正课:

1. navigator

2. \*\*\*event

3. screen

1. navigator

plugins: 保存浏览器插件信息的集合

什么是插件: 为浏览器添加新功能的小程序

何时: 只要为浏览器添加新功能时

如何判断: navigator.plugins["插件名"]!==undefined

userAgent: 保存浏览器名称和版本号的字符串

2. \*\*\*event:

什么是事件: 用户手动触发的或浏览器自动触发的页面内容状态的改变。

什么是事件处理函数: 当事件发生时，浏览器自动触发的函数

何时: 只要希望事件发生时，自动执行一项任务

如何: 提前为元素绑定事件处理函数

3种:

1. 在HTML中: <ANY on事件名="js语句">

缺: 不符合内容与行为分离的原则

2. 在js中集中绑定事件处理函数

elem.on事件名=function(){

//this->.前的elem对象

}//如果用箭头函数简写，则this失效

原理: 触发时: elem.on事件名()

问题: 用等号赋值的方式替换原事件处理函数

无法为同一个事件，绑定多个处理函数

解决: 事件监听

3. 在js中添加事件监听

什么是事件监听: 保存一个元素及其事件对应的一个处理函数打的小对象.

elem.addEventListener("事件名",function(){

...this->.前的elem

});

elem.removeEventListener("事件名",?)//和添加监听时完全一样

问题: 移除时，需要获得原处理函数的地址

解决: 如果一个处理函数可能被移除，则绑定时，就不能用匿名函数，必须用有名的函数

\*\*\*DOM事件模型: 3个节点

1. 捕获capture: 由外向内，记录各级父元素上绑定的事件处理函数——捕获阶段只记录，不执行

2. 目标触发: 优先触发最内层的实际触发事件的元素上绑定的事件处理函数

3. 冒泡/蔓延propagation: 由内向外，按照捕获阶段记录的反向顺序，依次执行父元素上的事件处理函数

\*\*\*事件对象: 事件发生时，自动创建的，封装事件信息的对象。提供了操作事件行为的API。

如何获得事件对象: 事件对象自动作为处理函数的第一个参数，自动传入! ANY.onclick=function(e){

//e->事件对象

}

如何使用e:

1. 阻止蔓延/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

为什么: 浏览器使用遍历方式查找事件监听对象，并触发处理函数。事件监听对象越多，遍历时间长，事件响应慢

如何: 如果多个子元素绑定相同的事件时，其实只要在父元素绑定一份即可！所有子元素可通过冒泡，共用！

难点:

1. 获得实际触发事件的目标元素

错误: this->父元素，不再指子元素

正确: e.target->实际触发事件的目标元素

强调: e.target不会随冒泡儿改变

2. 鉴别目标元素的种类:

if(e.target.nodeName=="xxx")

.className=="xxx"

普通绑定 vs 利用冒泡:

1. 绑定的元素对象:

普通绑定,直接将处理函数绑在子元素上，事件监听多  
 利用冒泡，将处理函数绑在父元素上，事件监听少

2. 动态添加的子元素:

普通绑定, 每动态添加一个子元素，都要重复绑定事件

利用冒泡, 即使动态添加子元素，也不用重复绑定事件。

子元素可随时通过冒泡，自动获得父元素上的事件

3. 阻止默认行为:

什么是: 阻止事件默认的行为

何时: 只要事件的默认行为不是想要的

如何: e.preventDefault

比如: 3种:

1. 用a元素作为按钮时:

默认: href="#锚点" 会被自动添加到url结尾

希望: 仅作为按钮，而不修改url

2. 用submit按钮提交表单:

默认: 无论如何都自动提交

希望: 根据验证的结果，手动控制是否提交

3. HTML5中用拖拽API，都要先阻止浏览器默认的拖拽行为的干扰

4. 鼠标位置: 3组:

相对于屏幕左上角: e.screenX, e.screenY

相对于文档显示区左上角: e.clientX, e.clientY

相对于当前元素左上角: e.offsetX, e.offsetY

5. 页面滚动:

事件: window.onscroll

属性: scrollTop: 页面顶部超出文档显示区顶部的距离

scrollTop=document.body.scrollTop

||document.documentElement.scrollTop

方法: window.scrollTo(x目标位置,y目标位置)

window.scrollBy(x增量,y增量)

3. screen: 封装当前显示设备信息的对象

screen.width screen.height

screen.availWidth screen.availHeight