BOM

一、BOM

专门操作浏览器窗口的API

1、window：2个角色

(1)代替了ES中Global充当全局对象

(2)封装了浏览器软件/窗口对象是属性和方法--BOM

2、打开新链接的方式

(1)在当前窗口打开，允许后退

HTML：<a href="xxx">在当前窗口打开，可后退</a>

JS：window.open("url","\_self");

(2)在当前窗口打开，不允许后退

history：记录【当前】窗口【本次】打开后，成功访问过的url

它是一个历史记录栈

history.go(-1)：相当于点一次后退

history.go(1)：相当于点一次前进

history.go(0)：相当于点一次刷新

location：当前窗口正在打开的url对象(地址栏)

location.href:"url";保存了当前窗口正在访问的url地址

设置url属性为新url，会在当前窗口打开新url

location.replace("url");

location.assign("url");设置当前窗口的新url

location.reload(true/false);

true：无论是否更改，都获取最新

false：被修改的页面，会重新获取；未被修改的页面，从缓存中取

HTML：不能实现，因为它没有权限访问地址栏

JS：location.replace("url");

(3)在新窗口打开，可重复打开

HTML：<a href="xxx" target="\_blank">在新窗口打开，可重复打开</a>

JS：window.open("url","\_blank");

其中,"\_blank"可省略不写，因其默认值就是blank

(4)在新窗口打开，不可重复打开

\*内存中，窗口都有唯一name

浏览器规定：同名窗口，只能开一个，后开的会覆盖新开的

其实target的属性值，就是窗口的name属性，可自定义

HTML：<a href="xxx" target="xx自定义">在新窗口打开，不可重复打开</a>

JS：window.open("url","xx自定义");

3、窗口大小和窗口定位

(1)在打开窗口时，就设定窗口大小：

var opendWindow=window.open("url","name",config);

var config="top=?,left=?,width=?,height=?,resizable=yes|no,location=no";

(2)文档显示区的宽和高

window.innerHeight/Width

文档显示区：仅用于网页的范围大小(不包括顶部和地址栏等)

(3)整个窗口的宽和高

window.height/width

(4)屏幕大小

screen.height/width，表示操作系统的桌面分辨率

screen.availHeight/Width，表示桌面分辨率去掉任务栏的宽度

windows7下几乎无差别

(5)调整大小

window.resizeTo(width,height);

window.resizeBy(宽度增量,高度增量);

(6)移动窗口位置

window.moveTo(left,top);

window.moveBy(x的增量,y的增量);

二、定时器：2种

让程序按指定时间间隔，自动执行【任务】

任务是所有定时器的核心

1、周期性定时器：让浏览器按照指定时间间隔反复执行同一任务

何时使用：一项任务，连续执行

如何使用：3步

(1)定义任务函数

定义了定时器每一次执行的一项任务

function task(){...}

(2)启动定时器

将任务函数放入定时器API中，同时设定时间间隔，最后返回定时器的【序号】

timer=setInterval(task,毫秒数);

timer是一个全局变量(名字自定义)

(3)停止定时器

只能通过序号来停止

clearInterval(timer);

timer=null;

2、一次性定时器：让浏览器先等待一定时间间隔后，再自动执行一次任务

如何使用：同周期性定时器一样，只是要将Interval换为Timeout

\*\*执行原理：先等待指定时间，再执行一次，自动退出

如果主程序没有执行完，或被阻塞，定时器回调函数永远无法执行，也就是说，定时器的回调函数必须等待主程序执行完才能执行

固定套路：利用一次性定时器，模拟周期性定时器的效果

何时使用：多个任务叠加在一起，避免反复创建多个周期性定时器；任务可能频繁停止或频繁启动

例如 for(var i=0;i<3;i++){

setTimeout(

function(){

console.log(i);

},10);

}

→每隔10毫秒执行一次，总共执行3次3

首先，for为主程序，在主程序没有执行完之前，定时器是不执行的，所以i最终循环的结果为3，所以结果都是输出3；i取0、1、2，所以执行了3次

三、navigator

window中封装浏览器属性和配置信息的对象

1、属性：

(1)navigator.cookieEnabled：识别浏览器是否启用cookie

返回值：true/flase

(2)navigator.userAgent：保存了浏览器名称和版本的字符串

(3)navigator.plugins：封装了所有插件对象的类数组对象

每个plugin对象的name属性保存了插件的名称

四、事件

浏览器自动触发的或用户手动触发的对象状态的改变

1、事件处理函数：当事件发生时，自动执行的函数对象，其实就是 onxxx

2、如何绑定事件处理函数：

(1)在html元素的开始标签中设置事件处理函数属性

<button on事件名="fun()"></button>

(2)在js中动态绑定事件处理函数

①同一元素的同一事件处理函数，只能绑定一个函数对象

elem.on事件名=函数对象

比如btn.onclick=function(){...} //this指btn

执行：btn.onclick();

②可同时绑定多个，可修改事件触发顺序

I DOM标准：

elem.addEventListener("事件名",函数对象,是否在捕获阶段触发);

比如btn.addEventListener("click",fun,false/true);

II IE8：

elem.attachEvent("on事件名",函数对象);

比如btn.attachEvent("onclick",fun);

3、事件周期

从事件触发到各级事件执行完的全过程

(1)DOM标准：3个阶段

(从根开始，有外向内，到目标元素)捕获

(实际触发事件的对象)目标捕获

(从目标向外，到根结束)冒泡触发

(2)IE8：2个阶段

(实际触发事件的对象)目标捕获

(从目标向外，到根结束)冒泡触发

(3)是否可修改事件的执行顺序

可以修改

→修改addEventListener的第三个参数为true，可在捕获阶段提前触发

\*同一事件不可触发2次！

\*如果点在外层元素上，捕获只到目标元素结束

(4)事件对象：当事件发生时，自动创建的封装事件信息的对象

如何获取事件对象：

DOM标准：事件对象作为事件处理函数第一个参数，传入函数中

arguments[0]→事件对象

IE8：事件对象是window下的一个全局属性，window.event

\*获得事件对象的兼容代码(固定套路)：

var e=window.event||arguments[0];

只适用于js中动态绑定事件处理函数

4、固定套路

(1)取消冒泡：在当前事件处理函数执行结尾，阻止继续向上冒泡

①DOM：

e.stopPropagation();

②IE：

e.cancelBubble=true;

兼容代码：

if(e.stopPropagation){

e.stopPropagation();

}else{

e.cancelBubble=true;

}

(2)利用冒泡

优化：多个子元素，定义了相同的事件处理函数，其实只需要将事件处理函数在父元素上定义一次即可

因为每个处理函数的绑定都是一个对象

如果处理函数绑定的越多，网页的执行效率越低

减少处理函数绑定的次数，可提高执行效率

获得目标元素：

var target=e.srcElement||e.target;

目标元素 vs this：

this：会随事件冒泡而改变

目标元素：不受冒泡影响，始终保存最初的目标元素对象

(3)取消事件：在事件执行过程中，发生异常状况，可阻止事件触发

比如onsubmit：自动提交之前触发

如果验证未通过，可取消继续提交

何时使用：取消元素默认的事件行为

①在html中绑定事件处理函数：2个return

比如<form onsubmit="return valiAll()">

对应的js中： valiAll(){

... ...

return true/false;

}

true：继续执行

false：取消执行

②在js中动态绑定处理函数(也可用于html绑定)

如何取消： if(e.preventDefault){

e.preventDefault(); //DOM

}else{

e.returnValue=false; //IE

}

5、事件坐标：3种

(1)相当于显示器左上角

最大值：screen.availWidth/Height

鼠标位置：e.screenX/Y

(2)相对于文档显示区左上角

最大值：window.innerWidth/Height

鼠标位置：e.clientX/Y

或e.x/y

(3)相对于父元素左上角

最大值：父元素的宽和高

鼠标位置：e.offsetX/offsetY

6、页面滚动

事件：window.onscroll

滚动的top距离：document.documentElement.scrollTop||document.body.scrollTop

比如在控制台：

document.body.style.height="2000px";

window.onscroll=function(){

console.log(document.documentElement.scrollTop||document.body.scrollTop);

}

就可在页面滚动窗口

五、cookie

客户端保存用户个人信息的文件

每个cookie是一个key/value对

key表示存储的属性名，value表示属性的值

何时使用：只要在客户端持久缓存数据

什么样的数据放在客户端：用户的个人信息、操作、配置

为什么要持久保存：即使当前回话结束，其他网页或下次启动会话，还需要使用该数据

1、属性：

(1)expires，过期时间

必须是一个晚于当前时间的时间点

下次访问相同网页时，浏览器会检查cookie，如果过期，才会删除cookie

(2)domain，规定了cookie只能被哪个域名使用

可设置

2、如何使用cookie：

(1)保存：

document.cookie="key=value"

(2)读取：

var cookies=document.cookie.split("; ");

cookies[i]:"key=value"