02jq的事件

今日内容：

- 复习DOM操作
- 作业评讲
- 复习属性操作
- 事件的应用
- 事件机制
    - 页面载入事件
    - 绑定事件
    - 切换事件
    - 移除事件
    - 其他事件
    - 案例练习
- jq动画和特效
    - 显示和隐藏
    - 滑动特效
    - 淡入淡出
    - 自定义动画
    - 案例练习

1、事件机制

jQuery的事件机制，指的是：jQuery对JavaScript操作DOM事件的封装，包括了：事件绑定、事件解绑、事件触发。

1.4.1jQuery事件的发展历程（了解）

简单事件绑定 >> bind事件绑定 >> delegate事件绑定 >> on【重点】

●简单事件绑定：

click(handler) 单击事件

blur(handler) 失去焦点事件

mouseenter(handler) 鼠标进入事件

mouseleave(handler) 鼠标离开事件

dbclick(handler) 双击事件

change(handler) 改变事件，如：文本框值改变，下来列表值改变等

focus(handler) 获得焦点事件

keydown(handler) 键盘按下事件

mouseover和mouseenter事件的区别

不论鼠标指针穿过被选元素或其子元素，都会触发 mouseover 事件，对应 mouseout。

只有在鼠标指针穿过被选元素时，才会触发 mouseenter 事件。对应 mouseleave。

这样的话，mouseenter子元素不会反复触发事件，否则在IE中经常有闪烁情况发生。

●bind方式（不推荐，1.7以后的jQuery版本被on取代）

作用：给匹配到的元素直接绑定事件

```
// 绑定单击事件处理程序
第一个参数：事件类型
第二个参数：事件处理程序
$("p").bind("click mouseenter", function(e){
```

```
    //事件响应方法
});
```

比简单事件绑定方式的优势：

1．可以同时绑定多个事件，比如：bind(“mouseenter mouseleave”, function(){})

缺点：要绑定事件的元素必须存在文档中。

3 delegate方式（特点：性能高，支持动态创建的元素）

作用：给匹配到的元素绑定事件，对支持动态创建的元素有效

```
// 第一个参数：selector，要绑定事件的元素
// 第二个参数：事件类型
    // 第三个参数：事件处理函数
$(".parentBox").delegate("p", "click", function(){
    //为 .parentBox下面的所有的p标签绑定事件
});
```

与前两种方式最大的优势：减少事件绑定次数提高效率，支持动态创建出来的元素绑定事件！

1.4.2on方式（最现代的方式，兼容zepto(移动端类似jQuery的一个库)，强烈建议使用的方式）（重点）

jQuery1.7版本后，jQuery用on统一了所有的事件处理的方法

作用：给匹配的元素绑定事件，包括了上面所有绑定事件方式的优点

语法：

```
// 第一个参数：events，绑定事件的名称可以是由空格分隔的多个事件（标准事件或者自定义事件）
// 第二个参数：selector, 执行事件的后代元素
```

```
// 第三个参数：data，传递给处理函数的数据，事件触发的时候通过event.data来使用
// 第四个参数：handler，事件处理函数
$(selector).on(events[,selector][,data],handler);
```

```
   // 表示给$(selector)绑定事件，当必须是它的内部元素span才能执行这个事件
$(selector).on( "click",“span”, function() {});


    // 绑定多个事件
    // 表示给$(selector)匹配的元素绑定单击和鼠标进入事件
    $(selector).on(“click mouseenter”, function(){});
```

1.4.3事件解绑

●unbind() 方式

作用：解绑 bind方式绑定的事件

```
$(selector).unbind(); //解绑所有的事件
$(selector).unbind(“click”); //解绑指定的事件
```

●undelegate() 方式

作用：解绑delegate方式绑定的事件

```
$( selector ).undelegate(); //解绑所有的delegate事件
$( selector).undelegate( “click” ); //解绑所有的click事件
```

off解绑**on**方式绑定的事件（重点）

```
// 解绑匹配元素的所有事件
$(selector).off();
// 解绑匹配元素的所有click事件
```

```
$(selector).off(“click”);
// 解绑所有代理的click事件，元素本身的事件不会被解绑
$(selector).off( “click”, “**” );
```

1.4.4事件触发

简单事件触发

```
$(selector).click(); //触发 click事件
```

trigger方法触发事件

```
$(selector).trigger(“click”);
```

triggerHandler触发 事件响应方法，不触发浏览器行为

比如:文本框获得焦点的默认行为

```
$(selector).triggerHandler(“focus”);
```

其他事件：

one（ ）：绑定触发一次的事件

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
        <script src="jquery-1.10.1.min.js" type="text/javascript"
charset="utf-8"></script>
        <script type="text/javascript">
        $(function(){
            function btn_click(){
                    $(this).val('5665665');
```

```
                }

                $('#btn').one('click',btn_click);
        });

        </script>
    </head>
    <body>
        <input type="button" name="btn" id="btn" value="点击显示号码"
/>
    </body>
</html>
```

trigger():自动触发指定的事件

在前端开发中，有时候我们需要页面加载完的时候，让文本框某个内容被选中，某个按钮处于焦点中，这个时候就可以用这个事件轻松实现。

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
        <script src="jquery-1.10.1.min.js" type="text/javascript"
charset="utf-8"></script>
        <script type="text/javascript">
            $(function(){
                $('input').trigger('select');//自动选择文本框

                $('input').bind('btn_click',function(){
                    $('.name').html($(this).val());
                });

                $('input').trigger('btn_click');
            });
        </script>
    </head>
    <body>
        姓名: <input type="text" value="隔壁老王" />
        <div class="name">
```

```
22
23          </div>
24      </body>
25  </html>
```

事件的应用1：邮箱验证

```
1   <!DOCTYPE html>
2   <html>
3       <head>
4           <meta charset="UTF-8">
5           <title></title>
6           <style type="text/css">
7               .yes{
8                   color: green;
9               }
10              .no{
11                  color: red;
12              }
13          </style>
14          <script src="jquery-1.10.1.min.js" type="text/javascript"
    charset="utf-8"></script>
15          <script type="text/javascript">
16              $(function(){
17                  $('#tex').trigger('focus');//页面加载后自动聚焦文本框
18
19                  $('#tex').focus(function(){
20                      $('span').html('请输入正确的邮
    箱').attr('class','yes');
21                  });
22
23                  $('#tex').blur(function(){
24                      var oVal=$('#tex').val();
25                      if(oVal.length==0){
26                          $('span').html('邮箱不能为
    空').attr('class','no');
27                      }
28                      else if(!chkemail(oVal)){
```

```
                        $('span').html('邮箱格式不正
确').attr('class','no');
                    }
                    else{
                        $('span').html('输入正确').attr('class','yes');
                    }
                });

                function chkemail(str){
                    var re=/\w+@[a-z0-9]+\.[a-z]+/i;
                    if(re.test(str)){
                        return true;
                    }
                    else{
                        return false;
                    }
                }


            });
        </script>
    </head>
    <body>
        <form action="" method="post">
            邮箱:
            <input type="text" id="tex" value="" />
            <span ></span>
        </form>
    </body>
</html>
```

补充知识点：

- ev　pageX　which
- preventDefault　stopPropagation

```
<!DOCTYPE html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
```

```
<title>无标题文档</title>

<script type="text/javascript" src="jquery-1.10.1.min.js"></script>
<script>

$(function(){

    /*$('div').click(function(ev){

        //ev : event对象

        //ev.pageX(相对于文档的) : clientX(相对于可视区)

        //ev.which : keyCode

        ev.preventDefault();  //阻止默认事件

        ev.stopPropagation();  //阻止冒泡的操作

        return false;   //阻止默认事件 + 阻止冒泡的操作

    });*/



});



</script>
</head>

<body>
<div>div</div>
<span>span</span>
</body>
</html>
```

- offset() position()

```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>无标题文档</title>
<style>
*{ margin:0; padding:0;}
#div1{ width:200px; height:200px; background:red; overflow:hidden; margin:20px; position:absolute;}
#div2{ width:100px; height:100px; background:yellow; margin:30px; position:absolute; left:25px;}
</style>
<script type="text/javascript" src="jquery-1.10.1.min.js"></script>
<script>

$(function(){

    //div2.offsetLeft

    //alert( $('#div2').offset().left );  //获取到屏幕的左距离

    alert( $('#div2').position().left );  //到有定位的父级的left值,把当前元素转化成类似定位的形式


});



</script>
</head>

<body>
<div id="div1">
    <div id="div2"></div>
</div>
</body>
</html>
```

- offsetParent()

```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>无标题文档</title>
<style>
*{ margin:0; padding:0;}
#div1{ width:200px; height:200px; background:red; overflow:hidden; margin:20px; position:absolute;}
#div2{ width:100px; height:100px; background:yellow; margin:30px; position:absolute; left:25px;}
</style>
<script type="text/javascript" src="jquery-1.10.1.min.js"></script>
<script>

$(function(){

    //parent() : 获取父级
    //offsetParent() : 获取有定位的父级

    //$('#div2').parent().css('background','blue');

    $('#div2').offsetParent().css('background','blue');

});



</script>
</head>

<body>
<div id="div1">
    <div id="div2"></div>
</div>
</body>
</html>
```

- size()
- each()

```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>无标题文档</title>
<style>
*{ margin:0; padding:0;}
#div1{ width:200px; height:200px; background:red; overflow:hidden; margin:20px; position:absolute;}
#div2{ width:100px; height:100px; background:yellow; margin:30px; position:absolute; left:25px;}
</style>
<script type="text/javascript" src="jquery-1.10.1.min.js"></script>
<script>

$(function(){

    //alert( $('li').size() );  //4 像length

    $('li').each(function(i,elem){   //一参：下标 二参 ： 每个元素

        $(elem).html(i);

    });

});
</script>
</head>

<body>


<ul>
    <li></li>
    <li></li>
    <li></li>
    <li></li>
</ul>
</body>
</html>
```

拖拽：

```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>无标题文档</title>
<style>
div{ width:100px; height:100px; background:red; position:absolute;}
</style>
<script type="text/javascript" src="jquery-1.10.1.min.js"></script>
<script>

$(function(){

    var disX = 0;
    var disY = 0;

    $('div').mousedown(function(ev){

        disX = ev.pageX - $(this).offset().left;
        disY = ev.pageY - $(this).offset().top;

        $(document).mousemove(function(ev){

            $('div').css('left',ev.pageX - disX);
            $('div').css('top',ev.pageY - disY);

        });

        $(document).mouseup(function(){

            $(document).off();

        });

        return false;

    });

});

```

```
40
41
42 </script>
43 </head>
44
45 <body>
46 <div></div>
47 </body>
48 </html>
```

- hover()：鼠标移入和移开合体
  - show() hide()：显示隐藏：宽、高、透明度一起变化
  - fadeIn() fadeOut()：淡入淡出
  - fadeTo()：淡入淡出
  - slideDown() slideUp()：滑动

```
1  <!DOCTYPE html >
2  <head>
3  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
4  <title>无标题文档</title>
5  <style>
6  #div1{ width:100px; height:100px; background:red;}
7  #div2{ width:100px; height:100px; background:yellow;}
8  </style>
9  <script type="text/javascript" src="jquery-1.10.1.min.js"></script>
10 <script>
11
12 $(function(){
13
14     $('#div1').hover(function(){
15
16         //$(this).css('background','blue');
```

```
        //$('#div2').hide(3000);
        //$('#div2').fadeOut(1000);  //默认400
        //$('#div2').slideUp();
        $('#div2').fadeTo(1000,0.5);
    },function(){
        //$(this).css('background','red');
        //$('#div2').show(3000);
        //$('#div2').fadeIn(1000);
        //$('#div2').slideDown();
        $('#div2').fadeTo(1000,1);
    });
});


</script>
</head>
<body>
<div id="div1"></div>
<div id="div2"></div>
</body>
</html>
```

jQuery动画是什么？

jQuery提供的一组网页中常见的动画效果，这些动画是标准的、有规律的效果；同时还提供给我们了自定义动画的功能。

1.5.1 隐藏显示动画

1 show方法

作用：让匹配的元素展示出来

```
// 用法一：
//

// 用法二：
// 参数为字符串类型，是jQuery预设的值，共有三个，分别是：slow、normal、fast
/* 跟用法一的对应关系为： */
/* slow：600ms、normal：400ms、fast：200ms */
$(selector).show(“slow”);

// 用法三：
// 参数一可以是数值类型或者字符串类型
// 参数二表示：动画执行完后立即执行的回调函数
$(selector).show(2000, function() {});

// 用法四：
// 不带参数，作用等同于 css(“display”,” block”)
/* 注意：此时没有动画效果 */
$(selector).show();
```

注意：

jQuery预设的三组动画效果的语法几乎一致：参数可以有两个，第一个是动画的执行时长，第二个是动画执行完成后的回调函数。

第一个参数是：可以是指定字符或者毫秒数

2　hide方法

作用：让匹配元素隐藏掉

用法参考show方法

```
$(selector).hide(1000); // 1000表示什么？
$(selector).hide( “slow” );
$(selector).hide(1000, function(){});
$(selector).hide();
```

1.5.2 滑入滑出动画

1滑入动画效果（联想：生活中的卷帘门）

作用：让元素以下拉动画效果展示出来

```
$(selector).slideDown(speed,callback);
```

注意：省略参数或者传入不合法的字符串，那么则使用默认值：400毫秒（同样适用于fadeIn/slideDown/slideUp）

```
$(selector).slideDown();
```

2　滑出动画效果

作用：让元素以上拉动画效果隐藏起来

```
$(selector).slideUp(speed,callback);
```

3滑入滑出切换动画效果

```
$(selector).slideToggle(speed,callback);
```

参数等同与1.5.1　隐藏和显示

1.5.3 淡入淡出动画

1　淡入动画效果

作用：让元素以淡淡的进入视线的方式展示出来

```
$(selector).fadeIn(speed, callback);
```

2　淡出动画效果

作用：让元素以渐渐消失的方式隐藏起来

```
$(selector).fadeOut(1000);
```

3淡入淡出切换动画效果

作用：通过改变不透明度，切换匹配元素的显示或隐藏状态

```
$(selector).fadeToggle('fast',function(){});
```

参数等同与1.5.1　隐藏和显示

4改变不透明度到某个值

与淡入淡出的区别：淡入淡出只能控制元素的不透明度从　完全不透明　到完全透明；而fadeTo可以指定元素不透明度的具体值。并且时间参数是必需的！

作用：调节匹配元素的不透明度

```
// 用法有别于其他动画效果
// 第一个参数表示：时长
// 第二个参数表示：不透明度值，取值范围：0-1
$(selector).fadeTo(1000, .5); //  0全透，1全不透

// 第一个参数为0，此时作用相当于：.css(“opacity”，.5);
$(selector).fadeTo(0, .5);
```

jQuery提供的动画使用方法总结：

jQuery提供的动画效果总结

1. 常用方式（以show为例）：

```
show();
show("fast");
show(1000);
show(1000, function() {
    /* 动画执行完成后，要执行的处理 */
});

// 参数一：动画执行时长
// 参数二：不透明度
fadeTo(1000, 0.5);
```

2. 共同点：

   使用方式相同，参数的作用相同。记住一个效果的使用方式其他的也就会用了。（fadeTo()方法除外）

3. 展示隐藏：

   show() fadeIn() slideDown() 表示展示

   hide() fadeOut() slideUp() 表示隐藏

4. 实现效果控制的样式：

   show()/hide()的动画效果，改变的是：宽高和不透明度

   fadeIn()/fadeOut()/fadeToggle()/fadeTo()的动画效果，改变的是：不透明度

   slideDown()/slideUp()/slideToggle()的动画效果，改变的是：高度

有规律的体现：

jQuery提供的这几个动画效果控制的CSS属性包括：高度、宽度、不透明度。{height:400px; width:300px; opacity:.4;}

这三个CSS属性的共性是：属性值只有一个，并且这个值是数值（去掉单位后）。

1.5.4 自定义动画

注意：所有能够执行动画的属性必须只有一个数字类型的值。

比如：要改变字体大小，要使用：fontSize（font-size），不要使用：font

动画支持的属性：

http://www.w3school.com.cn/jquery/effect_animate.asp

作用：执行一组CSS属性的自定义动画

```
// 第一个参数表示：要执行动画的CSS属性（必选）
// 第二个参数表示：执行动画时长（可选）
// 第三个参数表示：动画执行完后立即执行的回调函数（可选）
$(selector).animate({params},speed,callback);
```

1.5.5 停止动画

stop()

作用：停止当前正在执行的动画

为什么要停止动画？

如果一个以上的动画方法在同一个元素上调用，那么对这个元素来说，后面的动画将被放到效果队列中。这样就形成了动画队列。（联想：排队进站）

动画队列里面的动画不会执行，直到第一个动画执行完成。

```
// 第一个参数表示是否清空所有的后续动画
// 第二个参数表示是否立即执行完当前正在执行的动画
$(selector).stop(clearQueue,jumpToEnd);
```

解释：

当调用stop()方法后，队列里面的下一个动画将会立即开始。但是，如果参数clearQueue被设置为true，那么队列面剩余的动画就被删除了，并且永远也

不会执行。

如果参数jumpToEnd被设置为true，那么当前动画会停止，但是参与动画的每一个CSS属性将被立即设置为它们的目标值。比如：slideUp()方法，那么元素会立即隐藏掉。如果存在回调函数，那么回调函数也会立即执行。

注意：如果元素动画还没有执行完，此时调用sotp()方法，那么动画将会停止。并且动画没有执行完成，那么回调函数也不会被执行。

常用方式：

```
$(selector).stop();
```

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>无标题文档</title>
<style>
#div1{ width:100px; height:100px; background:red;}
#div2{ width:100px; height:100px; background:red; margin-top:10px;}
</style>
<script src="jquery-1.10.1.min.js"></script>
<script>

//animate() :
//第一个参数 : {} 运动的值和属性

//第二个参数 : 时间(运动快慢的)  默认 : 400

//第三个参数 : 运动形式 只有两种运动形式 ( 默认 : swing(慢快慢) linear(匀速) )

//第四个参数 :  回调函数

$(function(){

```

```
    $('#div1').click(function(){

        $(this).animate({width : 300 , height : 300} , 4000 , 'linear',function(){
            alert(123);
        });

        $('#div2').animate({width : 300 , height : 300} , 4000 , 'swing');

    });


});

</script>
</head>

<body>
<div id="div1"></div>
<div id="div2"></div>

</body>
</html>
```

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>无标题文档</title>
<style>
#div1{ width:100px; height:100px; background:red;}
#div2{ width:100px; height:100px; background:red; margin-top:10px;}
</style>
<script src="jquery-1.10.1.min.js"></script>
<script>

//animate() :
//第一个参数 : {} 运动的值和属性

```

```
//第二个参数 : 时间(运动快慢的)  默认 : 400

//第三个参数 : 运动形式 只有两种运动形式 ( 默认 : swing(慢快慢) linear(匀速) )

//第四个参数 :  回调函数

$(function(){

    $('#div1').click(function(){

        /*$(this).animate({width : 300} , 2000 , 'linear',function(){
            $(this).animate({height : 300});
        });*/

        $(this).animate({width : 300} , 2000).delay(1000).animate({height : 300} , 2000);

    });


    $('#div2').click(function(){

        //$('#div1').stop();   //默认 : 只会阻止当前运动

        //$('#div1').stop(true); //阻止后续的运动

        //$('#div1').stop(true,true); //可以立即停止到指定的目标点

        $('#div1').finish();  //立即停止到所有指定的目标点

    });


});

</script>
</head>

<body>
<div id="div1"></div>
<div id="div2"></div>
```

```
56
57 </body>
58 </html>
```