DOM

Document Object Model

DOM定义了表示和修改文档所需的方法。DOM对象即为宿主对象，由浏览器厂商定义，用来操作html和xml功能的一类对象的集合。也有人称DOM是对HTML以及xml的标准编程接口。

1. 方法集合的统称

2. 用来操作html和xml

xml -->xhtml -->html

xml与HTML最大的不同：

xml的标签可以自定义

1. document代表整个文档

2. document.getElementById('');在ie8以下不区分大小写，而且返回匹配name属性的元素

3. getElementsByTagName('')根据标签名选择，类数组展示所有该标签，（最常用）

4. getElementsByCalssName('')存在兼容性问题ie8及以下没有此方法

5. getElementsByName

6. querySelector()和css选择器选择方式相同　ie7及以下不兼容

1. querySelectorAll()　　ie7及以下不兼容 选出来的是副本，不是实时的，如果标签个数出现更改，将不会随之更改

遍历节点树

节点类型

：元素节点　————1

　属性节点————2

文本节点————3

注释节点————8

document————9

documentFragment————11

```
<div>
 123        1.文本节点
 <strong></strong>  2
3    <!-- this is a comment -->4
```

```
5     <span></span>6
7
  </div>
```

七个节点

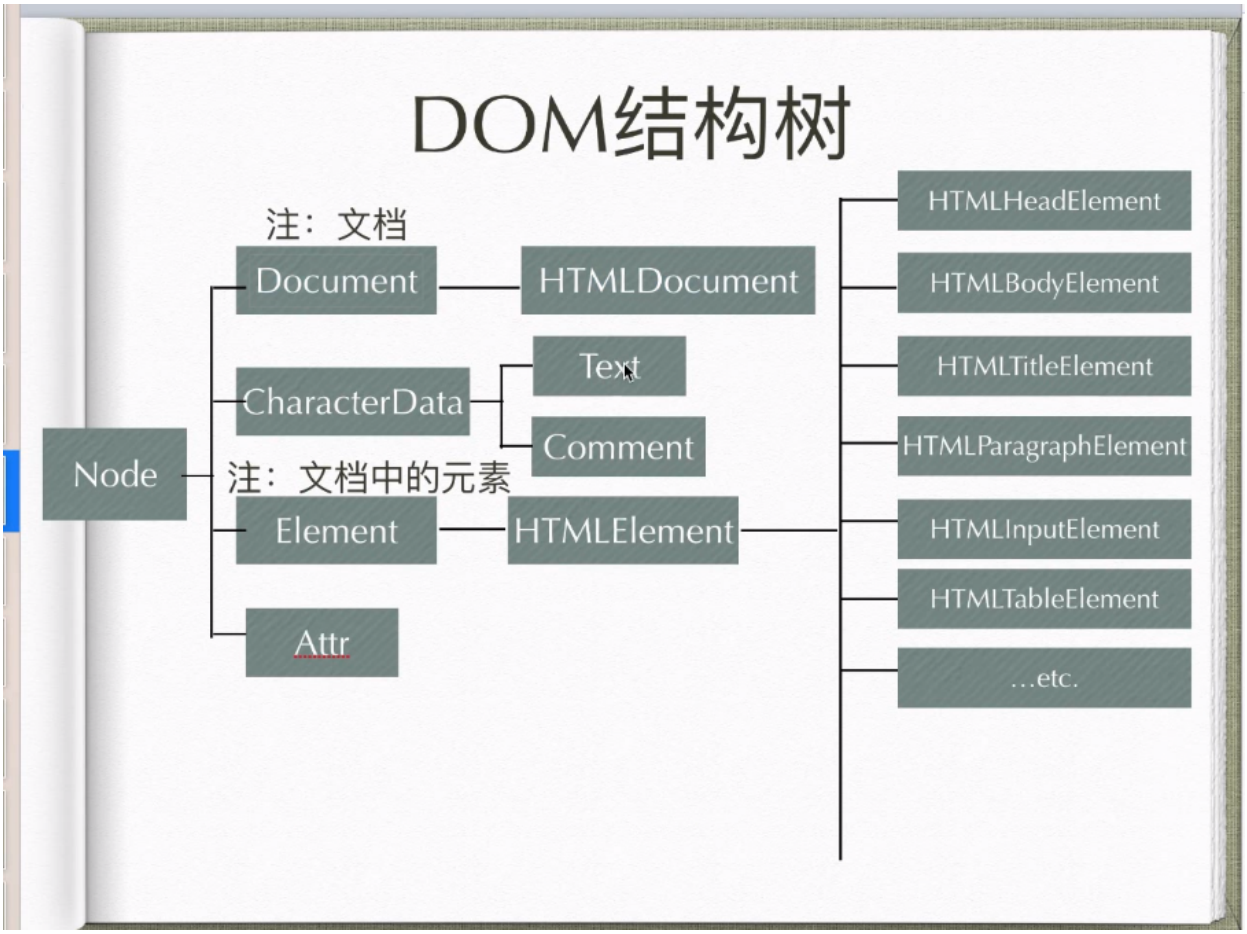
---

parentNode->父节点
childNodes->子节点们
firstChild -> 第一个子节点
lastChild -> 最后一个子节点
nextSibling -> 后一个兄弟节点
previousSibling -> 前一个兄弟节点

parentElement -> 元素父节点 (IE9及以下不兼容)
children -> 元素子节点（上个例子只有两个）
childElementCount == children.length (IE9及以下不兼容)
firstElementChild -> 第一个元素子节点 (IE9及以下不兼容)
lastElementChild -> 最后一个元素的子节点 (IE9及以下不兼容)
nextElementSibling 下一个元素节点 (IE9及以下不兼容)
previousElementSibling -> 上一个元素节点 (IE9及以下不兼容)

节点的属性：

1. **nodeName** 查看节点名（只可以读不可以更改）
2. **nodeValue** 查看文本节点或者注释节点的文本内容，可读写
3. **nodeType** 返回节点类型，通过值来表示 (important!)
4. **attributes** 节点属性的集合
5. **hasChildNodes()** 判断节点有没有子节点



存在继承关系

```
<div>
    <span>1</span>
</div>
  <div></div>
  <script>
   var div = document.getElementsByTagName('div')[0];
   括号内也可以写*代表选择所有 通配符选择器
   var span = div.getElementsByTagName('span')[0];
  </script>
```

document.documentElement代表整个html文档

增加

```
div = document.createElement('div')                    创造一个div元素节点
document.body.appendChild(div)                  在body中加入div标签

var text = document.creatTextNode('1234');      创建文本节点
var comment = document.creatComment('This is a comment');创建一个注释节点
```

插入

```
var div = document.getElementsByTagName('div')[0];
var text = document.creatTextNode('1234');
div.appendChild('text')
   //在div节点中插入文本节点，类似于push()，末尾加入
   该方法是剪切操作，选中的标签使用该方法插入原来位置会消失然后出现在插入的位置


<div>
   <span>
   </span>
</div>
var div = document.getElementsByTagName('div')[0];
var span = document.getElementsByTagName('span')[0];
var strong = document.createElement('strong');
var i = document.createElement('i');
div.insertBefore(strong,span);
div.insertBefore(i,strong);
//在父节点下在后边的子节点之前插入元素节点
```

删除

1，parent.removeChild('');
   //删除父节点下的指定子节点，其实是剪切出来可以再用
2，div.remove()
   //删除div节点，完全销毁

替换

   parentNode.replaceChild(new.origin);
   //返回结果仍然被剪切出来，

节点的属性

div.innerHTML	获取div下的内容
div.innerHTML = '123'	覆盖div内的内容
div.innerHTML += '456'	向div下追加内容
div.innerHTML = ' 123 '	向div下添加带属性内容的节点同样好使
div.innerText	取div下的所有文本内容，添加的都是字符串
div.innerText = '123'	如果div下有其他元素节点都将被替换

innerText(老版本火狐不兼容)/innerContent(老版本IE不好使)

节点的方法

div.setAttribute('class','demo');	（’属性名‘,‘属性值’）
给div设置行间属性	
div.getAttribute('class');	获取div的属性值

```
<div>
      <i></i>
      <b></b>
      <span></span>
  </div>


      Element.prototype.insertAfter = function(targetNode,afterNode){
          var beforeNode = afterNode.nextElementSibling;
          if(beforeNode == null){
              this.appendChild(targetNode);
          }else{
              this.insertBefore(targetNode,beforeNode);
          }
 }
var div = document.getElementsByTagName('div')[0];
var b = document.getElementsByTagName('b')[0];
var i = document.getElementsByTagName('i')[0];
var p = document.createElement('p');
var span = document.getElementsByTagName('span')[0];
```