DOM

Node:

> 文档节点

> 元素节点

> 文本节点

> 属性节点

> 注释节点

·节点的属性

> div.nodeName

> div.nodeValue

> div.nodetype

·层次节点

> 父节点与子节点

> 兄弟节点

·获取节点

> 获取所有节点

childNodes 包含所有节点类型，不单单只是元素

> 获取第一个和最后一个子节点

firstChild

lastChild

注意：不一定是元素节点

> 获取父节点

parentNode

注意：父节点只有一个。

> 获取兄弟节点

previousSibling

nextSibling

注意：只能获取一个

·元素的宽高属性

> offsetWidth 和 offsetHeight

区别：包含了 内边距 和 边框

> clientWidth 和 clientHeight

区别：不包含边框

·子元素与父元素的距离

> offsetLeft 和 offsetTop

距离body左边界和上边界的距离

注意：但如果该子元素使用了定位属性，则参照的就是距离它最近的父级元素

Document

·获取元素

·用id获取元素

> getElementById('获取元素的id')

找到,则返回该元素

找不到，返回null

·用标签名获取元素

> getElementsByTagName('标签名')

返回一个伪数组,或一个节点列表.

·用类名获取元素

> getElementsByClassName('元素名');

·用name属性值获取元素

> getElementsByName('name值')

返回一个伪数组对象.

·用选择器获取元素

> querySelector()querySelectorAll()

注意：只能找到元素列表的第一个元素

> querySelectorAll()

注意：可以全部找到

·获取和设置元素中的其他信息

·获取元素名

> console.log(div.tagName)

注意：该属性只能获取,不能设置.

·获取元素节点里的内容

> 元素名.innerHTML

·设置标签内的内容

> 元素名.innerHTML = '内容';

·获取元素节点中的文本

> 元素名.innerText;

·设置元素中的文本

> 元素名.innerText = '内容';

·获取元素的类名

> 元素名.className

返回值是一个字符串，同时也可以设置新的类名

·获取元素样式

> 元素名.style

返回一个字符串，同时也可以设置新的样式

·获取元素属性

> getAttribute('属性名')

返回值就是字符串形成的属性值

·设置元素属性

> setAttribute('属性名','属性值')

·删除元素属性

> removeAttribute('需要删除的属性名')

·增加元素

·创建元素节点

> document.createElement('标签名')

注意：创建了不是就存在了，需要将这个已经创建的元素放到你想放到的元素(父级元素)中去

document.body.appendChild(div);

·创建文本节点

> document .createTextNode('一段文字');

注意：最后插入到需要创建的元素中

p.appendChild(text);

·css样式赋予

> style.cssText 可以一次性赋予样式

div.style.cssText='width:100px;height:100px;background-color:red;'

·在某元素前插入创建的新元素

> insertBefore('需要插入的元素'，'在谁之前插入')

> appendChild()

用法基本一致

区别：insertBefore()可以选择插入的位置

·删除和替换元素

·元素的替换

> replaceChild(将要替换的新元素，被替换的旧元素)

·元素的删除

> removeChile(需要删除的元素)

·事件

·鼠标事件

> 点击事件

mydiv.onclick = function(){}

> 双击事件

mydiv.ondblclick = function(){}

> 鼠标按下

mydiv.onmousedown = function(){}

> 鼠标抬起

mydiv.onmouseup = function(){}

> 鼠标移动

document.onmousemove = function(){}

注意：只要鼠标移动就触发

·鼠标移入移出:

(1)第一套移入出会多次触发

mydiv.onmouseover = function(){}

mydiv.onmouseout = function(){}

(2)第二套移入出不会多次触发

mydiv.onmouseenter = function(){}

mydiv.onmouseleave = function(){}

·键盘事件

> 键盘按钮按下

Onkeydown

> 键盘按钮抬起

Onkeyup

·Event事件

> Event对象常用属性

e.clientX和e.clientY

> 键盘的坐标值

e.KeyCode

> Event的兼容性

//事件触发时 调用的函数

document.onclick = function(e){

// 非IE | 低版本IE

var ev = e ||window.event;

console.log(e.clientX);

}

·对象事件

> 页面（元素）加载完成后执行

window.onload

> 窗口变化大小时触发事件

window.onresize

·表单事件

> 获取焦点

onfocus

> 失去焦点

onblur

> 用户输入触发

oninput

> 内容改变时触发

onchange

·事件绑定

> 事件绑定的方法

btn2.onclick = function(){

alert('按钮2');

}

> 解除事件绑定的方法

btn2.onclick =null;

·事件监听

> 事件监听的方法

增加监听器addEventListener（监听事件，时间函数，冒泡）

IE：attachEvent（'监听的事件，需要加on',触发的事件函数）

> 事件监听解绑和封装

<script>

var mydiv = document.getElementById('mydiv');

//封装监听

function addE(target,type,fn){

if(target,addEventListener){

target.addEventListener(type,fn);

}else {

target.attachEvent('on'+type,'fn');

}

}

addE(mydiv,'click',a);

//封装解绑

function removeE(target,type,fn){

if(target,removeEventListener){

target.removeEventListener(type,fn);

}else {

target.datachEvent('on'+type,'fn');

}

}

</script>