**正课:**

**1. navigator**

**2. 定时器**

**3. 事件**

**1. navigator:**

**cookieEnabled: 判断当前浏览器是否启用cookie**

**什么是cookie: 客户端本地持久保存用户个人信息的小文件**

**为什么: 程序内存中的数据(变量，数组，对象)都是临时存储。程序关闭，内存释放！**

**何时: 只要希望在客户端持久保存一个数据**

**原理:**

**userAgent: 保存当前浏览器名称和版本号的字符串**

**何时: 判断浏览器的种类和版本**

**2. 定时器:**

**2种:**

**1. 周期性: 让程序每隔一段时间间隔，反复执行一项任务**

**如何: 3件事:**

**1. 任务函数: 让定时器每次执行什么样的任务**

**2. 启动定时器: var timer=setInterval(任务函数,间隔ms)**

**3. 停止定时器: clearInterval(timer)**

**2种:**

**1. 用户手动停止**

**2. 自动停止定时器: 在任务函数中判断临界值。一旦到大临界值，则自动调用clearInterval()**

**2. 一次性: 让程序等待一段时间，后延迟执行一项任务，执行任务后自动关闭定时器**

**如何: 3件事：**

**1. 任务函数: 让定时器执行的任务**

**2. 启动定时器: var timer=setTimeout(任务函数,延迟ms);**

**3. 停止定时器: clearTimeout(timer)**

**3. 事件:**

**什么是事件: 浏览器自己触发的或用户手动触发的页面状态的改变。**

**什么是事件处理函数: 事件发生时自动调用的函数**

**何时: 只要希望发生事件时，自动执行一项任务**

**如何: 先将处理函数绑定到元素的事件属性上，当事件发生时自动调用**

**如何绑定: 3种:**

**1. 在HTML中绑定:**

**<elem 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**

**2. 在js中绑定: 2种:**

**1. 赋值方式:**

**elem.on事件名=function(){**

**... ...**

**}**

**问题: 一个事件属性只能绑定一个处理函数**

**2. 添加事件监听:**

**elem.addEventListener("事件名",处理函数)**

**原理:**

**1. 创建一个事件监听对象，封装元素,事件名和处理函数**

**2. 将事件监听对象加入浏览器的事件队列中**

**3. 当事件发生时，浏览器通过遍历找到符合条件的监听对象，调用其处理函数执行。**

**elem.removeEventListener("事件名",处理函数)**

**强调: 移除时必须获得原处理函数对象**

**事件模型: 事件发生时的执行过程**

**DOM标准: 事件模型分3个阶段**

**1. 捕获: 由外向内，记录各级父元素绑定的处理函数**

**2. 目标触发: 优先触发目标元素上的处理函数**

**目标元素: 实际触发事件的元素**

**3. 冒泡: 由内向外，依次触发各级父元素上的处理函数**

**为什么: 所有浏览器的开发者都认为，点在内层元素上，等效于点在外层元素上**

**事件对象:**

**什么是: 事件发生时，自动创建的，封装事件信息，并提供操作事件的API**

**何时: 只要希望获得事件的信息或操作事件的行为时**

**如何:**

**获得事件对象: 事件对象总是默认作为处理函数的第一个参数自动传入: function(e){ ... }**

**API:**

**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: 浏览器通过遍历方式触发事件，监听个数多，遍历就慢**

**如何: 如果多个子元素要绑定相同的事件，则只需要在父元素上绑定一次即可，所有子元素共用**

**2大难题:**

**1. this: 事件绑定在父元素上，所以this->父元素**

**解决: e.target 自动获得目标元素，且不随冒泡而改变。**

**2. 鉴别目标元素: 不是所有目标元素都是想要的，都要先判断目标元素是否符合要求，再执行操作！**

**鄙视: 利用冒泡 vs 普通绑定**

**1. 绑定位置不同:**

**普通绑定: 绑在子元素上**

**利用冒泡: 绑在父元素上**

**2. 事件监听的个数:**

**普通绑定: 多**

**利用冒泡: 只有一个**

**3. 对动态生成的新元素:**

**普通绑定: 无法自动获得事件处理函数**

**利用冒泡: 可自动获得事件处理函数**

**3. 阻止默认行为:**

**何时: 如果希望不执行HTML元素自带的行为时**

**三种情况:**

**1. <a href="#xxx">自动在地址栏结尾添加#xxx锚点地址**

**2. 阻止input submit自动提交表单**

**3. 取消默认拖拽行为: 下个老师讲**

**如何: e.preventDefault()**

**4. 获得鼠标位置:**

**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**

**页面滚动事件:**

**响应页面滚动事件:**

**window.onscroll=function(){**

**获得页面滚动的距离: scrollTop**

**scrollTop=document.body.scrollTop**

**||document.documentElement.scrollTop**

**}**