用其他手段来处理这样的数据会更好。这些基于事件的模型，比如 Simple API for XML（SAX），适用于处理数据流，即随着数据的流动而依次处理数据。基于事件的 API 消除了在内存中构造树的需要，但是却不允许开发人员实际更改原始文档中的数据。  
SAX等模型会从头至尾的顺序解析整个XML文档，当遇到一个节点的开始或或者结束时，会产生一个时间，程序员可以注册事件处理函数以对各个节点进行处理。  
另一方面，DOM 还提供了一个 API，允许开发人员添加、编辑、移动或删除树中任意位置的节点，从而创建一个引用程序。这个模型就是基于树型数据结构的，他在使用之前必须加载整个文档或者至少是结构良好的文档片段  
解析器是一个软件应用程序，设计用于分析文档（这里是指 XML 文件），以及做一些特定于该信息的事情。在诸如 SAX 这样基于事件的 API 中，解析器将向某种监听器发送事件。在诸如 DOM 这样基于树的 API 中，解析器将在内存中构造一颗数据树。

···········1510A马金梁：············  
Document－－最顶层的节点，所有的其他节点都是附属于它的。  
Element－－表示起始标签和结束标签之间的内容，例如<tag></tab>或者<tag/>。这是唯一可以同时包含特性和子节点的节点类型。  
　　一、获取节点：  
1.document：

getElementById：语法：document.getElementById(元素id) 功能：通过元素id获取节点。

getElementByName：语法：document.getElementByName(元素name属性) 功能：通过元素name属性获取节点。

getElementByTagName：语法：document.getElementByTagName(元素标签) 功能：通过元素标签获取节点。

2.节点指针：

firstChild ： 语法：父节点.firstChild 功能：获取元素的首个子节点。

lastChild ： 语法：父节点.lastChild 功能：获取元素的最后一个子节点。

childNodes ： 语法：父节点.childNodes 功能：获取元素的子节点列表。

previousSibling：语法：兄弟节点.previousSibling 功能：获取已知节点的前一个节点。

nextSibling ：语法：兄弟节点.nextSibling 功能：获取已知节点的后一个节点。

parentNode ：语法：子节点.parentNode 功能：获取已知节点的父节点。

　　二、建--新建DOM节点

1.创建节点：

creatElement：语法: document.creatElement(元素标签) 功能：创建元素节点。

creatAttribute：语法：document.creatAttribute(元素属性) 功能：创建属性节点。

creatTextNode：语法：document.creatTextNode(文本内容) 功能：创建文本节点。

　　　三、增

　　　　1、element.appendChild() 为元素添加一个新的子元素

　　　　2、insertBefore 任意位置插入元素  
前面放的是父级   
可以跟两个参数 第一个参数 是要插入的元素 第二个参数 要要插入位置元素的前面  
　　　  
　 　四、删

　　　　　element.removeChild() 删除一个子元素

　　五、改

　　　　1.cloneNode：语法：需要被复制的节点.cloneNode(true/false) 功能：创建指定节点的副本参数：true 复制当前节点及其所有子节点，false 仅复制当前节点

　　　　2、element.isEqualNode() 检查两个元素是否相等

3、替换节点：replaceChild：语法：replaceChild(要插入的新元素,将被替换的老元素) 功能：将某个子节点替换为另一个。

　　属性操作

　　　　1、获取属性：   
getAttribute：语法：元素节点.getAttrbute(元素属性名) 功能：获取元素节点中指定属性的属性值。

2、设置属性：   
setAttribute：语法：元素节点.setAttribute(属性名，属性值) 功能：创建或者改变元素节点的属性。

3、删除属性：

removeAttribute：语法：元素节点.removeAttribute(属性名) 功能：删除元素中的指定属性。

···············1510A李鹏亮··············  
js中的DOM操作汇总：

一、DOM创建

DOM节点（Node）通常对应于一个标签，一个文本，或者一个HTML属性。DOM节点有一个nodeType属性用来表示当前元素的类型，它是一个整数：

Element，元素  
Attribute，属性  
Text，文本

DOM节点创建最常用的便是document.createElement和document.createTextNode方法：

var node1 = document.createElement('div');  
var node2 = document.createTextNode('hello world!');

二、DOM查询  
复制代码

// 返回当前文档中第一个类名为 "myclass" 的元素  
var el = document.querySelector(".myclass");

// 返回一个文档中所有的class为"note"或者 "alert"的div元素  
var els = document.querySelectorAll("div.note, div.alert");

// 获取元素  
var el = document.getElementById('xxx');  
var els = document.getElementsByClassName('highlight');  
var els = document.getElementsByTagName('td');  
Element也提供了很多相对于元素的DOM导航方法：

// 获取父元素、父节点  
var parent = ele.parentElement;  
var parent = ele.parentNode;//只读，没有兼容性问题  
var offsetParent=ele.offsetParent;//只读，找到最近的有定位的父节点。  
　　　　　　　　　　　　　　　　　　　　　//没有定位父级时，默认是body;但在IE7以下，如果当前元素没有定位属性，返回body，如果有，返回HTML;  
　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　　 //如果当前元素某个父级触发了haslayout，则返回触发了haslayout这个元素。

// 获取子节点，子节点可以是任何一种节点，可以通过nodeType来判断  
var nodes = ele.children;//标准下、非标准下都只含元素类型，但对待非法嵌套的子节点，处理方式与childNodes一致。   
var nodes = ele.childNodes;//非标准下：只包含元素类型，不会包含非法嵌套的子节点。  
　　　　　　　　　　　　　　　　//标准下：包含元素和文本类型，会包含非法嵌套的子节点。

//获取元素属性列表  
var attr = ele.attributes;

// 查询子元素  
var els = ele.getElementsByTagName('td');  
var els = ele.getElementsByClassName('highlight');

// 当前元素的第一个/最后一个子元素节点  
var el = ele.firstChild;//对待标准和非标准模式，如childNods  
var el = ele.lastChild;  
var el = ele.firstElementChild;//非标准不支持  
var el = ele.lastElementChild;  
// 下一个/上一个兄弟元素节点  
var el = ele.nextSibling;  
var el = ele.previousSibling;  
var el = ele.nextElementSibling;  
var el = ele.previousElementSibling;

复制代码

兼容的获取第一个子元素节点方法：

var first=ele.firstElementChild||ele.children[0];

三、DOM更改  
复制代码

// 添加、删除子元素  
ele.appendChild(el);  
ele.removeChild(el);

// 替换子元素  
ele.replaceChild(el1, el2);

// 插入子元素  
parentElement.insertBefore(newElement, referenceElement);

//克隆元素  
ele.cloneNode(true) //该参数指示被复制的节点是否包括原节点的所有属性和子节点

复制代码

四、属性操作  
复制代码

// 获取一个{name, value}的数组  
var attrs = el.attributes;

// 获取、设置属性  
var c = el.getAttribute('class');  
el.setAttribute('class', 'highlight');

// 判断、移除属性  
el.hasAttribute('class');  
el.removeAttribute('class');

// 是否有属性设置  
el.hasAttributes();

复制代码

js原生dom操作第二种归纳：  
1)创建节点：除了可以使用createElement创建元素，也可以使用createTextNode创建文本节点.  
  
document.body指向的是<body>元素;document.documentElement则指向<html>元素  
  
//创建节点  
var createNode = document.createElement("div");  
var createTextNode = document.createTextNode("hello world");  
createNode.appendChild(createTextNode);  
document.body.appendChild(createNode);  
document.documentElement.appendChild(createNode);  
2)插入节点：可以使用appendChild，insertBefore，insertBefore接收两个参数，第一个是插入的节点，第二个是参照节点，如insertBefore(a,b)，则a会插入在b的前面  
  
  
  
//插入节点  
var createNode = document.createElement("div");  
var createTextNode = document.createTextNode("hello world");  
createNode.appendChild(createTextNode);  
var div1 = document.getElementById("div1");  
document.body.insertBefore(createNode,div1);  
  
  
3)替换和删除元素，从replaceChild和removeChild的字面意思看，就是删除子节点，因此调用者，需要包含子节点div1，不然调用会报错。返回的节点是替换的或删除的元素，被替换/删除的元素仍然存在，但document中已经没有他们的位置了。  
  
  
  
//替换元素  
var replaceChild = document.body.replaceChild(createNode,div1);  
  
  
//删除元素  
var removeChild = document.body.removeChild(div1);  
4)节点的属性：  
  
firstChild:第一个子节点  
  
lastChild:最后一个子节点  
  
childNodes:子节点集合，获取其中子节点可以someNode.childNodes[index]或者someNode.childNodes.item(index)  
  
nextSibling:下一个兄弟节点  
  
previousSibling：上一个兄弟节点  
  
parentNode：父节点  
  
<ul id="ul"><li>sdsssssss</li><li>qqqq</li><li>wwww</li><li>eeee</li></ul>  
  
  
复制代码  
//节点属性  
var ul = document.getElementById("ul");  
var firstChild = ul.firstChild;  
console.log(firstChild.innerHTML);  
var lastChild = ul.lastChild;  
console.log(lastChild.innerHTML);  
var length = ul.childNodes.length;  
console.log(length);  
  
var secondChild = ul.childNodes.item(1);  
console.log(secondChild.innerHTML);  
  
var forthChild = ul.childNodes.item(2).nextSibling;  
console.log(forthChild.innerHTML);  
  
var thridChild = forthChild.previousSibling;  
console.log(thridChild.innerHTML);  
  
var parentNode = forthChild.parentNode;  
console.log(parentNode.innerHTML);  
复制代码  
  
  
5) 文档片段，好处在于减少dom的渲染次数，可以优化性能。  
  
复制代码  
//文本片段  
var fragment = document.createDocumentFragment();  
var ul = document.getElementById("ul");  
var li = null;  
for (var i = 4; i >= 0; i--) {  
li = document.createElement("li");  
li.appendChild(document.createTextNode("item "+i));  
fragment.appendChild(li);  
}  
ul.appendChild(fragment);  
复制代码  
  
  
6）克隆元素  
  
someNode.cloneNode(true):深度克隆，会复制节点及整个子节点  
  
someNode.cloneNode(false):浅克隆，会复制节点，但不复制子节点   
  
//克隆  
var clone = ul.cloneNode(true);  
document.body.appendChild(clone);  
  
  
  
  
注意：  
  
1、childNodes.length存在跨浏览器的问题   
  
可以看到有关列表的html片段没有用   
  
  
  
<ul id="ul">  
<li>sdsssssss</li>  
<li>qqqq</li>  
<li>wwww</li>  
<li>eeee</li>  
</ul>  
这种书写格式而是使用没有换行的格式书写，是因为在不同的浏览器中，获取ul.childNodes.length的结果有差异：  
  
在ie中，ul.childNodes.length不会计算li之间的换行空格，从而得到数值为4  
  
在ff、chrome,safari中，会有包含li之间的空白符的5个文本节点，因此ul.childNodes.length为9  
  
若要解决跨浏览器问题，可以将li之间的换行去掉，改成一行书写格式。   
  
  
  
2、cloneNode存在跨浏览器的问题  
  
在IE中，通过cloneNode方法复制的元素，会复制事件处理程序，比如，var b = a.cloneNode(true).若a存在click,mouseover等事件监听，则b也会拥有这些事件监听。  
  
在ff,chrome,safari中，通过cloneNode方法复制的元素，只会复制特性，其他一切都不会复制  
  
因此，若要解决跨浏览器问题，在复制前，最好先移除事件处理程序。