jQuery

通过 jQuery，您可以选取（查询，query） HTML 元素，并对它们执行操作。

引入：<script src="js/jquery-1.11.0.min.js"></script>

ready() ：当 DOM（文档对象模型） 已经加载，并且页面（包括图像）已经完全呈现时，会发生 ready 事件。

在<script>脚本中进行编写：

```
$(document).ready(function(){

});
```

或者简写为：

```
$(function(){

});
```

二. jQuery 语法

jQuery 语法是为 HTML 元素的选取而编制的，可以对元素执行某些操作。

$(selector).action();

美元符号$定义 jQuery;

选择符(selector)“查询”和“查找”HTML 元素;

jQuery 的 action()执行对元素的操作;

例如：

```
<script>
    /* 文档就绪函数 ready*/
    $(document).ready(function(){
        /* 点击函数 click */
        $("p").click(function(){
            //对元素做操作
        });
    });
</script>
```

三. jQuery 选择器

选择器允许您对元素组或单个元素进行操作。

通过 jQuery 选择器获得到的对象是 jQuery 对象。

1.使用*;　　如：$("*")　　选择所有元素

2.使用#id;　如：$("#lastname") 选择 id="lastname" 的元素

3.使用.class;　　如：$(".intro") 选择所有 class="intro" 的元素
4.使用.class.class; 如：$(".intro.demo")所有 class="intro"且 class="demo"的元素
$(".b2.b21").css("border",'1px solid blue');　设置
var a = $(".d1").css("background-color");　获得

5.使用 element;　如：$("p")　所有 <p> 元素

6.使用:first;　　如：$("p:first")　　第一个 <p> 元素
其中:冒号前面可以用选择器：class 或 id 或元素都可以。

7.使用:last;　　如：$("p:last")　　最后一个 <p> 元素
其中:冒号前面可以用选择器：class 或 id 或元素都可以。

8.使用:even;　　如：$("tr:even")　　所有偶数 <tr> 元素

9.使用:odd;　　如：$("tr:odd")　　所有奇数 <tr> 元素

10.使用 element element;　　如：$("ul li") 列表中 ul 中的所有 li 都被选中

11.使用:eq(index);　　如：$("ul li:eq(3)") 列表中的第四个元素（index 从 0 开始）

12.使用:gt(no); 如：$("ul li:gt(1)")列出 index 大于 1 的元素（index 从 0 开始）

13.使用:lt(no);　　如：$("div:lt(2)")列出 div 小于 2 的元素（index 从 0 开始）

14.使用:empty;　　如：$("div:empty")　div 块下无子（元素）节点的所有元素

15.使用:not(selector); 如：$("td:not(:empty)")　　所有不为空的表格中的 td 元素

16.使用:header;　　如：$(":header")　　所有标题元素 <h1> - <h6>

17.使用:animated;　所有动画元素

```
/*function aniDiv(){
    $("div:eq(3)").animate({width:300},"slow");
    $("div:eq(3)").animate({width:100},"slow",aniDiv);
}
aniDiv();
$(":animated").css("background-color","blue");*/
```

18.使用:contains(text); 如：$("p:contains('哈哈哈')")包含指定字符串的 p 元素

19.使用:hidden;　　如：$("p:hidden")　找出所有隐藏的 <p> 元素

20.使用:visible;　$("table:visible") 找出所有可见的表格

21.使用 s1,s2,s3　　$("th,td,.intro")　所有带有匹配选择的元素

22.
[attribute]　$("[href]")　　所有带有 href 属性的元素
[attribute=value]　$("[href='#']")　　所有 href 属性的值等于 "#" 的元素
[attribute!=value] $("[href!='#']")　　所有 href 属性的值不等于 "#" 的元素
[attribute$=value] $("div[class$=d2]")所有 div 块中有 class 属性为 d2 的元素

23.
:input　$(":input")　　所有 <input> 元素
:text　$(":text") 所有 type="text" 的 <input> 元素
:password　$(":password") 所有 type="password" 的 <input> 元素
:radio $(":radio")　　所有 type="radio" 的 <input> 元素
:checkbox　$(":checkbox") 所有 type="checkbox" 的 <input> 元素
:submit　　$(":submit")　所有 type="submit" 的 <input> 元素
:reset $(":reset")　　所有 type="reset" 的 <input> 元素
:button　　$(":button")　所有 type="button" 的 <input> 元素
:file　$(":file") 所有 type="file" 的 <input> 元素

24.
:enabled　$(":enabled")　所有激活的 input 元素
:disabled　$(":disabled") 所有禁用的 input 元素
:selected　$(":selected") 所有被选取的 input 元素
:checked　$(":checked")　所有被选中的 input 元素

四. jQuery 对象访问

1.each(callback) ： 迭代符合选择器的所有元素

```
$("li").click(function(){
        $("li").each(function(){
            $(this).removeAttr("class");
            /*$(this).removeClass("on");*/
            /*$(this).removeProp("class");*/
        });
        $(this).attr("class","on");
        /* $(this).addClass("on");*/
        /*$(this).prop("class","on");*/
});
```

2.length：符合选择器的元素的个数。

```
alert($("li").length);
```

3.selector：返回你用什么选择器来找到这个元素的。

```
alert($("li").selector);
```

4.get([index]) 返回的是所选中的元素，下标从 0 开始，返回的是 DOM 对象

```
alert($("li").get(1).innerText);
```

:eq(index) 返回的是所选中的元素，下标从 0 开始，返回的是 jQuery 对象

```
alert($("li:eq(1)").text());
```

5.index([selector|element]) 搜索匹配的元素,返回相应元素的索引值,从 0 开始计数。

```
var $l = $("li");
alert($l.index());
alert($l.index($(".l4")));
alert($(".l3").index("li"));
```

五. jQuery 筛选

1. 过滤：

1.1 eq(index|-index) 通过下标返回指定元素的 jQuery 对象，正数从前往后，负数倒过来，从 0 开始

```
$("p").eq(1).css({
    "background-color":"pink"
});
$("p").eq(-2).css({
    "background-color":"orange"
});
```

1.2 first() 获取第一个元素

```
$("p").first().css({
    "background-color":"pink"
});
```

1.3 last() 获取最后一个元素

```
$("p").last().css({"background-color":"orange"});
```

1.4 hasClass(class) 检查当前的元素是否含有某个特定的类，如果有，则返回 true。

```
$("p").each(function(){
    if($(this).hasClass("on")){
        $(this).css({"background-color":"orange"});
    }
});
```

1.5 filter(expr|obj|ele|fn) 筛选出与指定表达式匹配的元素集合。

```
/*$("p").filter(".on:first").css({
    "background-color":"orange"
});*/
/*var $p = $("p:first");
$("p").filter($p).css({
    "background-color":"orange"
});*/
/*var p = document.getElementsByTagName("p")[0];
$("p").filter(p).css({
    "background-color":"orange"
});*/
/*var ps = $("p").filter(function(){
    return $(this).hasClass("on");
});
for(var i=0;i<ps.length;i++){
    //将 DOM 对象加上$，可以变为 jQuery 对象
    $(ps[i]).css("background-color","orange");
}*/
```

1.6 is(expr|obj|ele|fn) 若有符合元素则返回 true

```
$("p,h1").click(function(){
    alert($(this).is("p"));
});
```

1.7 map(callback)

```
var str = $("p").map(function(){
   return $(this).text();
}).get().join(",");
$("h1").text(str);
```

1.8 has(expr|ele) 保留包含特定后代的元素，去掉那些不含有指定后代的元素。

```
$("li").has("ul").css({
    "background-color":"orange"
});
```

1.9 not(expr|ele|fn) 从匹配元素的集合中删除与指定表达式匹配的元素

```
$("p").not($(".on")).css({"background-color":"orange"});
$("p").filter($(".on")).css({"background-color":"orange"});
```

2.0 slice(start,[end]) 选取从 start 开始，取不到 end 的，从 0 开始，若为负数表示从后往前

```
$("p").slice(0,2).css({  //表示取到前两个 p 元素
    "background-color":"orange"
});
$("p").slice(-1).css({  //表示最后一个 p 元素
    "background-color":"orange"
});
```

2.查找：

2.1 children([expr])取得一个包含匹配的元素集合中每个元素所有子元素的元素集合。

```
$("ul").children(".l1").css({
    "background-color":"orange"
});
```

2.2 find(e|o|e) 搜索所有与指定表达式匹配的元素。

```
$("li").find("div").css({
    "width":"100px",
    "height":"100px",
    "background-color":"orange"
});
```

2.3 next([expr])取得一个包含匹配的元素集合中每一个元素紧邻的后面同辈元素的元素集合。

```
$("p").next(".on").css({
    "background-color":"orange"
});
```

2.4 nextAll([expr]) 查找当前元素之后所有的同辈元素。

```
$("p").nextAll().css({
    "background-color":"orange"
});
```

2.5 其余参考 API

六. jQuery 事件及事件对象

jQuery 事件

1.bind() 向匹配元素附加一个或更多事件处理器

语法:$(selector).bind({event:function(){}, event:function(){}, ...})

```
$("input").bind({
     mouseover:function(){$("body").css("background-color","lightblue");},
     mouseout:function(){$("body").css("background-color","orange");},
     click:function(){$(this).hide()}
 });
```

2.blur() 当元素失去焦点时发生 blur 事件。语法:$(selector).blur(function)

```
 $(":text").blur(function(){
       $("body").css({
             "background-color":"lightblue",
             "border":"2px solid blue"
       });
 });
```

3.当元素的值发生改变时会发生 change 事件。该事件仅适用于文本域（text field），以及 textarea 和 select 元素。语法:$(selector).change(function)

```
$("select").change(function(){
      $(this).css({
           "background-color":"lightblue"
      });
});
```

4.当点击元素时，会发生 click 事件。当鼠标指针停留在元素上方，然后按下并松开鼠标左键时，就会发生一次 click。语法: $(selector).click(function)

```
$(":button").click(function(){
      //让 p 段落元素向上滑动(隐藏)向下滑动(展开)
      $("p").slideToggle();
});
```

5.当双击元素时，会发生 dblclick 事件。语法:$(selector).dblclick(function)

```
$(":button").dblclick(function(){
      $("p").slideToggle();
});
$("p").click(function(){
      $(":button").dblclick();
});
```

6.delegate() 方法为指定的元素（属于被选元素的子元素）添加一个或多个事件处理程序，并规定当这些事件发生时运行的函数。使用 delegate() 方法的事件处理程序适用于当前或未来的元素（比如由脚本创建的新元素）。

语法:$(selector).delegate(childSelector,event,data,function)

```
var data = "abc";
$("div").delegate("p,select","click dblclick",data,function(event){
	//$(this).slideToggle();
	if(event.type=="click"){
		$("p").html("hello:"+data);
		$("<div style='width:100px;height:100px;background-color:
olivedrab'></div>").insertAfter(":button");
	}else if(event.type=="dblclick"){
		$("p").html("byebye:"+data);
		$("<div style='width:100px;height:100px;background-color:
cornflowerblue'></div>").insertBefore(":button");
	}
});

//给 div 子元素 p 添加点击事件
$("div").delegate('p','click',function(){
	$(this).slideToggle();
});
$(":button").click(function(){
	$("<p>这是一个新的段落标签</p>").insertAfter($(":button"));
});
```

7.error() 元素遇到错误（没有正确载入） 语法:$(selector).error(function)

```
$("img").error(function(){
	$("img").replaceWith("<p><b>图片未加载！</b></p>");
});
$("button").click(function(){
	$("img").error();
});
```

8.preventDefault() 方法阻止元素发生默认的行为（例如，当点击提交按钮时阻止对表单的提交）。语法:event.preventDefault()
isDefaultPrevented()方法返回指定的 event 对象上是否调用了 preventDefault()方法。
语法:event.isDefaultPrevented()

```
$(":submit").click(function(event){
	if($('p').text()=="哈哈"){
		event.preventDefault();
	}
	alert("Default prevented: " + event.isDefaultPrevented());
});
```

9.focus() 当元素获得焦点时 语法:$(selector).focus(function)

```
$(":text").focus(function(){
      $(":text").css({
           "background-color":"lightblue",
           "border":"3px solid olivedrab"
      });
});
```

10.load() 当指定的元素（及子元素）已加载时,该事件适用于任何带有 URL 的元素（比如图像、脚本、框架、内联框架）。语法:$(selector).load(function)

```
$("img").load(function(){
      $("p").text("哈哈哈");
});
```

11.one() 方法为被选元素附加一个或多个事件处理程序，当使用 one() 方法时，每个元素只能运行一次事件处理器函数。语法:$(selector).one(event,data,function)

```
$("p").one("click mouseover",function(){
        $(this).animate({fontSize:"+=6px"});
});
```

12.resize() 当调整浏览器窗口的大小时，发生 resize 事件.语法:$(selector).resize(function)

```
var x=0;
$(window).resize(function(){
        $(".p2").text(x+=1);
        $("img").css({
              "border":"3px solid blue"
         });
});
```

13.scroll()当用户滚动指定的元素时;scroll 事件适用于所有可滚动的元素和 window 对象（浏览器窗口）。语法:$(selector).scroll(function)

```
$(window).scroll(function(){
       $(".inner1").css({
             "background-color":"olive"
        });
       $(".inner2").css({
              "display":"block"
       });
});

$(window).scroll(function(){
    if($(window).scrollTop()>200){
        $(".inner2").css({
            "display":"block"
        });
    }
});
```

14.select()当 textarea 或文本类型的 input 元素中的文本被选择时，会发生 select 事件。语法:$(selector).select(function)

```
$("input").select(function(){
      alert("111");
});
```

15.submit()当提交表单时该事件只适用于表单元素.语法:$(selector).submit(function)

```
$("form").submit(function(){
       var name = $(":text").val();
       if(name==""){
              $("p").html("<h1 style='color:red'>用户名不能为空</h1>");
       }
});
```

16.trigger() 方法触发被选元素的指定事件类型

```
$("input").select(function(){
       $("p").after("111");
});
$(":button").click(function(){
       $("input").trigger("select");
});
$("form").submit(function(){
       var name = $(":text").val();
       if(name==""){
            $("p").html("<h1 style='color:red'>用户名不能为空</h1>");
       }
});
$(":button").click(function(){
       $("form").trigger("submit");
});
```

17.unload()当用户离开页面时 (对谷歌和火狐无效)

```
$(window).unload(function(){
       alert("Goodbye!");
});
```

jQuery 事件对象

1.pageX() 属性，是鼠标指针的位置，相对于文档的左边缘。
  pageY() 属性，是鼠标指针的位置，相对于文档的上边缘。

```
$(document).mousemove(function(event){
      $(".p2").text("X: " + event.pageX + ", Y: " + event.pageY);
});
```

2.result 属性，包含了当前事件最后触发的那个处理函数的返回值，除非值是 undefined。

```
$(":button").dblclick(function(e) {
       return ("最后一次点击的鼠标位置是： X" +e.pageX + ", Y" + e.pageY);
});
$(":button").dblclick(function(e) {
       $(".p2").html(e.result);
});
```

3.target 属性，规定哪个 DOM 元素触发了该事件

```
function handler(event) {
    var $target = $(event.target);
    if( $target.is("li") ) {
        $target.children().toggle();
    }
}
$("ul").click(handler).find("ul").hide();
```

4.timeStamp 属性，返回事件触发时距离 1970 年 1 月 1 日的毫秒数。

```
$(":button").click(function(e){
       $("p").text(e.timeStamp);
});
```

5.type 属性，描述触发哪种事件类型。语法:event.type

```
$("p").bind('click dblclick mouseover mouseout',function(event){
       $("p").html("Event: " + event.type);
});
```

6.which 属性，指示按了哪个键或按钮。语法:event.which

```
$(":text").keydown(function(event){
      $("p").html("Key: " + event.which);
});
```

七. jQuery 效果

1.隐藏和显示：使用 hide() 和 show() 方法来隐藏和显示 HTML 元素

语法： $(selector).hide(speed,callback); $(selector).show(speed,callback);

```
function change1(){
    $("body").css({
        "background-color":"orange"
    });
}


function change2(){
    $("body").css({
        "background-color":"lightblue"
    });
}


$(document).ready(function(){
    $(".b1").click(function(){
        $("p").show("fast",change1());
    });
    $(".b2").click(function(){
        $("p").hide("slow",change2());
    });
});
```

2.淡入淡出：

```
1.jQuery fadeIn()用于淡入已隐藏的元素,语法:$(selector).fadeIn(speed,callback);
$(".b1").click(function(){
    $(".d1").fadeIn();
    $(".d2").fadeIn("fast");
    $(".d3").fadeIn("slow");
    $(".d4").fadeIn(5000);
});


2.jQuery fadeOut()方法用于淡出可见元素,语法:$(selector).fadeOut(speed,callback);
$(".b2").click(function(){
    $(".d1").fadeOut();
    $(".d2").fadeOut("fast");
    $(".d3").fadeOut("slow");
    $(".d4").fadeOut(5000);
});
```

3. jQuery fadeToggle()方法可以在 fadeIn()与 fadeOut()方法之间进行切换。如果元素已淡出，则 fadeToggle()会向元素添加淡入效果。如果元素已淡入，则 fadeToggle()会向元素添加淡出效果。语法： $(selector).fadeToggle(speed,callback);

```
$(".b3").click(function(){
    $(".d1").fadeToggle();
    $(".d2").fadeToggle("fast");
    $(".d3").fadeToggle("slow");
    $(".d4").fadeToggle(5000);
});
```

4. jQuery fadeTo() 方法允许渐变为给定的不透明度（值介于 0 与 1 之间）。
语法： $(selector).fadeTo(speed,opacity,callback);

```
$(".b4").click(function(){
    $(".d1").fadeTo("slow",0.1);
    $(".d2").fadeTo("slow",0.7);
    $(".d3").fadeTo("fast",0.1);
    $(".d4").fadeTo("fast",0.7);
});
```

3.滑动:

1. jQuery slideDown()方法用于向下滑动元素。语法：

```
$(selector).slideDown(speed,callback);
$(".b1").click(function(){
    $(".d1").slideDown("slow");
    $(".d2").slideDown("fast");
    $(".d3").slideDown(3000);
});
```

2. jQuery slideUp() 方法用于向上滑动元素。语法：

```
$(selector).slideUp(speed,callback);
$(".b2").click(function(){
    $(".d1").slideUp("slow");
    $(".d2").slideUp("fast");
    $(".d3").slideUp(3000);
});
```

3. jQuery slideToggle() 方法可以在 slideDown() 与 slideUp() 方法之间进行切换。如果元素向下滑动，则 slideToggle() 可向上滑动它们。如果元素向上滑动，则 slideToggle() 可向下滑动它们。语法:$(selector).slideToggle(speed,callback);

```
function change(){
    $("body").css({
        "background-color":"lightblue"
    });
}
$(".b3").click(function(){
    $(".d1").slideToggle("slow",change());
    $(".d2").slideToggle("fast");
    $(".d3").slideToggle(3000);
});
```

4.动画:

```
jQuery animate() 方法用于创建自定义动画。 语法:
$(selector).animate({params},speed,callback);
$(document).ready(function(){
    $(":button").click(function(){
        $("div").animate({
            width:"200px",height:"200px",left:"150px",opacity:"0.8"
        },"slow");
        $("span").animate({
            fontSize:"3em",left:"30px",top:"50px"
        },"slow");
    });
});
```

5.停止动画:

```
jQuery stop() 方法用于在动画或效果完成前对它们进行停止。stop() 方法适用于所有
jQuery 效果函数，包括滑动、淡入淡出和自定义动画。
语法:$(selector).stop(stopAll,goToEnd);
$(document).ready(function(){
    $("#d1").click(function(){
        $("#d2").slideDown(5000);
    });
    $(":button").click(function(){
        $("#d2").stop();
    });
});
```

八.jQuery Callback 函数

```
/* 当动画 100% 完成后，即调用 Callback 函数。 */
$(document).ready(function(){
    $("#d1").click(function(){
        /* 保证 div 块全部展开之后,再弹出对话框 */
        $("#d2").slideDown(3000,function(){
        alert(111);
        });


   /* jQuery - Chaining 允许在相同元素上,通过 jQuery 把动作/方法链接起来。*/
        $("#d2").css({"background-
color":"orange","width":"100px"}).slideDown(2000).slideUp(2000);
```

```
    });
});
```

九.validation 校验插件

一款优秀的表单验证插件——validation 插件

特点：

内置验证规则：拥有必填、数字、email、url 和信用卡号码等 19 类内置验证规则

自定义验证规则：可以很方便的自定义验证规则

简单强大的验证信息提示：默认了验证信息提示，并提供自定义覆盖默认提示信息的功能

实时验证：可以通过 keyup 或 bulr 事件触发验证，而不仅仅在表单提交的时候验证。

9.1 插件下载

官网地址：http://jqueryvalidation.org

帮助文档位置：http://jqueryvalidation.org/documentation

目录结构：

9.2 导入

validate 是 jQuery 的插件，即必须在 jQuery 的基础上运行。我们需要导入 jQuery 库、validate 库、国际化资源库（可选）。

9.3 使用前提

validate 需要手动的声明，对哪个表单进行校验默认校验规则如下：

序号	规则	描述
1	required:true	必须输入的字段。
2	remote:"check.php"	使用 ajax 方法调用 check.php 验证输入值。
3	email:true	必须输入正确格式的电子邮件。
4	url:true	必须输入正确格式的网址。
5	date:true	必须输入正确格式的日期。日期校验 ie6 出错，慎用。
6	dateISO:true	必须输入正确格式的日期（ISO），例如：2009-06-23，1998/01/22。只验证格式，不验证有效性。
7	number:true	必须输入合法的数字（负数，小数）。
8	digits:true	必须输入整数。
9	creditcard:	必须输入合法的信用卡号。
10	equalTo:"#field"	输入值必须和 #field 相同。

11	accept:	输入拥有合法后缀名的字符串（上传文件的后缀）。
12	maxlength:5	输入长度最多是 5 的字符串（汉字算一个字符）。
13	minlength:10	输入长度最小是 10 的字符串（汉字算一个字符）。
14	rangelength:[5,10]	输入长度必须介于 5 和 10 之间的字符串（汉字算一个字符）。
15	range:[5,10]	输入值必须介于 5 和 10 之间。
16	max:5	输入值不能大于 5。
17	min:10	输入值不能小于 10。

9.4 检验方式

校验基本语法：

```
$(ele).validate({
        rules:{},
        messages:{}
});
```

其中：

rules 规则语法：

```
        rules:{
                字段名:校验器,
                字段名:校验器,
                ……
        }
```

校验器语法：{校验器:值,校验器:值,….}

message 提示语法：

```
messages:{
        字段名:{校验器:”提示”， 校验器:”提示”}
}
```

9.5 案例

实现登录页面的姓名和密码校验：

1).用户名不能为空

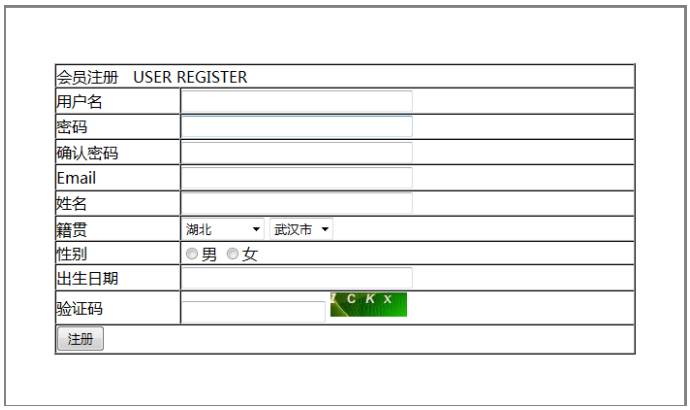
2).密码不能为空,密码只能是数字,密码不得少于 6 位

作业：

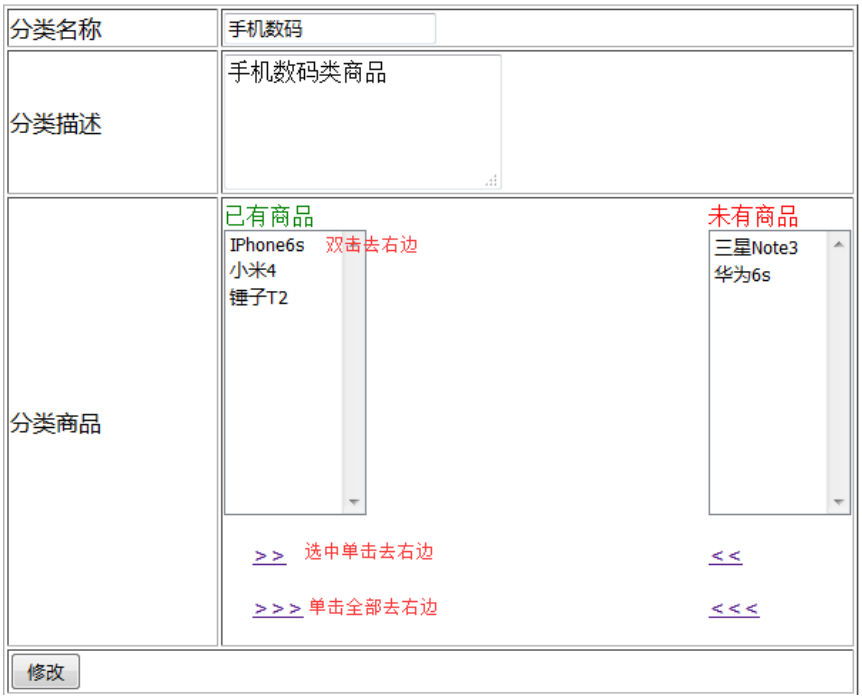
1. tab 切换

2. 购物车 checkbox 选择

3. 省市二级联动



4.下拉列表左右选择



5.validation 完成注册页面表单校验

要求：

1).用户名不能为空，用户名不得少于 3 位

2).密码不能为空,密码只能为数字,密码长度不能少于 6 位

3).确认密码不能为空,两次密码输入不一致

4).邮箱不能为空,请输入有效的邮箱地址

5).姓名不能为空,姓名长度不超过 5 位

6).必须选择性别