学习方法

1. 建立逻辑思维
   1. 多分析
   2. 理解逻辑算法
   3. 多练、多敲、多读
2. 编程习惯
3. 大胆动手
4. 保持好奇好
5. 乐观面对错误

就业能力

1. 分析能力
2. 独立思考能力
3. 程序阅读能力
4. 程序排错能力
5. 独立编写程序能力
6. 团队协作能力
7. 学习能力
8. 善于总结的能力

专业技能

Event事件下

1. 浏览器的默认行为
   1. 右键菜单：oncontextmenu
   2. 超链接
   3. 拖拽：ondragstart
2. 阻止默认行为（兼容）
   1. 阻止右键菜单 ： return false
   2. 阻止超链接的默认行为
      1. Event.preventDefault()://标准
      2. Event.returnValue = false: //IE
   3. 阻止拖拽的默认行为：return false
3. 事件监听器（兼容）
   1. Target.addEventListener(“click”,fun,false);
      1. 可以绑定多个函数在一个对象上
      2. 执行顺序按照绑定的顺序来
      3. 第三个参数是否使用捕获（反向冒泡），默认false,为冒泡
   2. Target.attachEvent(“onclick”,fun);
      1. 执行顺序按照绑定的反序
   3. 解除事件监听器
      1. Target.removeEventListener(“click”,fun,false);
      2. Target.detachEvent(“onclick”,fun);
4. 事件捕获
   1. 捕获阶段是一个和冒泡阶段完全相反的过程，即事件由祖先元素向子元素传播，和一个石子儿从水面向水底下沉一样，要说明的是在 IE，opera浏览器中，是不存在这个阶段的。从各浏览器提供的注册事件监听的方法中可见一斑，例如适用于ie,opera的attachEvent， 有两个参数，attachEvent(“on”+type,fn)，而适用于所谓标准浏览器的addEventListener则有三个参数，addEventListener(type,fn,boolean)，前面两个参数不用解释，第三个参数boolean，就是决定注册事件发生在捕获阶段还是冒泡阶段.

第二节：应用

1. 掌握兼容问题
2. 自定义右键菜单
3. 掌握事件监听的封装
4. 掌握冒泡和捕获的区别

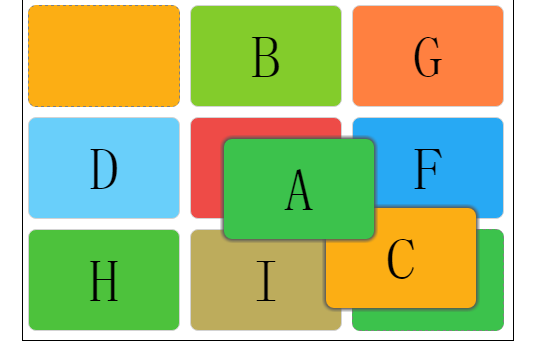
第三节

1. 事件委托机制
   1. 什么是事件委托：通俗的讲，事件就是onclick，onmouseover，onmouseout，等就是事件，委托呢，就是让别人来做，这个事件本来是加在某些元素上的，然而你却加到别人身上来做，完成这个事件。
      1. 利用冒泡的原理，把事件加到父级上，触发执行效果。
   2. 作用：提高性能。  
      <ul id="ul">  
       <li>aaaaaa</li>  
       <li>bbbbbb</li>  
       <li>cccccc</li>  
      </ul>  
      window.onload = function(){  
       var oUl = document.getElementById("ul");  
       var aLi = oUl.getElementsByTagName("li");  
       for(var i = 0; i < aLi.length; i ++){  
       aLi[i].onmouseover = function(){  
       this.style.background = "red";  
       };  
       aLi[i].onmouseout = function(){  
       this.style.background = "";  
       };  
       }  
      };  
        
      //用事件委托的方式来实现这样的效果。  
      window.onload = function(){  
       var oUl = document.getElementById("ul");  
       var aLi = oUl.getElementsByTagName("li");  
       /\*  
       这里要用到事件源：event对象，事件源，不管在哪个事件中，只要你操作的那个元素就是事件源。  
       ie:window.event.srcElement  
       标准下：event.target  
       nodeName:找到元素的标签名  
       \*/  
       oUl.onmouseover = function(evt){  
       var e = evt || window.event;  
       var target = e.target || e.srcElement;  
       if(target.nodeName.toLowerCase() == "li"){  
       target.style.background = "red";  
       }  
       }  
       oUl.onmouseout = function(evt){  
       var e = evt || window.event;  
       var target = evt.target || evt.srcElement;  
       if(target.nodeName.toLowerCase() == "li"){  
       target.style.background = "";  
       }  
       }  
      };
2. 拖拽效果（onmousedown/onmousemove/onmouseup）
   1. 拖拽思路
      1. 给目标元素添加onmousedown事件，拖拽的前提是在目标元素按下鼠标左键
      2. 当onmousedown发生以后，此刻给document添加onmousemove事件，意味着此刻鼠标在网页的移动都将改变目标元素的位置
      3. 在onmousemove事件中，设定目标元素的left和top,公式：目标元素的left=鼠标的clientX – (鼠标和元素的横坐标差，即,offsetX),目标元素的top = 鼠标的clientY - （鼠标和元素的纵坐标差，即offsetY）
      4. 当onmousedown发生以后，此刻给document添加onmouseup事件，意味着此刻鼠标在网页的任意位置松开鼠标，都会放弃拖拽的效果
      5. 在onmouseup事件中，取消document的onmousemove事件即可。
3. JSON对象/构造函数
   1. JSON是一种基于文本的数据交换方式，或者叫做数据描述格式。
   2. JSON的优点：
      1. 基于纯文本，跨平台传递极其简单
      2. JS原生支持，后台语言几乎全部支持
      3. 轻量级数据格式，占用字符数量极少，特别适合互联网传递
      4. 可读性较强，虽然比不上XML那么一目了然，但在合理的依次缩进之后还是很容易识别的
      5. 容易编写和解析，当然前提是你要知道数据结构
   3. JSON的格式：
      1. JSON只有两种数据类型描述符，大括号{}和方括号[],其余英文冒号：是映射符，英文逗号,是分隔符，英文双引号””是定义符。{“key”:”value”,”key”:”value”,……} 或[1,2,3,4,5]
      2. 大括号{}用来描述一组”不同类型的无序键值对集合”(每个键值对可以理解为OOP(面向对象)的属性描述)，方括号[]用来描述一组”相同类型的有序数据集合”(可对应OOP的数组).
      3. 上述两种集合中若有多个子项，则通过英文逗号，进行分隔。
      4. 键值对以英文冒号：进行分隔，并且建议键名都加上英文双引号””，以便于不同语言的解析。
      5. JSON内部常用数据类型是字符串、数字、布尔、日期、null这么几个，字符串必须用双引号引起来，其余的都不用
   4. 构造函数
      1. 使用JS语法创建对象  
         //定义一个函数，作为构造函数  
         function Person(name,sex){  
          this.name = name;  
          this.sex = sex;  
         }  
         //创建一个实例  
         var p = new Person(“张三”,”男”);  
         //输出Person实例  
         alert(p.name);  
         //通过该方式创建的对象是一般的脚本对象
      2. 使用JSON语法创建对象  
         var obj = {name:”张三”,”sex”:”男”};  
         alert(obj.sex);

第四节：应用

1. 鼠标移动轨迹
2. 弹出窗口拖拽
3. 扩展案例：拖拽轨迹回放

第五节：综合应用

1. 九宫格拖拽交换  
   

周六练习

* + 1. 掌握事件流的概念
    2. 掌握兼容的解决方案
    3. 掌握事件委托的应用场景
    4. 滑动条拖拽

