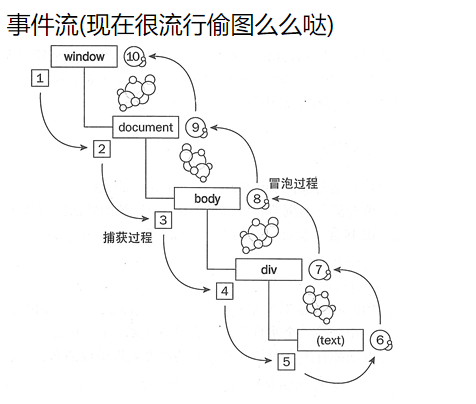
为什么会有事件流：

事件流的起源：就是在浏览器发展到第四代的时候， IE提出的是冒泡流，而网景提出的是捕获流。

前面说过，IE提出的是冒泡流，而网景提出的是捕获流，后来在W3C组织的统一之下，JS支持了冒泡流和捕获流，但是目前低版本的IE浏览器还是只能支持冒泡流(IE6,IE7,IE8均只支持冒泡流)，所以为了能够兼容更多的浏览器，建议大家使用冒泡流。



**注意：这里的window 指的是：表示浏览器中打开的窗口**

BOM就是浏览器窗口对象模型，顶级对象就是window   
window，document都是一个实例对象，他们都属于Object，你不可以在new他，必须直接调用他的方法和属性   
Window对象共有七个属性：   
defauleStatus：指定窗口状态栏中的信息。   
status：指定当前窗口状态栏中的信息。   
frames：是一个数组，其中内容是窗口中所有的框架。   
parent：指当前窗口的父窗口。   
self：指当前窗口。   
top：代表当前所有窗口的最顶层窗口。   
window：代表当前窗口。   
Window对象有五个方法：   
alert：显示带有一个“确定”按钮的对话框。   
confirm：显示带有“确定”与“取消”两个按钮的对话框。   
prompt：显示带有输入区的对话框。   
open：打开一个新窗口。   
close：关闭用户打开的窗口。

1、一个完整的JS事件流是从window开始，最后回到window的一个过程  
2、事件流被分为三个阶段(1~5)捕获过程、(5~6)目标过程、(6~10)冒泡过程

DOM事件流：将事件分为三个阶段：捕获阶段、目标阶段、冒泡阶段。先调用捕获阶段的处理函数，其次调用目标阶段的处理函数，最后调用冒泡阶段的处理函数。

事件冒泡 ：当一个元素接收到事件的时候 会把他接收到的事件传给自己的父级，一直到window 。（注意这里传递的仅仅是事件 并不传递所绑定的事件函数。所以如果父级没有绑定事件函数，就算传递了事件 也不会有什么表现 但事件确实传递了。）

**Eg1**

**事件委托**（有时候也叫事件代理）

利用了事件冒泡，只指定一个事件处理程序，就可以管理某一类型的所有事件。例如，click事件会一直冒泡到Document层次。也就是说我们可以为整个页面指定一个onclick事件处理程序，不必每个元素添加处理事件。

(五大美女案例)

事件委托的优缺点：

通过上面的介绍，大家应该能够体会到使用事件委托对于web应用程序带来的几个优点：  
1.可以大量节省内存占用，减少事件注册。  
2.可以方便地动态添加和修改元素，不需要因为元素的改动而修改事件绑定。  
3.JavaScript和DOM节点之间的关联变少了，这样也就减少了因循环引用而带来的内存泄漏发生的概率。  
缺点：  
不是所有的事件都能冒泡的。blur、focus、load和unload不能像其它事件一样冒泡。事实上blur和focus可以用事件捕获而非事件冒泡的方法获得（在IE之外的其它浏览器中）。  
在管理鼠标事件的时候有些需要注意的地方。如果你的代码处理mousemove事件的话你遇上性能瓶颈的风险可就大了，因为mousemove事件触发非常频繁。而mouseout则因为其怪异的表现而变得很难用事件代理来管理。

为什么要阻止事件冒泡：

兼容性问题简单聊一下：

一般情况，除了IE，其他浏览器都好说话

IE是分阶段的。