正课:

1. 定时器原理:

2. navigator:

3. event:

4. 页面滚动:

附加: 对话框特效

1. 定时器原理:

定时器中的任务函数，必须等待主程序所有语句执行后，才能执行。

2. navigator:

什么是: 保存浏览器配置信息的对象

包括:

.cookieEnabled: 判断当前浏览器是否启用cookie

什么是cookie: 在客户端持久存储用户私密数据的小文件

为什么: 内存中所有数据都是临时的! 程序关闭，内存中一切变量都释放!

何时: 只要希望在客户端持久保存数据，都用cookie

.plugins: 包含浏览器所有插件信息的集合

什么是插件: 为浏览器添加新功能的小软件

如何判断是否安装指定插件:

.userAgent: 保存浏览器名称和版本号的字符串

何时: 只要判断浏览器名称和版本号

4. event:

什么是事件: 人为触发的，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，自动执行的函数。

如何绑定: 3种:

1. 在HTML中绑定:

绑定: <ANY on事件名="js语句">

当事件发生时: 自动执行js语句

问题: 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。

2. 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(){ ... }

当事件发生时: ANY.on事件名() //this->ANY

问题: 用赋值方式绑定事件处理函数

3. 在js中绑定，每个事件可绑定多个处理函数:

ANY.addEventListener('事件名',handler)

在浏览器中为ANY元素的指定事件，添加一个事件监听对象。将事件监听对象加入到浏览器的监听队列中。

触发事件时: 浏览器会遍历监听队列中的每个监听对象，找到触发事件元素上对应事件的监听对象，调用其处理函数

移除事件监听:

ANY.removeEventListener('事件名',handler)

说明: handler必须是绑定时使用的原函数对象

强调: 如果一个处理函数，有可能被移除，则不能使用匿名函数绑定。应使用有名的函数绑定

DOM事件模型:

什么是: 从事件触发到处理函数执行，所经过的过程

3个阶段:

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。——仅记录，不触发!

2. 目标触发: 优先触发实际点击的元素上绑定的处理函数

3. 冒泡执行: 由内向外，按捕获阶段记录的处理函数的倒序，依次执行父元素上的处理函数。

事件对象:

什么是: 当事件发生时，自动创建的，封装事件信息的对象

何时: 只要希望获得事件信息，或修改事件的默认行为时

如何获取: 事件对象默认总是以处理函数第一个参数，自动传入

如何使用:

1. 取消冒泡: e.stopPropagation()

2. 利用冒泡:

优化: 尽量减少事件监听的个数

为什么: 浏览器查找事件监听，采用的是遍历的方式

事件监听多，浏览器查找就慢

何时: 如果对多个子元素绑定相同事件时，都要利用冒泡

如何: 只要在父元素绑定一次，所有子元素共用即可!

2个难题:

1. 获得目标元素:

目标元素: 最初实际触发事件的当前元素

如何获得:

错误: this->父元素

正确: e.target

2. 筛选目标元素:

比如: 通过nodeName, class, 内容。。。

3. 阻止默认行为:

何时: 只要事件的默认行为不是想要的

如何: e.preventDefault();

何时:

1. 用a当按钮时，a会自动向地址栏中添加#锚点地址。

2. 提交表单时，如果验证没通过，可阻止提交

自定义表单提交:

input button + onclick + form.submit

input submit + form.onsubmit事件 + e.preventDefault()

3. HTML5中拖拽API: 首先要阻止浏览器默认的拖拽行为

4. 鼠标坐标: 3组:

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

4. 页面滚动:

事件: window.onscroll

获得页面滚动过的高度: body顶部超出文档显示区顶部的距离

scrollTop=document.documentElement.scrollTop

||document.body.scrollTop;

滚动API:

window.scrollTo(left, top)

window.scrollBy(left的增量,top的增量)