正课:

1. 事件:
2. 事件:

什么是: 浏览器自动触发的，或者用户手动触发的页面内容/状态的改变

什么是事件处理函数: 在事件发生时，自动指定的回调函数

何时: 如果希望在发生事件时，执行一项任务

如何: 事件处理函数都要先绑定到元素上：3种:

1. 在HTML中: <ANY on事件名=”js语句”

问题: 不符合内容与行为分离的原则，不便于维护

1. 在js中,用赋值方式: elem.on事件名=function(){

this->当前触发事件的.前的元素

}

当事件发生时: elem.onclick()

问题: 一个事件只能绑定一个处理函数

1. 在js中，添加事件监听对象:

elem.addEventListener(“事件名”,function(){

… …

})

elem.removeEventListener(“事件名”,原处理函数)

强调: 1. 如果一个处理函数可能被移除，绑定时就不能用匿名函数，应该用具名函数

2. 移除时，必须通过函数名找到原函数对象

事件模型: DOM规定: 3个阶段

1. 捕获: 由外向内，依次记录各级父元素上绑定的处理函数
2. 目标触发: 优先触发目标元素上的处理函数

目标元素: 最初实际触发事件的那一个元素

1. 冒泡: 由内向外，按捕获的记录，反向执行各级父元素上绑定的处理函数

所有浏览器的开发者一致认为，点在内层元素上，等效于也点在外层元素上

事件对象:

什么是: 事件发生时，自动创建的保存事件信息的对象

何时: 1. 想获得事件信息时

2. 想改变事件的默认行为时

如何:

获取事件对象: 事件对象总是作为处理函数的第一个参数自动传入处理函数.

elem.onclick = function(e) { e->事件对象}

↑

当事件发生时: elem.onclick(新建event对象)

API:

1. 取消冒泡: e.stopPropagation()
2. 利用冒泡:

优化: 尽量减少事件监听对象的个数！

因为: 浏览器触发事件处理函数是采用遍历的方式查找监听对象。如果监听对象多，遍历慢

解决: 当多个子元素，需要绑定相同的事件时，其实只在父元素绑定一次即可！所有子元素通过冒泡共用！

2个难题:

1. 获得目标元素，而不是父元素

错误: this->父元素

正确: e.target

1. 判断目标元素是否想要的

在处理函数中先用if判断e.target是否想要的

只有是想要的时，才执行操作

3. 阻止默认行为:

何时: 只要元素自带的默认行为不是想要的，就可阻止

如何: e.preventDefault();

常用:

1. 用a当按钮时: 阻止a自动添加#xxx到地址栏
2. 其实submit按钮也能阻止提交！

input type=submit 触发 form.onsubmit=function(e){

如果验证不通过

e.preventDefault()

}

4. 鼠标位置: 3组:

1. 相对于屏幕左上角的坐标: e.screenX e.screenY

2. 相对于文档显示区左上角的坐标: e.clientX e.clientY

3. 相对于当前元素左上角的坐标: e.offsetX e.offsetY

滚动事件:

window.onscroll=function(){

var scrollTop=document.body.scrollTop

||document.documentElement.scrollTop

}