千锋教育
QIAN FENG EDUCATION

做真实的自己，用良心做教育

事件上

H5教学部

目录

做真实的自己，用良心做教育

千锋教育
QIAN FENG EDUCATION

事件介绍

1, 事件的概念

日常生活中事件: 发生并得到处理的操作(即事情来了, 然后处理)

比如：

1, 电话铃声响起(事件发生) --- 需要接电话（处理）

2, 学生举手请教问题(有事了) --- 需要解答（处理）

3, 咱们班有个同学被欺负了(出事了) --- 去给他报仇（处理）

在JS中的事件是: 用户在页面上操作, 然后我们要调用函数来处理.

比如:

1, 点击了登录按钮, 调用登录函数执行登录操作

2, 鼠标拖拽, 调用函数实现拖拽

事件触发：

用户在页面上操作(如点击按钮, 鼠标滑过, 鼠标点击, 鼠标松开, 文本框获得焦点, 失去焦点等), 就是事件触发.

事件介绍

2, 事件的模式

JavaScript有两种事件实现模式: 内联模式, 脚本模式.

内联模式 :

直接在HTML标签中添加事件. 这种模型是最传统简单的一种处理事件的方法。但是这种模式中事件和HTML是混写的, 并没有将JS与HTML分离, 当代码量多了以后,对后期代码维护和扩展很不利.

例如:

```
<input type="button" value="按钮" onclick="alert('hello');" />
```

注意: 单双引号

//执行自定义的JS函数

```
<input type="button"value="按钮" onclick="btnClick();" />
```

注意: 内联模式调用的函数不能放到window.onload里面, 否则会找不到该函数.

事件介绍

脚本模式:

脚本模式能将JS代码和HTML代码分离, 符合代码规范.

使用脚本模式我们需要先获取到元素节点对象, 再针对该节点对象添加事件; 如我们可以采用三种方式来获得节点对象: getElementById(), getElementsByTagName(), getElementsByName()

例如:

```
var box = document.getElementById('box');
添加事件方式一： 通过匿名函数，可以直接触发对应的代码 (推荐)
box.onclick = function() {  //给box节点对象添加点击事件onclick
        console.log('Hello world!');
};

添加事件方式二： 通过指定的函数名赋值的方式 来执行函数
box.onclick = func;    //注意这里不能写成func()
function func() {        //给box节点对象添加点击事件onclick
        console.log('Hello world!');
};
```

事件介绍

事件处理由三个部分组成：
1, 触发事件的元素节点对象
2, 事件处理函数
3, 事件执行函数

例如：单击文档任意处。

```
document.onclick = function(){
        console.log('单击了文档页面的某一个地方');
};
```

在上面的程序中：
document : 是触发事件的对象, 表示触发事件的元素所在区域;
onclick : 表示一个事件处理函数(on+事件类型click)
function(){} : 匿名函数是被执行的函数, 用于触发事件后执行;

所有的事件处理函数都会都有两个部分组成，on+事件类型;
例如：onclick事件处理函数就是由on加上click;

注意: 事件处理函数一般都是小写字母

事件的分类

事件的分类

JavaScript 可以处理的事件种类有三种：鼠标事件, 键盘事件和 HTML事件.

1，鼠标事件

鼠标事件，页面所有元素都可触发鼠标事件;

click： 当单击鼠标按钮并在松开时触发

```
onclick = function() {
        console.log('单击了鼠标');
};
```

dblclick： 当双击鼠标按钮时触发。

```
ondblclick = function() {
        console.log('双击了鼠标');
};
```

mousedown：当按下了鼠标还未松开时触发。

```
onmousedown = function() {
        console.log('按下鼠标');
};
```

事件的分类

mouseup： 释放鼠标按钮时触发。

```
onmouseup = function() {
        console.log('松开了鼠标');
};
```

mouseover：当鼠标移入某个元素的那一刻触发。

```
onmouseover = function() {
        console.log('鼠标移入了');
};
```

mouseout：当鼠标刚移出某个元素的那一刻触发。

```
onmouseout = function() {
        console.log('鼠标移出了');
};
```

mousemove：当鼠标指针在某个元素上移动时触发。

```
onmousemove = function() {
        console.log('鼠标移动了');
};
```

事件的分类

mouseenter：当鼠标移入某个元素的那一刻触发。

```
onmouseenter = function() {
        console.log('鼠标移入了');
};
```

mouseleave：当鼠标刚移出某个元素的那一刻触发。

```
onmouseleave = function() {
        console.log('鼠标移出了');
};
```

mouseover和 mouseenter的区别是:
mouseover: 元素的子元素移入也会触发事件
mouseenter : 元素的子元素移入不会触发事件

事件的分类

示例:
1, 有一块空白区域,
当鼠标移动到区域内,显示"亲爱的, 我爱你",
当我鼠标移开的时候,显示"对不起, 开玩笑",
当我鼠标不停的在区域内移动的时候, 变换颜色

事件的分类

2. 键盘事件

键盘事件，在键盘上按下键时触发的事件;
(一般由window对象或者document对象调用)

keydown：当用户按下键盘上某个键触发，如果按住不放，会重复触发。

```
window.onkeydown = function() {
        console.log(按下了键盘上的某个键);
};
```

keypress：当用户按下键盘上的字符键触发，如果按住不放，会重复触发

```
window.onkeypress = function() {
        console.log('按下了键盘上的字符键');
};
```

keyup：当用户释放键盘上的某个键触发。

```
window.onkeyup = function() {
        console.log(松开键盘上的某个键);
};
```

事件的分类

3. HTML事件
    HTML事件，跟HTML页面相关的事件;

load：当页面完全加载后触发

```
window.onload = function() {
    console.log('页面已经加载完毕');
};
```

unload：当页面完全卸载后触发

```
window.onunload = function() {
    console.log('页面已经卸载完毕');
};
```

select：当用户选择文本框(input 或 textarea)中的内容触发。

```
input.onselect = function() {
    console.log('选择了文本框中的内容');
};
```

change：当文本框(input 或 textarea)内容改变且失去焦点后触发。

```
input.onchange = function() {
    console.log('文本框中内容改变了');
};
```

事件的分类

focus：当页面或者元素获得焦点时触发。

```
input.onfocus = function() {
        console.log('文本框获得焦点');
};
```

blur：当页面或元素失去焦点时触发。

```
input.onblur = function() {
        console.log('文本框失去焦点');
};
```

submit：当用户点击提交按钮在<form>元素节点上触发。

```
form.onsubmit = function() {
        console.log( ‘提交form表单’ );
};
```

reset：当用户点击重置按钮在<form>元素节点上触发。

```
form.onreset = function() {
        console.log('重置form表单');
};
```

scroll：当用户滚动带滚动条的元素时触发。

```
window.onscroll= function() {
        console.log('滚动了滚动条了');
};
```

事件对象Event

1, 事件对象(event对象) 是什么?

event对象是在触发事件时, 浏览器会通过函数把事件对象作为参数传递过来, 在事件触发执行函数时一般会得到一个隐藏的参数, 该参数也是放在arguments数组中

```
//普通函数的arguments
function func() {
    console.log(arguments.length); //1, 得到一个传递的参数
}
func( “hello” );

//事件绑定的执行函数
box.onclick = function(){
     console.log(arguments.length); //1, 得到一个隐藏参数
};
```

通过上面两组函数中, 我们发现, 通过事件绑定的执行函数是可以得到一个隐藏参数的. 说明浏览器会自动分配一个参数，这个隐藏参数其实就是event对象(事件对象).

事件对象Event

2，获取事件对象

```
box.onclick = function() {
      console.log(arguments[0]); //获得该事件对象([object MouseEvent])
};
```

我们还可以使用更简单的获取事件对象的方式: 通过给函数添加一个参数

```
//接受事件对象, 名称不一定非要evt(这里的evt是形参,也可以自己给定其他名称)
box.onclick = function(evt){
    console.log(evt);   //[object MouseEvent]
};
```

事件对象Event

通过事件的执行函数传入的event对象(事件对象) 不是在所有浏览器都有值，在IE浏览器上event对象并没有传过来, 这里我们要用window.event来获取, 而在火狐浏览器上window.event无法获取, 而谷歌浏览器支持event事件传参和window.event两种, 为了兼容所有浏览器, 我们使用以下方式来得到event事件对象:

```
box.onclick = function(evt){
    var e= evt || window.event; //获取到event对象(事件对象)
    console.log(e);
};
```

其中window.event中的window可以省略, 最终我们可以写成:

```
box.onclick = function(evt){
    var e= evt || event; //获取到event对象(事件对象)
    console.log(e);
};
```

注意: evt||event不要倒过来写

事件对象Event

3, 事件对象的属性

button: 鼠标按下了哪个键

值	说明
0	表示主鼠标按钮(常规一般是鼠标左键)
1	表示中间的鼠标按钮(鼠标滚轮按钮)
2	表示次鼠标按钮(常规一般是鼠标右键)

例如:

```
document.onclick = function(evt) {
        var e = evt || event;
        console.log(e.button);
};
```

事件对象Event

clientX: 浏览器可视区域的x坐标
clientY: 浏览器可视区域的y坐标
pageX: 浏览器内容区域的x坐标
pageY: 浏览器内容区域的y坐标
screenX: 显示器屏幕的x坐标
screenY: 显示器屏幕的y坐标
offsetX: 鼠标点击的元素位置距离元素左边界的x坐标
offsetY: 鼠标点击的元素位置距离元素上边界的y坐标

例如:

```
document.onmousedown= function(evt) {
        var e = evt || event;
        console.log(e.clientX + ',' + e.clientY);
        console.log(e.screenX + ',' + e.screenY);
        console.log(e.pageX +   “,”   + e.pageY);
};
```

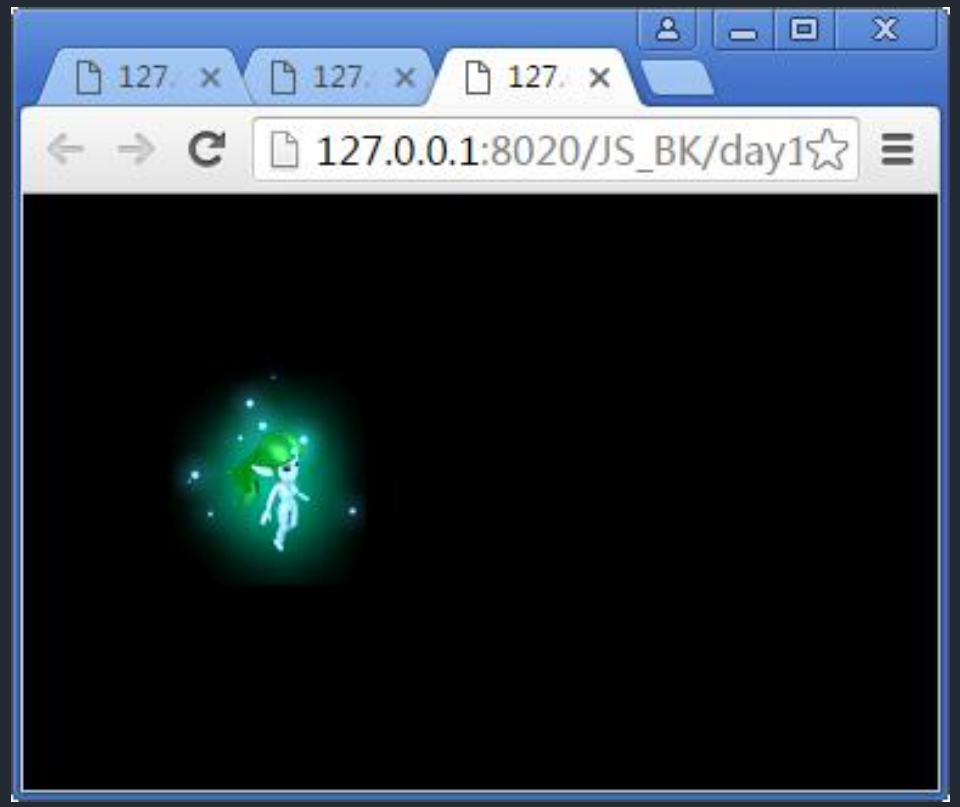
事件对象Event

target: 目标对象,存放绑定事件的元素节点对象

```
document.onclick = function(evt) {
    var e = evt || event;
    console.log("document: " + e.target);
}
box.onclick = function(evt) {
    var e = evt || event;
    console.log("box: " + e.target);
}
oInput.onclick = function(evt) {
    var e = evt || event;
    console.log("input: " + e.target);
}
```

事件对象Event

示例:
1, 鼠标点击某处, 让精灵移动到该处 (如下图)
2, 鼠标移动时, 让精灵跟随鼠标移动



事件对象Event

4, 键盘事件的组合键

键盘上的某些键可以配合鼠标来触发一些特殊的事件. 如: Shfit, Ctrl, Alt.

属性	说明
shiftKey	判断是否按下了 Shfit 键
ctrlKey	判断是否按下了 ctrlKey 键
altKey	判断是否按下了 alt 键

事件对象Event

键码：keyCode属性

所有按键(包括功能键control, alt,shift, tab, 方向键等, 不包括亮度,音量..的按键)在发生keydown和keyup 事件时，event对象的 keyCode属性中会包含一个代码，与键盘上一个特定的键对应。对数字字母字符集，keyCode属性的值与 ASCII 码中对应.

```
document.onkeydown = function(evt) {
    var e = evt || event;
    console.log(e.keyCode);  //按任意键，得到相应的 keyCode
};
```

注意: 大写字母或小写的编码相同, 都为大写字母。

事件对象Event

字符编码: charCode属性

Firefox, Chrome和Safari的event对象支持charCode属性, 只有按下字符键并且使用keypress事件时才会得到charCode, 这个charCode的值和ASCII码对应, 和keyCode也类似, 但是charCode的字母区分大小写. (字符键: 数字, 字母(区分大小写), 字符, 空格, 回车)

```
document.onkeypress = function(evt) {
    var e = evt || event;
    console.log(e.charCode);
}
```

注：可以使用 String.fromCharCode()将 ASCII 编码转换成实际的字符

事件对象Event

练习:
有一个红色的div块
1, 如果我按下ctrl+c变换颜色
2, 如果我按下ctrl + shift + r 重置颜色,恢复初始颜色
3, 如果我按下向上箭头,向上移动, 同理还可以向下,左,右移动
4, 如果我按下ctrl + 上下左右,走的步数变大

事件中this

在JS事件中, this表示触发事件的元素节点对象;

```
var box = document.getElementById('box');
box.onclick = function() {
    console.log(this.nodeName);  //this表示box对象
};
```

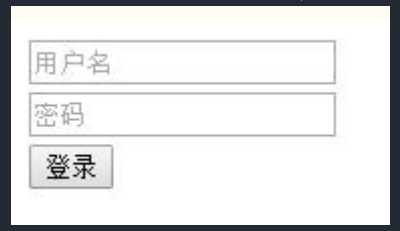
通过for循环添加事件, 使用this

```
var aInput = document.getElementsByTagName('input');
for (var i=0; i<aInput.length; i++) {
    aInput[i].index = i;
    aInput[i].onclick = function() {
        console.log(this.index);
    };
}
```

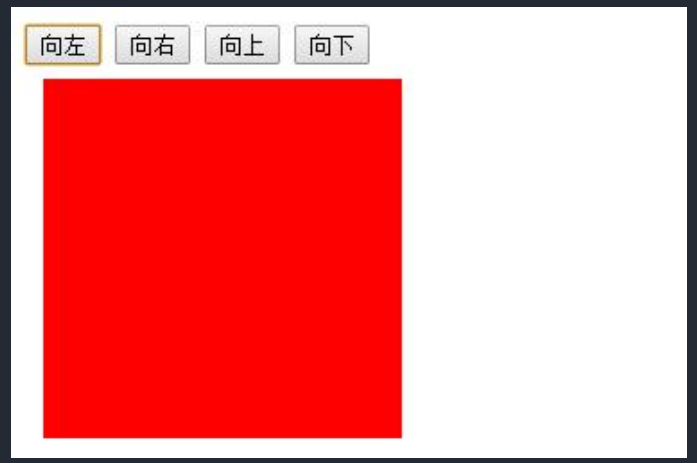
练习

1, 如下图, 在输入框中输入用户名和密码,

- 当鼠标失去焦点时: 检测用户名长度至少6位, 且只能为数字和字母;检测密码长度至少8位
- 给登录按钮添加点击事件: 点击后弹出用户名和密码



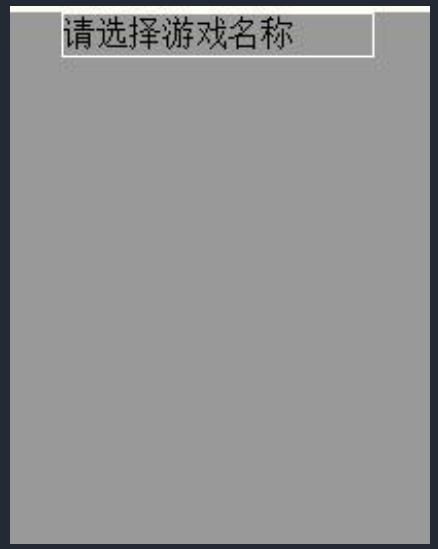
2, 有一个红色div块，有四个button, 上下左右, 当我点击对应的按钮,这个块就进行对应的行走。

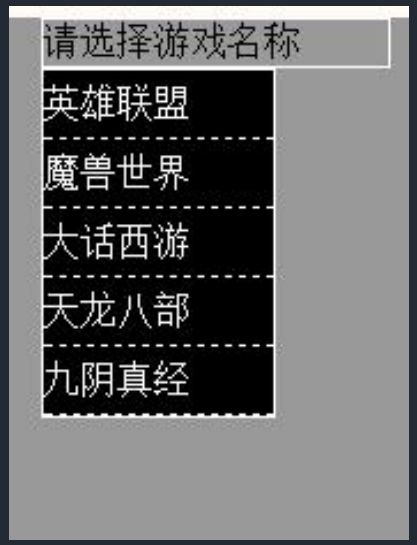


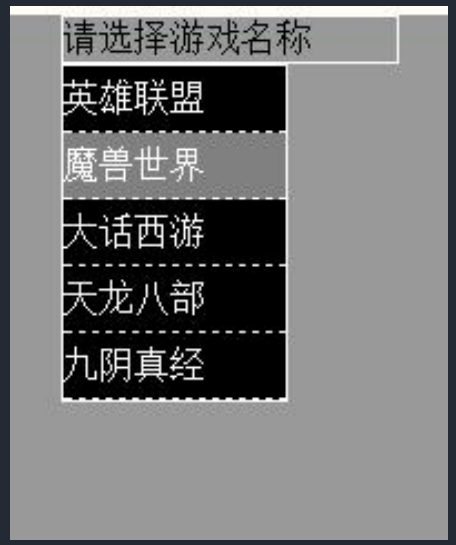
练习

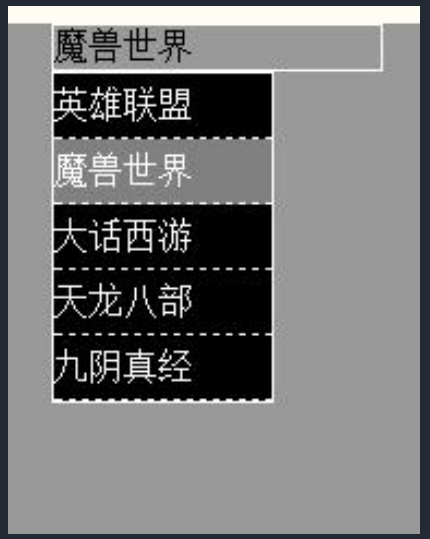
3, 制作下拉菜单:

1, 最开始效果如左图1
2, 鼠标滑过”请选择游戏名称”区域时,效果如图2
3, 鼠标滑过下拉选项区域时, 让下拉选项可以继续显示,移开后隐藏
4, 鼠标在选项中滑过时, 显示效果图3
5, 选择某一项, 将顶部的名称改成你选择的游戏名称









练习

4, 制作如下图效果: 当鼠标滑过小图片时, 让大图片也显示该图

THANK YOU

千锋教育
QIAN FENG EDUCATION

做真实的自己，用良心做教育