返回目录

第七章　表单

本章导读

HTML提供了丰富的数据输入标记，主要以input标记为主，textarea和select也其是重要组成部分。虽然JavaScript提供了prompt和confirm两个函数用于输入数据，无论从美观、亦或可输入数据类型等方面都不能满足需求。HTML标记提供了文本框(单行、多行)、单选按钮、复选框、列表框、密码框、文件上传框、按钮等多种形式的输入方式，基本能满足用户对输入的需求。HTML的输入标记同样适用CSS、jQuery和JavaScript等，更是增强了HTML输入标记的能力。

表单要素是重要的输入方式，是实现页面向Web服务器提交数据的重要手段，Web服务器获取页面输入信息后，可以保存信息，也可根据输入信息的不同，实现不同页面的转向。

本章将不涉及服务器端编程，因此本章内容仅在同一个页面内实现数据处理和交互。

学习目标：

1. 认识Form；
2. 了解HTML输入元素；
3. 了解Form元素相关事件；
4. 掌握Form元素常用案例；

本章目录

HTML提供了丰富的数据输入标记，主要以input标记为主，textarea和select也其是重要组成部分。虽然JavaScript提供了prompt和confirm两个函数用于输入数据，无论从美观、亦或可输入数据类型等方面都不能满足需求。HTML标记提供了文本框(单行、多行)、单选按钮、复选框、列表框、密码框、文件上传框、按钮等多种形式的输入方式，基本能满足用户对输入的需求。HTML的输入标记同样适用CSS、jQuery和JavaScript等，更是增强了HTML输入标记的能力。

第一节　认识HTML输入标记

通过改变input标记的type属性，可以表现为单行文本框(type=text)，单选按钮(type=radio)、复选框(type=checkbox)、 密码框(type=password)、文本上传框(type=file)、隐藏(type=hidden)、按钮(type=button)、图片提交(type=image)、 提交按钮(type=submit)、

图7-1 HTML输入标记示意图

表1：HTML输入标记及其说明

标记格式	使用说明
<input type=text> 单行文本输入框	该标记形式如[ ]， 在文本框中可以输入单行文本。 ◆value属性：通过设置value属性设置默认值，即页面启动时就显示的文本内容； ◆maxlength属性：设定可输入文本的最大长度，汉字按一个字符计算； ◆readonly属性：设定文本框只能读取，不能修改；
<input type=password> 密码输入框	该标记形式如[••••••]， 当在密码输入框输入内容时，为了安全显示为特定符号如圆点。密码输入框不接受汉字作为密码。 ◆value属性：通过设置value属性设置默认值，即页面启动时就显示的文本内容； ◆maxlength属性：设定可输入文本的最大长度，汉字按一个字符计算； ◆readonly属性：设定文本框只能读取，不能修改；
<input type=radio> 单选按钮	该标记形式如◉ ○ 一组单选按钮每次仅能选中其中一个，非此即彼，通过设置属性Name分组。 ◆name属性：如果页面中有多组Radio，可通过设置不同的Name属性分组，比如：职称选择组的Name为"Title"，性别选择组的Name为"Sex" ◆checked属性：当在单选钮标记设置该属性时，表示默认该按钮被选中，中间有一个黑色圆点，该属性无需设置属性值； ◆value属性：该属性用于单选钮标记时，用于设置取值，与显示没有关系，与单选钮对应的文本用其他HTML标记设定；
<input type=checkbox> 复选框	该标记形式如☑ ☐ 一组复选框钮可以选择多个，如设定某人的爱好，就可能有多种爱好，同type=radio一样，可通过设置属性Name分组。 ◆name属性：如果页面中有多组checkbox，可通过设置不同的Name属性分组，比如：爱好组的Name为"Taste"，运动组的Name为"Sport"等； ◆checked属性：当在复选框标记设置该属性时，表示默认选中，中间有一个小勾(√)，该属性无需设置属性值； ◆value属性：该属性用于设置每个复选框的取值，与显示没有关系，与其对应文本用其他HTML标记设定；
	该标记形式如[I am Button]

<input type=button> 命令按钮	命令按钮通常用于启动JavaScript程序。HTML还提供了Button标记， 其格式为<Button>I am Button</Button>效果如 I am Button ， 其功能与之相似。
<input type=file> 文件名输入框	用于向上传文件，将会有文本框和自动匹配的浏览文件的按钮，如图中“照片”对应的条目。 当选择“浏览”按钮时，系统将自动弹出常见的文件选择对话框以便选择文件等。 选择该标记，并不能自动完成文件上传，还需要服务器端配合。
<input type=hidden> 隐藏输入	该类型输入标记将不在页面显示，程序员可把需要且不必用户填写的内容放在该标记中， 和type=text一样，其值可以通过value属性指定。
<input type=reset> 复位按钮	如图“重新设定”所示，该标记显示为一个命令按钮，当其在form(表单)内时， 单击该按钮将清空已填写内容恢复到初进页面的状态。
<input type=submit> 提交按钮	如图“提交”按钮所示，该标记显示为一个命令按钮，当其在form(表单)内时，将自动连接到form内指定的链接， 并把页面输入信息传递到该链接页面以便处理。
<input type=image> 图像提交按钮	图像提交按钮的功能与type=submit的功能一样，都是向服务器提交。由于图片可多样化， 因此能个性化显示效果。图片来源和Img标记一样，都是通过src属性设置。
<TextArea></TextArea> 多行文本框	TextArea标记用于多行文本输入，而<input type=text>只能用于单行文本输入。 TextArea是一个独立标记，不能通过input标记设置type属性而成，这是该标记与前面诸标记的不同。 默认显示的内容可以直接放在一对TextArea之间。
<Select></Select> 列表框	列表框和TextArea一样是一个独立的HTML标记，不能通过input标记设置type属性而成。 列表框可以在有限的空间内显示多个条目，当有较多的条目时，通常采用列表框。 列表框内的条目通过option对设置，其格式为 <option value="属性值">显示内容</option> 每个条目对应一对option。option的属性值与显示内容不一定一样，可以是对照关系。
<label></label> 标签	该元素为input元素定义标注，当鼠标单击其内容时，被标注的元素自动获得焦点，或选中。label元素有for属性，应与被标注元素的id属性值相同。参见例程7-4。

HTML输入标记都可以设置disabled(禁用)属性，当有该属性时，被禁用的 input 元素既不可用，也不可点击。如果需要改变该属性的值，可以通过JavaScript或jQuery予以改变。

HTML输入标记还可以设置Name属性，这与数据传递到服务器有关，请参看第二节的“Form方式”。

例程7-1

```
第1行  <HTML>
第2行      <HEAD>
第3行          <TITLE>页面输入表单(form)</TITLE>
第4行              <Style type=text/css>
第5行                  *{font-size:16px;}
第6行                  #InputBox{text-align:Center;font-size:30px;font-weight:bold;margin-bottom:20px}
第7行                  #DataInput{width:400px;border:1px dotted gray;text-align:Left;padding:20px 10px 0px 10px}
第8行                  span{font-size:12px}
第9行                  input,textarea{font-weight:bold}
第10行             </Style>
第11行         </HEAD>
第12行         <BODY>
第13行             <Center><Div ID="DataInput">
第14行                 <Form>
第15行                     <Div id="InputBox">信息录入</Div>
第16行                         姓名
第17行                         <input type=text style="width:200px;height:24px;"
```

```
name="Name"><BR><BR>
第18行                                出生
第19行                                    <Select name="BirthDay">
第20行                                        <option value="LT60">1960以前</option>
第21行                                        <option value="1960">1960年代</option>
第22行                                        <option value="1970">1970年代</option>
第23行                                        <option value="1980">1980年代</option>
第24行                                        <option value="1990">1990年代</option>
第25行                                        <option value="2000">2000年代</option>
第26行                                        <option value="2010">2010年代</option>
第27行                                    </Select><BR><BR>
第28行                                账号
第29行                                    <input type=text style="width:100px;height:24px;"
name="LogName">
第30行                                    <Span>(设定登录本网站账号，不超过8个字符)</Span><
BR><BR>
第31行                                密码
第32行                                    <input type=password style="width:100px;height:24
px;" name="PassWord">
第33行                                    <Span>(不超过8个字符)</Span><BR><BR>
第34行                                性别
第35行                                    <input type=radio checked name="Sex" valu
e="1">男
第36行                                    <input type=radio name="Sex" value="0">女
<BR><BR>
第37行                                爱好
第38行                                    <input type=checkbox name="Taste">运动
第39行                                    <input type=checkbox name="Taste">阅读
第40行                                    <input type=checkbox name="Taste">购物
第41行                                    <input type=checkbox name="Taste">上网 <B
R><BR>
第42行                                <Div>自我简介</Div>
第43行                                    <TextArea style="width:100%;height:100px"
Name="AboutMe"></TextArea><BR><BR>
第44行                                照片
第45行                                    <Input type=file><BR><BR>
第46行                                <input type="reset" value="重新设定">
第47行                                <input type=submit value="提交">
第48行                            </Form>
第49行                        </Div></Center>
第50行                    </BODY>
第51行                </HTML>
```

第二节　与服务器交互

HTML的各种输入标记，可以在页面内完成数据输入，但更多是向服务器传递数据。目前，在WEB页面上向服务器传递信息的渠道主要有以下四种，一是通过带参数的链接，二是通过form表单，三是通过AJAX，四是通过一些下载到本地的各种插件，如：Adobe Flash、Microsoft Silverlight等。

向服务器端提交数据，大多涉及服务器端的编程。服务器端采用的运行平台、开发语言有很多，如：Java系列，PHP、ASP(ASP .net)等都是较为知名开发和运行平台。由于侧重点不同，本章将简单介绍通过带参数的链接和form表单向服务器传递数据，AJAX方式将有专门的章节予以讨论，插件模式将不涉及。

1、链接参数方式

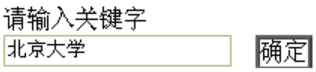

图7-2 链接参数调整

不少链接形如“http://www.baidu.com/s?wd=北京大学”，其中?后面的内容是链接参数，每个参数由参数名称(如wd)和参数值(北京大学)构成，参数和参数之间用&分开。当链接所指向的服务器受到链接参数后，将有程序接收该参数，并根据参数不同反馈与其对应内容。“http://www.baidu.com/s?wd=北京大学”是百度(BaiDu)的搜索链接，只要改变参数wd所等于的内容，即可完成不同的关键字的搜索。例程7-2的执行效果如图7-2所示。采用链接参数方式向服务器传递信息，比较常用，但缺点是在网址栏能看见传递的信息，如果对该信息有安全要求，就未必妥当。

例程7-2

```
第1行   <HTML>
第2行       <HEAD>
第3行           <TITLE>向服务器传递数据-链接参数</TITLE>
第4行           <SCRIPT LANGUAGE="JavaScript" src="./jquery-1.4.2.min.js"></SCRIPT>
第5行           <Style type=text/css>
第6行               a{text-decoration:none;color:black;border:3px ridge gray;}
第7行           </Style>
第8行       </HEAD>
第9行       <BODY>
第10行          <Center>
第11行              <Div style="text-align:Left;width:250px;">
第12行                  请输入关键字<BR>
第13行                  <input type=text ID=KeyWord><A href="#" ID="Go2Baidu">确定</A>
第14行              </Div>
第15行          </Center>
第16行          <SCRIPT LANGUAGE="JavaScript">
第17行              $("#KeyWord").focus();//当页面启动时，关键字文本框获得焦点
第18行              $("#KeyWord").blur(function(){//当关键字文本框失去焦点时执行
第19行                  var Input_Content=$(this).attr("value");//取得关键字内容
第20行                  if(Input_Content.length==0)return false;//如果关键字文本框为空，返回
第21行                  //修改用A模拟的命令按钮，修改其href属性值
第22行                  $("#Go2Baidu").attr("href","http://www.baidu.com/s?wd="+Input_Content);
第23行              });
第24行          </SCRIPT>
第25行      </BODY>
第26行  </HTML>
```

2、Form方式

Form是一个HTML标记，其格式如下:

```
<form action="接收数据的链接地址" method="数据提交方式" target="打开链接的位置"></form>
```

在Form标记对之间，可以放入各种类型的HTML标记，其中包括各种类型Input标记、TextArea和Select标记等。 在Form标记中，action属性值为接收数据的地址；method属性值为页面向服务器传递数据的方式，可分别设为get和post； target的含义与A(链接)中的Target属性含义一致，为显示打开链接的位置，默认为_self(在当前窗口打开)、 如果设为_blank将在一个新窗口打开。

为了能将Form中各种HTML输入标记(如Input标记)的值传递给服务器，要求每个输入标记都必须包含Name属性， 相当于链接参数的名称。当Form提交时，将自动将这些标记的值传递给服务器。

Form的method属性取值不同，向服务器传递数据的方式也不同。采用Get方式时，将把HTML输入标记的值作为链接参数传递给服务器，且将链接显示在地址栏，如例程7-3所示的第10行代码。如果采用Post方式，则不在网址栏通过链接参数显示(链接与action属性的设定值保持一致)，安全性更好，且传递的数据总量也比Get方式大。另外，不同的传递方式对服务器端的开发也有所不同。

例程7-3和例程7-2的显示效果相同，但其代码差别较大。例程7-2相当于用JavaScript和jQuery一起构造链接，而例程7-3则是通过Form方式让系统自动完成。例程7-3还有一些缺陷，比如：当“关键字”文本框没有输入内容时，也可以单击“确定”按钮，如果要求必须输入内容才能单击确定，可以通过JavaScript程序或者jQuery代码予以处理。

例程7-3

```
第1行  <HTML>
第2行      <HEAD>
第3行          <TITLE>向服务器传递数据-form方式</TITLE>
第4行      </HEAD>
第5行      <BODY>
第6行          <Center>
第7行              <form action="http://www.baidu.com/s" method="get">
第8行                  <Div style="text-align:Left;width:250px;">
第9行                      请输入关键字<BR>
第10行                     <input type=text name="WD">
第11行                     <input type=submit value="确定">
第12行                 </Div>
第13行             </form>
第14行         </Center>
第15行     </BODY>
第16行 </HTML>
```

第三节　与jQuery相关

jQuery能很好地操控HTML输入标记和Form，根据标记特点，将按选中标记、value属性处理以及相关事件三个部分予以阐述。

1、选中元素

要操控HTML标记，首先是选中该标记。对于HTML的输入标记，同样可以通过ID、Class以及标记名称选中。 不过jQuery还提供了专门针对HTML输入标记的方法。如表9-2所示。

表2：与HTML输入标记相关的jQuery选择器

jQuery选择器	使用说明
$(":input") 匹配输入标记	匹配所有 input, textarea, select 和 button(包括type=button和button标记)元素。注意， 如果省略input前面的冒号写成$("input") 将仅仅只选择input系列的元素，不包含textarea、select和button。
$(":text") 匹配单行文本框	匹配所有单行文本框，其效果与$("input:text") 等同。
$(":password") 匹配密码输入框	匹配所有密码输入框，其效果与$("input:password") 等同。
$(":radio") 匹配单元组	匹配所有单选钮，其效果与$("input:radio") 等同。
$(":checkbox") 匹配复选框	匹配所有复选框，其效果与$("input:checkbox") 等同。

$(":submit") 匹配提交按钮	匹配所有提交按钮，其效果与$("input:submit")等同。
$(":image") 匹配图像提交	匹配所有图像提交，其效果与$("input:image")等同。 注意该命令不能匹配Img标记。
$(":reset") 匹配重置按钮	匹配所有重置提交，其效果与$("input:reset")等同。
$(":button") 匹配命令按钮	匹配命令按钮，指type=button和button标记，$("input:button") 仅能匹配<input type=button>，不能匹配button标记。
$(":file") 匹配文件输入	匹配所有type=file的标记，其效果与$("input:file")等同。
$(":hidden") 匹配隐藏标记	匹配所有type=hidden的input标记和其他隐藏标记(即display:none的标记)。 如果要匹配可见标记用$(":visible")即可。
$(":disabled") 匹配不可用标记	匹配所有不可用标记。
$(":enabled") 匹配可用标记	匹配所有可用标记。
$(":checked") 匹配被选中标记	匹配所有复选框、单选框中被选中的标记，不包括select标记被选中的option。
$(":selected") 匹配option标记	匹配所有select标记下的所有被选中的option元素。

2、jQuery读写value属性值

设置HTML输入标记的value属性，可以通过$.attr("value")取得属性值，或者$.attr("value","新属性值")设置value属性值。 也可以通过$.removeAttr("value")方法移去value属性。

jQuery针对HTML输入标记提供了更加直接方法val()读取或者设置这些标记的value属性值，其方法如下：

◆$.val()：读取jQuery匹配元素集中的第一个元素的当前值；

◆$.val(value)：设置每一个匹配元素的value属性值；

◆$.value(array)：type=checkbox的input,type=radio的input标记，以及select标记等都能使用为之赋值；

◆$.value(fn)：fn代表函数，有两个参数，分别是元素的index值和value值，通过该函数， 可以根据元素的index和value分别设定value属性值。

3、用jQuery处理HTML输入标记的事件

一般网页元素主要是对鼠标事件如click、mouseover、mouseout等做出反应，但对输入标记， 还经常处理键盘事件(keypress、keyup、keydown)，焦点事件(blur、focus、focusin、focusout)， 选择事件(select)、提交事件(submit)、改变事件(change)等。表9-3是对相关事件的使用说明。

表3：HTML输入标记相关事件

事件类型	事件说明
blur() 失去焦点	格式1：$.blur() 触发jQuery选中标记的blur事件。 格式2：$.blur(fn) 当jQuery选中标记发生blur事件时，执行fn所代表的函数。
focus() 获得焦点	格式1：$.focus() 触发jQuery选中标记的focus事件。 格式2：$.focus(fn) 当jQuery选中标记发生focus事件时，执行fn所代表的函数。 注意:某些元素不支持focus事件，不同浏览器不同版本对该事件的支持也不尽相同， 可以用focusin事件予以解决。在有输入标记的页面，可以将光标默认放到第一个文本框，便于用户输入。 几乎所有的搜索引擎启动时，都将光标置于搜索框大都采用该事件处理。
	格式1：$.focusin() 触发jQuery选中标记的focusin事件。

focusin() 获得焦点	格式2：$.focusin(fn) 当jQuery选中标记及其子标记发生focusin(获得焦点)事件时，执行fn所代表的函数。 注意:内部子元素获得焦点时，其父元素同样会触发该事件。 focus事件只能检查元素自身获得焦点的事件。
focusout() 失去焦点	格式1：$.focusout() 触发jQuery选中标记的focusout(失去焦点)事件。 格式2：$.focusout(fn) 当jQuery选中标记及其子标记发生focusout(失去焦点)事件时，执行fn所代表的函数。 注意:内部子元素失去焦点时，其父元素同样会触发该事件。 blur事件只能检查元素自身失去焦点事件。
select() 选择事件	格式1：$.select() 触发jQuery选中标记的select(选择)事件。 格式2：$.select(fn) 当jQuery选中标记发生select(选择)事件时，执行fn所代表的函数。 当用户在input文本框和TextArea选中某段文字时，会触发该事件。
submit() 提交事件	格式1：$.submit() 触发jQuery选中标记的submit(提交)事件。 格式2：$.submit(fn) 当jQuery选中标记发生submit(选择)事件时，执行fn所代表的函数。 当用户单击提交按钮或者提交图片时，将会触发submit事件，也可以用程序触发该事件。 事件触发时，执行的程序fn定义。在页面上，单击提交功能时，通常会检验输入内容是否符合规定， $.submit(fn)，有关检验的程序，就可以写在fn所代表的程序中。
keypress() 按键事件	格式1：$.keypress() 触发jQuery选中标记的keypress(按键)事件，当在键盘上按键时，将会触发该事件。 格式2：$.keypress(fn) 当jQuery选中标记发生keypress(按键)事件时，执行fn所代表的函数。 当按下一个键时，首先触发keydown、然后是keypress、最后是keyup。 keypress捕获单个字符，主要捕获数字(包括Shift+数字的符号)、字母(包括大小写)、小键盘等除了F1-12、 SHIFT、Alt、Ctrl、Insert、Home、PgUp、Delete、End、PgDn、ScrollLock、Pause、NumLock、 {菜单键}、{开始键}和方向键外的标准字符。可以捕获的单个字符区分大小写， 数字键不区分是来自小键盘还是主键盘。
keydown() 按下键事件	格式1：$.keydown() 触发jQuery选中标记的keydown(按下键)事件，当在键盘上按下键时，将会触发该事件。 格式2：$.keydown(fn) 当jQuery选中标记发生keydown(按下键)事件时，执行fn所代表的函数。当在键盘按下键时， 首先触发该事件。KeyDown可捕获标准键盘上除PrScrn外的所有按键，包括组合键。 每个键位的按键值keyValue相同，也就是不区分大小写，小键盘的数字与主键盘数字的keyValue也不相同。
keyup() 松开键事件	格式1：$.keyup() 触发jQuery选中标记的keyup(松开键)事件，当在键盘上按下键后松开时，将会触发该事件。 格式2：$.keyup(fn) 当jQuery选中标记发生keyup(松开键)事件时，执行fn所代表的函数。Keyup可捕获标准键盘上除PrScrn外的所有按键，包括组合键。 每个键位的按键值keyValue相同，也就是不区分大小写，小键盘的数字与主键盘数字的keyValue也不相同。按下键后，keyup事件不一定触发，当按下键时，移动鼠标，将可能不会触发keyup事件。
change() 改变事件	格式1：$.change() 触发jQuery选中标记的change(改变)事件，当文本框中的内容发生改变时，触发该事件。 格式2：$.change(fn) 当jQuery选中标记发生change(改变)事件时，执行fn所代表的函数。 change事件常用于合法性判断，如输入邮政编码的文本框，当内容发生改变时，检查输入的内容是否都是数字等。

第四节　单选钮

单选钮是一组互斥且仅能单选的按钮，如性别仅有男女，且只能选中其中一个，职称有多种，但每个人在特定时间仅能有一种职称。同一组单选钮其Name属性相同，不同组单选钮其Name属性不同。不同组之间的选择，不影响另外一组的选择，同组内的选择，当单击时自动选中当前，取消其他选择。如例程7-4，图7-3所示。

例程7-4

```
第1行  <Html>
```

```
第2行        <Head>
第3行            <Title>单选钮</Title>
第4行            <Script language="JavaScript" src="jQuery-1.9.1.min.js"></Script>
第5行        </Head>
第6行        <Body>
第7行            性别：<BR>
第8行            <input type="radio" name="Sex" value="1" id="male"><label for="male">男</label>
第9行            <input type="radio" name="Sex" value="0" id="female" checked><label for="female">女</label>
第10行           <BR><BR>
第11行           职称：<BR>
第12行           <input type="radio" name="Title" value="0">助教
第13行           <input type="radio" name="Title" value="1">讲师
第14行           <input type="radio" name="Title" value="2" checked>副教授
第15行           <input type="radio" name="Title" value="3" >教授
第16行           <button>测试</button>
第17行           <Script language="JavaScript">
第18行               $("button").click(function(){
第19行                   alert($(":radio:checked[name='Sex']").val());//根据性别选择
第20行                   alert($(":radio:checked[name='Title']").val());//根据职称选择
第21行               });
第22行           </Script>
第23行       </Body>
第24行   </Html>
```

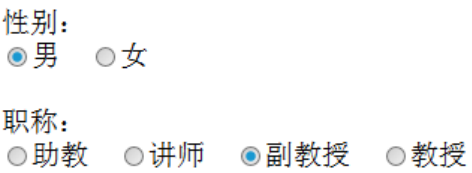

图7-3　单选钮示意图

通过jQuery能方便地获取每组单选钮的值，如例程7-4第19、20行所示，如$(":radio:checked[name='Sex']").val()的含义表示input类型为radio且name为Sex被选中按钮的值。如果某组单选钮没有一个被选中，则其值为undefined。

有时需要通过代码设定某个单选钮的状态，可参考如例程7-5所示，该代码用于替换例程7-4中的JavaScript代码。

例程7-5

```
第1行    <Script language="JavaScript">
第2行        //以下代码独立执行观察效果为宜
第3行        $("button").click(function(){
第4行            $(":radio[name='Sex'][value='1']").attr("checked",true);//表示Sex单选钮组之value为1的元素被选中
第5行            $(":radio:checked[name='Sex']").attr("checked",false);//表示设置已选中Sex单选钮为不选中状态
第6行            $(":radio:not(:checked)[name='Sex']").attr("checked",true);//表示设置未选中Sex单选钮为选中状态
第7行            //如有多个为被选中，也仅有一个被设置为选中，且为最后一个。
第8行        });
```

```
第9行    </Script>
```

另外，在撰写HTML代码时，如果某个单选钮含有checked属性，则该单选钮将被默认选中，如例程7-4第9、14行所示。

第五节　复选框

复选框是一组可以多重选择的元素，与单选钮不同，复选框之间没有互斥关系，如爱好，一个人可以有多个爱好，如图7-4，其实现代码如例程7-6所示。

图7-4　复选框示意图

例程7-6

```
第1行   <Html>
第2行       <Head>
第3行           <Title>单选钮</Title>
第4行           <Script language="JavaScript" src="jQuery-1.9.1.min.js"></Script>
第5行       </Head>
第6行       <Body>
第7行           爱好:<BR>
第8行           <input type="checkbox" name="Hobby" value="1">篮球
第9行           <input type="checkbox" name="Hobby" value="2">足球
第10行          <input type="checkbox" name="Hobby" value="4">唱歌
第11行          <input type="checkbox" name="Hobby" value="8">逛街
第12行          <input type="checkbox" name="Hobby" value="16">网购
第13行          <input type="checkbox" name="Hobby" value="32">阅读
第14行          <BR><BR>
第15行          <button id="selectAll">全选</button>
第16行          <button id="selectAllNot">全不选</button>
第17行          <button id="selectInverse">反选</button>
第18行          <button id="selectTest">测试</button>
第19行          <Script language="JavaScript">
第20行              $("#selectAll").click(function(){
第21行                  $(":checkbox[name='Hobby']").attr("checked",true);//全选
第22行              });
第23行              $("#selectAllNot").click(function(){
第24行                  $(":checkbox[name='Hobby']").attr("checked",false);//全不选
第25行              });
第26行              $("#selectInverse").click(function(){
第27行                  //被选中的加上state=Yes属性
第28行                  $(":checkbox:checked[name='Hobby']").attr("state","Yes");
第29行                  $(":checkbox:not(:checked)[name='Hobby']").attr("checked",true);//未选中
的变为选中
第30行
第31行                  $(":checkbox[name='Hobby'][state='Yes']").attr("checked",false).removeAtt
r("state");
```

```
第32行                    });
第33行                    $("#selectTest").click(function(){
第34行                        var selectedResult=0;
第35行                        $(":checkbox:checked[name='Hobby']").each(function(){
第36行                            selectedResult+=parseInt($(this).val());
第37行                        });
第38行                        alert(selectedResult);
第39行                    });
第40行            </Script>
第41行        </Body>
第42行    </Html>
```

第六节　列表框

列表框是网页中常用的元素，以有限的空间能呈现更多信息，如图7-5。在实际应用中常有联动需求，即根据上一级信息，下一级信息随之发生变化，如例程7-7所示，当选择其中某个城市或者省区时，右侧列表框内容随之发生变化。

图7-5　联动列表框示意图

例程7-7

```
第1行    <Html>
第2行        <Head>
第3行            <Title>列表框</Title>
第4行            <Script language="JavaScript" src="jQuery-1.9.1.min.js"></Script>
第5行            <Style type="text/css">
第6行                .List{display:none}
第7行            </Style>
第8行        </Head>
第9行        <Body>
第10行            联动列表框:<BR>
第11行            <select id="Level0">
第12行                <option value="Blank">----请选择省份----</option>
第13行                <option value="BJ">北京</option><option value="SH">上海</option><option value="JS">江苏</option>
第14行            </select>
第15行            <select ID="Level1" disabled>
第16行                <option>----请选择城市----</option>
第17行            </select><BR><BR><BR><BR>
第18行            <button id="btnTest">测试</button>
第19行            <select class="List" id="BJ">
第20行                <option value="BJ00">东城</option><option value="BJ01">西城</option>
第21行                <option value="BJ02">崇文</option><option value="BJ03">宣武</option>
```

```
第22行                    <option value="BJ04">朝阳</option>
第23行                </select>
第24行                <select class="List" id="SH">
第25行                    <option value="SH00">黄浦</option><option value="SH01">卢湾</option>
第26行                    <option value="SH02">徐汇</option><option value="SH03">长宁</option>
第27行                    <option value="SH04">静安</option>
第28行                </select>
第29行                <select class="List" id="JS">
第30行                    <option value="JS00">南京</option><option value="JS01">镇江</option>
第31行                    <option value="JS02">苏州</option><option value="JS03">南通</option>
第32行                    <option value="JS04">扬州</option>
第33行                </select>
第34行                <Script language="JavaScript">
第35行                    $("#Level0").change(function(){//当改变发生时
第36行                        var selectedValue=$(this).val();
第37行
第38行                        var Level1=$("#Level1");//选中Level1
第39行
第40行                        if(selectedValue=="Blank"){
第41行                            Level1.attr("disabled",true).children("option").slice(1).remove
();
第42行                        }else{
第43行                            Level1.children("option").slice(1).remove();//移走除第1个外的所有
选项
第44行                            var relatedOption=$("#"+selectedValue).children();//找到与选择相
关的选项
第45行                            Level1.attr("disabled",false).append(relatedOption);//将相关选项
添加到Level1
第46行                            Level1.children().eq(0).attr("selected",true);//将第一个选项选中
第47行                        }
第48行                    });
第49行                    $("#btnTest").click(function(){
第50行                        var level0Val=$("#Level0").val();
第51行                        var level1Val=$("#Level1").val();
第52行                        alert(level0Val+"->"+level1Val);
第53行                        alert($("#Level0>option:selected").text()+"->"+$("#Level1>option:selecte
d").text());
第54行                    });
第55行                </Script>
第56行            </Body>
第57行        </Html>
```

例程7-8是例程7-7中JavaScript代码替换，是列表框的其他一些应用。

例程7-8

```
第1行     <Script language="JavaScript">
```

```
第2行          //当前选中条目的索引编号
第3行          $("#btnTest").click(function(){
第4行                  var nowSelectedIndex=$("#Level0").get(0).selectedIndex;//get(0)将jQuery转换为DOM元素
第5行                  alert(nowSelectedIndex);
第6行
第7行                  $("#Level0").get(0).selectedIndex=2;//选中编号为2的option，即上海
第8行
第9行                  var maxIndex=$("#Level0 option:last").index();//即最后一个option在同辈中的index编号
第10行                 alert(maxIndex);
第11行
第12行                 $("#Level0>option:eq(0)").remove();//删除编号为0的option元素
第13行         });
第14行   </Script>
```

第七节　文本框

文本框常用于网页文本输入，包括：中英文字、数字等。例程7-9用文本框输入图片数字，其效果如图7-6所示。当在文本框输入数字后，立即转换为对应的图片并呈现在文本框之前。文本框不能输入数字以外的其他内容，且支持backspace，用于修改输入错误。

图7-6　文本框输入图片数字

例程7-9

```
第1行   <Html>
第2行          <Head>
第3行                  <Title>文本框</Title>
第4行                  <Script language="JavaScript" src="jQuery-1.9.1.min.js"></Script>
第5行                  <Style type="text/css">
第6行                          #showImgNum img{Height:60px;}
第7行                          input{height:60px;line-height:60px;font-size:50px;border:0px;border-bottom:1px so
lid red;width:40px}
第8行                  </Style>
第9行          </Head>
第10行         <Body>
第11行                 <span id="showImgNum"></span><Input type=text id="inputNum">
第12行                 <Script language="JavaScript">
第13行                         $("#inputNum").focus();//让文本框获得焦点
第14行
第15行                         $("#inputNum").keydown(function(event){
第16行                                 var inputVal=event.which;//0-9的ASCII为48~57
第17行                                 //合规inputVal应介于48-57之间，同时允许回删键(backspace，ascii值为8)
第18行                                 if(!((inputVal>=48 && inputVal<=57) || inputVal==8))return false;
第19行                                 if(inputVal==8){//删除showImgNum中最后一个img元素
第20行                                         $("#showImgNum>img:last").remove();//选中最后showImgNum下最后一个
img元素并删除
第21行                                 }else{
```

```
第22行                                    var imgNumHtml=convertNum2Img(inputVal-48);
第23行                                }
第24行                                $("#showImgNum").append(imgNumHtml);
第25行                                return false;
第26行                            });
第27行
第28行                            function convertNum2Img(Num){//Num为正整数
第29行                                var temp=parseInt(Num);
第30行                                if(temp==0)return "<img src='0.jpg'>";
第31行                                var Result="";
第32行                                while(temp>0){
第33行                                    Result="<img src='"+temp%10+".jpg'>"+Result;
第34行                                    temp=(temp-temp%10)/10;
第35行                                }
第36行                                return Result;
第37行                            }
第38行                        </Script>
第39行                </Body>
第40行            </Html>
```

例程7-9第16行var inputVal=event.which;获得keydown输入内容，第18行对输入合法性进行判断，如果输入内容不介于48-57之间或8(backspace)，则return false;相当于取消(终止)本次输入。如果输入合法，则通过函数convertNum2Img()将数字转换为对应图片代码。如果输入backspace(第19-20行)，则删除最后一个图片数字，其实现代码是$("#showImgNum>img:last").remove();//选中id为showImgNum内最后img子元素，并删除。

例程7-9还对输入文本框进行修饰，以匹配图片尺寸，其仅有下边线。

例程7-9是文本框较为典型的应用，涉及到输入合法性处理、输入取消、键盘事件等，在实际中有较多应用。