BOM:Browser Object Model 操作浏览器窗口的API 没有标准 浏览器兼容性问题

包括：

window 封装左右DOM和BOM的API 代替global充当全局作用域的对象

history

location

navigator

document:封装DOM树，并提供DOM的API

event

screen

window

属性：innerWidth innerHeight浏览器窗口中，文档显示区的宽高

方法 .open(‘url’, ‘name’) .close()

window.onload 页面加载完在加载此

每个网页在window中都有一个name，同一个name只能打开一个网页

打开链接方式总结

内置name\_self:用当前窗口自己的名字打开新窗口

\_blank:不指定窗口名，随机生成窗口名

在当前窗口打开，可后退

html:<a href=’’ target=’\_self’></a>

js:window.open(‘url’,’\_self’)

在当前窗口打开，不可后退

js:location.replace(‘url‘);

在新窗口打开，可打开多个

html: <a href=’’ target=’\_blank’></a>

js:window.open(‘url’,’\_blank’);

在新窗口打开，只能打开一个

html: <a href=’’ target=’自定义窗口名’></a>

js: window.open(‘url’,’自定义窗口名’)

history:保存当前窗口打开后，成功访问过的url,为素组，新打开的url会push进history

API：history.go(n)正数前进 负数后退 刷新0

location:保存当前窗口正在打开的url对象

属性：.href完整url地址

.protocol 协议

.host 主机名+端口号

.hostname 主机名

.port 端口号

.pathname 相对路径

.hash #锚点地址

.search ?查询字符串

方法：location.assign(url); 在当前窗口打开，可后退

location.replace(url) 当前窗口打开禁止后退

location.reload()普通刷新：优先从浏览器本地缓存中获取资源

location.reload(true)强制刷新，总是从服务器获取资源

定时器：3种

周期性定时器：让程序每隔一段时间反复执行一项任务

timer=setInterval(task, interval)

启动定时器

停止定时器：clearInterval(timer) timer 定时器的序号，在内存中唯一标识定时器的整数只能在定时器启动的时候获得

习惯上要在停止定时器的时候清除timer=null

一次性定时器：

让程序先等待一段时间，再执行一次任务，执行后自动停止

task

启动定时器 timer = setTimeout(task, wait)

停止定时器：clearTimeout（timer）

requestAnimationFrame()用于动画，根据机器性能，自动选择绘制间隔时间，但是不同浏览器名字不同

定时器原理：定时器的任务函数必须等待主程序所有语句执行后，才能执行。

navigator 保存浏览器配置信息的对象

.cookieEnabled:判断浏览器是否启用cookie：在客户端永久储存用户私密数据的小文件 返回值为bool

.plugins 包含浏览器所有插件的集合

.userAgent 判断浏览器的名称和版本号

event 事件，人为触发的。浏览器自动触发的页面内容的改变

事件处理函数：当事件发生时自动执行的函数

在HTML中绑定：<ANY on 事件名=“js语句”>当事件发生时，自动执行js语句不符合内容与行为分离的原则，不便于维护和重用

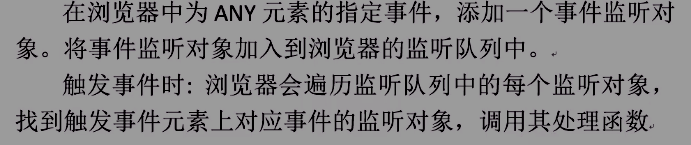
但是在组件开发中，反而要求内容行为和样式集中定义在一个小组件内，自成体系。

在js中绑定，每个事件只能绑定一个处理函数

ANY.on 事件名 = function(){} 发生：ANY.on 事件名（）

在js中绑定，每个事件绑定多个处理函数

ANY.addEventListener(‘事件名’,handler);



ANY.removerEventListener(事件名’,handler);移除事件名 该函数不能用匿名函数。

DOM事件模型：从事件触发到处理函数执行所经过的过程

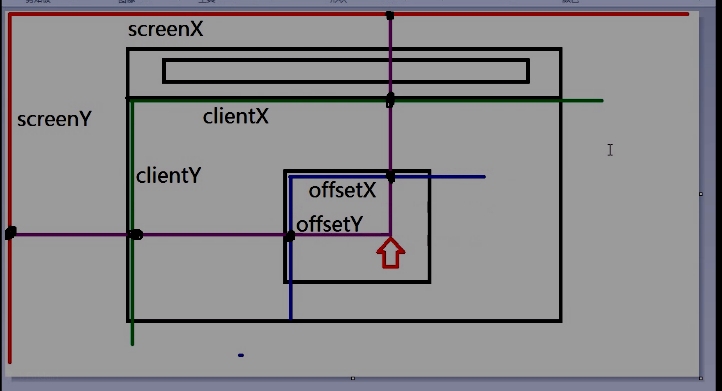
1. 捕获阶段，由外向内，记录各级父元素上绑定的事件处理函数，仅记录不触发
2. 目标触发，优先触发实际目标的处理函数
3. 冒泡执行，由内向外，按照捕获的函数执行父元素上的函数

事件对象：当事件发生时，自动创建的，封装事件信息的对象，只要希望获得事件信息或者修改事件的默认行为时

1. 取消冒泡：e.stopPropagation();
2. 利用冒泡：减少事件监听的个数，对多个子元素绑定相同事件时，在父元素绑定一次
   1. 获得目标元素：e.target
   2. 筛选目标函数：nodeName.class….
3. 阻止默认行为：只要默认行为不想要 e.preventDefault();
   1. <a> a会添加#
   2. 表单验证没通过组织提交
   3. HTML5中拖拽API，组织浏览器默认拖拽行为

鼠标坐标

1. 相对于屏幕左上角：e.screenX e.screenY
2. 相对于文档显示区左上角 e,clientX e.clientY
3. 相对于当前元素左上角 e.offsetX e.offsetY



页面滚动 window.onscroll

获得页面滚动过的高度：scrollTop = documentElement.scroTop||document.body.scroTop;

滚动API：

window.scrollTo(x,y);绝对定位

window.scrollBy(x,y);相对定位

onkeyup 键盘抬起事件

onkeydown 键盘按下事件

用事件e访问按键e.keyCode获取建号

token