单元 3

控件的使用

学习目标

【知识目标】

- 掌握使用服务器控件设计 Web 页面的方法。
- 掌握 ASP.NET 验证控件检验数据合法性的方法。
- 掌握用户控件设计可复用功能模块的方法。

【能力目标】

- 能够熟练使用服务器控件设计交互功能较好的 Web 页面。
- 能够熟练使用ASP.NET验证控件检验数据合法性。
- 能够使用户控件设计可复用的功能模块。

【素质目标】

- 养成良好的编码习惯。
- 提高语言表达、文字写作能力。

教学PPT

PPT

教学录像

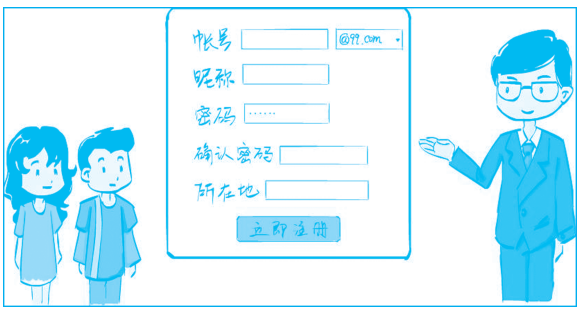
源代码

教学案例

引例描述

小李同学通过前面的学习发现，在清楚理解电子商务购物网站项目的系统分析与环境构建、项目的整体规划与布局后，该如何对具体设计代码网站的具体页面？网页如何与用户进行交互进而完成购物？小李去请教王老师，王老师告诉他，使用服务器控件可以有效地解决这些问题。

ASP.NET 的网页就像是由零件构成的汽车，网页中的标准控件是小零件、容器控件是较大的零件，而用户控件是用户自行设计的能够重复使用的零件。



ASP.NET Web 应用程序建立在浏览器 / 服务器（B/S）结构的基础上，Web 程序不仅要能够处理客户端事务，而且要实现与服务器端交互，微软公司在 ASP.NET 中创建了很多专门运行于服务器端的 Web 控件，称为 ASP.NET 服务器控件。

ASP.NET Web 窗体是一种声明性的文本文件，其扩展名为 aspx，一个窗体包含多个元素以执行不同操作，最终输出 HTML 网页内容。为完成服务器（Server）与浏览器（Browser）的动态交互，通常页面包含不同功能类型的控件，这些控件位于该页面的 form 表单元素内，且该表单元素必须具有 runat 属性，其值为 server。

```
<form id="form1" runat="server">
```

在创建 ASP.NET 网页时，按照功能分类，主要使用 HTML 服务器控件和 Web 服务器控件。除内置控件外，ASP.NET 页面框架还提供了创建用户控件和自定义控件的能力，用户控件和自定义控件可以增强和扩展现有控件以构建更丰富的用户界面。

通过本单元的学习，使读者理解 ASP.NET 服务器控件的功能、与传统 HTML 控件的区别，掌握服务器控件的基本属性、基于控件对象的事件驱动编码机制。能够熟练使用基本服务器控件设计具有交互功能的动态页面、理解验证控件的验证功能意义，熟练使用其完成服务器控件输入数据的合法性验证功能。

在掌握基本服务器控件的基础上，重点学习几种常用的高级服务器控件：

日历控件、文件上传控件，能够使用其完成日期生成、文件上传等常用功能模块；结合代码的重用思想理解用户控件的创建并掌握其使用方法。

任务 3.1　使用基本服务器控件设计用户登录页面

基本服务器控件的使用

任务陈述

任务构思与目标：根据需求新建 SelectOKShop 网站的用户登录页面，页面效果如图 3-1 所示，并添加用户注册页的链接，在网页的最上端实现“设为主页”功能和时间显示功能，在网页最下端实现“联系我们”邮件发送功能。

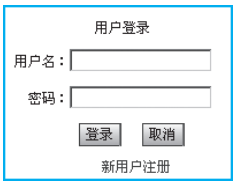


图 3-1　用户登录

任务设计：添加 Index.aspx 页面，作为网站首页。根据文本类型服务器控件和按钮类型服务器控件功能设计用户登录页面，设计两个 TextBox 控件用于接收用户输入的用户名和密码，两个 Button 按钮控件用于进行“确定”和“取消”操作。并运用页面布局知识实现控件的布局，并在当前页面弹出提示框显示正确添加的用户账户相关信息，从而实现网站登录功能。

知识准备

3.1.1　ASP.NET 控件概述

服务器控件是相对于客户端控件而言的，客户端控件是指运行于客户端的 HTML 控件，而服务器控件则包含运行于服务器端的控件，ASP.NET 之所以开发方便和快捷，关键是它有一组强大的控件库。ASP.NET 控件包括 Web 服务器控件、HTML 服务器控件和 HTML 控件三类，三者的区别如下：

1. HTML 控件

即通常说的 HTML 标记语言，其在已往的静态页面和其他网页里存在，不能在服务器端控制的，只能在客户端通过 javascript 和 vbscript 等程序语言进行控制。例如：

```
<input type="button" id="btn" value="button"/>
```

2. HTML 服务器控件

即在 HTML 控件的基础上添加 runat="server" 所构成的控件，其与 HTML 控件的区别是运行方式不同，HTML 控件运行在客户端，而 HTML 服务器控件运行在服务器端。当 ASP.NET 网页执行时，会检查标注是否有 runat 属性，如果标注没有设定，则 HTML 标注就会被视为符串，并被送到字符串流

等待传输到客户端，客户端的浏览器会对其进行解释；如果 HTML 标注有设定 runat="server" 属性，Page 对象会将该控件放入控制器，服务器端的代码就能对其进行控制，控制执行完毕后再将 HTML 服务器控件的执行结果转换成 HTML 标注，并作为字符串流发送到客户端进行解释。例如：

```
<input id="Button" type="button" value="button" runat="server" />
```

3. Web 服务器控件

也称为 ASP.NET 服务器控件，是 Web Form 编程的基本元素，也是 ASP.NET 所特有的控件。它会按照 Client 的情况产生一个或者多个 HTML 控件，而不是直接描述 HTML 元素。例如：

```
<asp:Button ID="Button1" runat="server" Text="Button"/>
```

Web 服务器控件和 HTML 服务器控件的区别如下。

（1）Web 服务器控件提供更加统一的编程接口，如每个 Web 服务器控件都有 Text 属性。

（2）隐藏客户端的不同，使程序员可以把更多的精力放在业务上，而不用去考虑客户端的浏览器的具体种类。

（3）Web 服务器控件可以保存状态到 ViewState 里，这样页面在从客户端回传到服务器端或者从服务器端下载到客户端的过程中都可以保存。

（4）事件处理模型不同，HTML 标注和 HTML 服务器控件的事件处理都是在客户端的页面上，而 Web 服务器控件则是在服务器上，例如：

```
<input id="Button1" type="button" value="button" runat="server"/>
```

Button1 为 HTML 服务器控件，但此时单击此按钮，页面不会回传到服务器端，原因是没有为其定义鼠标单击事件。

```
<input id="Button1" type="button" value="button" runat="server" onserverclick=
"test"/>
```

为 HTML 服务器控件添加 onserverclick 事件，单击此按钮页面会发回服务器端，并执行 test(object sender, EventArgs e) 方法。

```
<asp:Button ID="Button1" runat="server" Text="Button"/>
```

Button1 是 Web 服务器控件，并且没有为其定义 Click，但是单击该按钮时，页面也会发回到服务器端。

由此可见：HTML 标注和 HTML 服务器控件的事件是由页面来触发的，而 Web 服务器控件则是由页面把 Form 发回到服务器端，由服务器进行处理。

提示：HTML 控件添加 runat="server" 属性，就转换为 HTML 服务器控件，同时要添加 id 属性来标识该服务器控件。所有 HTML 服务器控件必须位于带有 runat="server" 属性的 <form> 标签内。runat="server" 属性指示该表单应在服务器进行处理，同时指示其包括在内的控件可被服务器脚本访问。

【实例 3-1】　使用 HTML 服务器控件设计如图 3-2 所示的报名调查信息表。

源代码

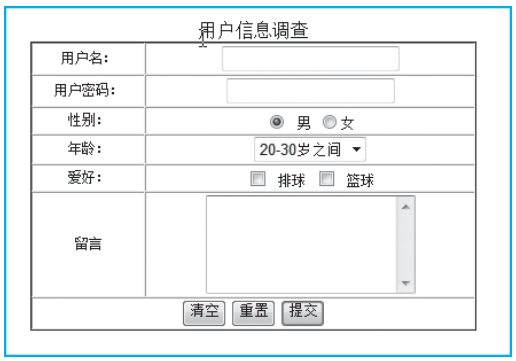


图 3-2　报名信息调查表

【页面设计代码】Default1.aspx

```
    <form id="form2" runat="server">
    <div style="text-align: center"> 用户信息调查
        <table style="width: 389px; height: 52px;border:1px solid black" cellspacing ="0" rules="all">
            <tr>
                <td rowspan="6"><img src="Images/reg-1.gif" alt=" 图片 "/></td>
                <td > <span style="font-size: 9pt"> 用户名：</span></td>
                <td > <input id="Name" type="text" runat="server" /></td>
            </tr>
            <tr>
                <td > <span style="font-size: 9pt"> 用户密码：</span></td>
                <td > <input id="Password" type="password" runat="server" /></td>
            </tr>
            <tr>
                <td > <span style="font-size: 9pt"> 性别：</span></td>
                <td >  <input id="Male" type="radio" name="gender" value =" 男
"runat="server" /><span style="font-size: 9pt"> 男  <input id="Female" type="radio"
name="gender" value =" 女 "runat="server"/> 女 </span></td>
            </tr>
            <tr>
                <td> <span style="font-size: 9pt"> 年龄：</span></td>
                <td>          <select id="Age" runat="server">
                              <option value ="1">20-30 岁之间 </option>
                              <option value ="2">30-40 岁之间 </option>
                              <option value ="3">40-50 岁之间 </option>
                              <option value ="4">50-60 岁之间 </option>
                              </select>
                </td>
             </tr>
              <tr>
```

```
                    <td > <span style="font-size: 9pt"> 爱好：</span></td>
                    <td ><input id="Ball1" type="checkbox" runat="server" checked=
"checked" value =" 排球 " />
                                <span style="font-size: 9pt"> 排球 </span>
                                <input id="Ball2" type="checkbox" runat="server" value=" 篮球 "/>
                                <span style="font-size: 9pt"> 篮球 </span>
                    </td>
                </tr>
                <tr>
                    <td > <span style="font-size: 9pt"> 留言 </span></td>
                    <td > <textarea id="Liuyan" cols="20" rows="5" runat ="server" ></textarea></td>
                </tr>
                <tr>
                    <td colspan="3" class="style1">
                      <input id="Button" type="button" value=" 清空 " runat="server" onclick=
"ServerClick" />
                       <input id="Reset" type="reset" value=" 重置 " runat="server"    />
                      <input id="Submit" type="submit" value=" 提交 " runat="server" onserverclick=
"Submit_ServerClick" />
                       </td>
                </tr>
            </table>
        <span id="Message" runat="server" />
    </div>
```

HTML 服务器控件最常用的控件事件包括 OnServerEvent 。通过为 ServerClick 事件提供自定义事件处理程序，可以在单击控件时执行特定的指令集。例如，< 输入类型 = 按钮 > 控件有一个 OnServerClick 事件。用户单击 HtmlInputButton 控件时，该控件所在窗体的输入内容被发送到服务器并得到处理。最后，将响应发送回浏览器。

【程序代码】Default1.aspx.cs 页面代码：

```
    <head id="Head1" runat="server">                                    // 服务器端脚本
        <script language="javascript" type="text/javascript">      // 单击提交按钮
      protected void Submit_ServerClick(object sender, EventArgs e)
            {
                string love = "";   // 获取用户名和密码信息
                Message.InnerHtml = "<h4> 您输入的个人信息为 </h4>";
                Message.InnerHtml += " 姓名：" + Name.Value + "<br/>";
                Message.InnerHtml += " 密码：" + Password.Value + "<br/>";
                if (Male.Checked) Message.InnerHtml += " 性别：" + Male.Value + "<br/>";
                if (Female.Checked) Message.InnerHtml += " 性别：" + Female.Value + "<br/>"; //
获取性别信息
                if(Age.Value == "1") Message.InnerHtml += " 你的年龄在 20-30 岁之间 "+   "<br/>";
                if(Age.Value == "2") Message.InnerHtml += " 你的年龄在 30-40 岁之间 "+   "<br/>";
                if(Age.Value == "3") Message.InnerHtml += " 你的年龄 40-50 岁之间 "+   "<br/>";
```

```
                if (Age.Value == "4") Message.InnerHtml += " 你的年龄 50-60 岁之间 " +
"<br/>"; // 获取年龄信息
                if (Ball1.Checked) love += Ball1.Value;
                if (Ball2.Checked) love += Ball2.Value;
                if (Ball2.Checked && Ball1.Checked) love = Ball1.Value + "、" + Ball2.Value;
                Message.InnerHtml += " 你的爱好为：" + love + "<br/>"; // 获取爱好信息
                Message.InnerHtml += " 你的留言为：" + Liuyan.Value + "<br/>"; // 获取
留言信息
            }
        </script>
    </head>
```

3.1.2　Web 服务器控件

Web 服务器控件位于 System.Web.UI.Webcontrols 命名空间中，并集成在 ASP.NET 的基本类库中，是 ASP.NET 中功能强大的编程单元，使用 Web 服务器控件可以实现复杂的事务处理。Web 服务器控件位于 Visual Studio.NET 2010 工具箱的标准组、数据组、导航组、验证组中。

1. Web 服务器控件格式

```
<asp:control_name id="……" runat="server" />
```

Web 服务器控件以“asp:”作为前缀标志，以“/”作为结束标志。所有属性均定义在一对尖括号“<>”内，属性之间用空格分开。

```
例：<asp:Button ID="Button1" runat="server" Text="Button" />
    <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
```

2. Web 服务器控件的类型

常用的 Web 服务器控件请见表 3-1。

表 3-1　常用的 Web 服务器控件一览表

Web 服务器控件类别	Web 服务器控件名称
文本类型控件	Label、TextBox、Literal 控件
按钮类型控件	Button、LinkButton、HyperLink 和 ImageButton 控件
选择类型控件	CheckBox、RadioButton、RadioButtonList、CheckBoxList、DropDownList、ListBox 控件
数据绑定控件	DataGrid、DataList 和 Repeater 控件
图像控件	Image 控件
验证控件	RequiredFieldValidator、RangeValidator、ValidationSummary 、RegularExpressionValidator、CompareValidator、CustomValidator 控件
导航控件	TreeView、Menu、SitMapPath 控件
广告控件	AdRotator 控件
视图控件	MultiView、View 控件
日历控件	Calendar 控件

以上控件种类和功能各不相同，使用控件就是根据需求正确设置或获取控件的属性、调用控件的方法和为控件编写相应的事件处理程序，Web 服务器控件虽然种类较多，但是各控件之间有相通之处，如 Web 服务器控件部分共用的常用属性见表 3-2。

表 3-2 Web 服务器控件公用的常用属性

属 性	说 明
ID	控件在程序中使用的名称
AccessKey	获取或设置控件的键盘快捷键
BackColor	获取或设置控件的背景色
BorderColor	获取或设置控件的边框颜色
BorderWidth	控件的边框宽度
BorderStyle	控件的边框样式
CssClass	分配给控件的样式表类
Enable	控件是否有效
Font	控件的字体
ForeColor	控件的前景色
Height	控件的高度
Width	控件的宽度
Visible	控件是否可见
ToolTip	获取或设置当用户将鼠标指针停放在控件上时显示的文本

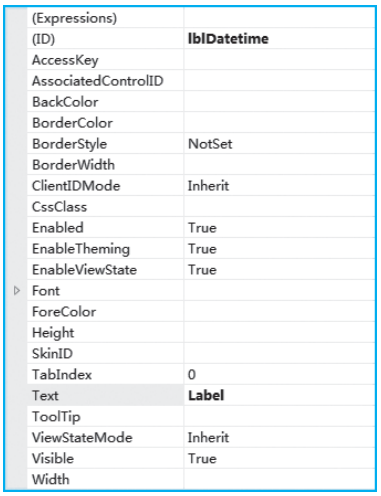
微课 3.1.1 文本类控件 1

微课 3.1.2 文本类控件 2

3.1.3 文本类型控件

1. Label 控件

Label 控件又称标签控件，主要用来显示文本信息。该控件功能较单一，除了表 3-2 所列的通用属性外，其最常用的属性是 Text 属性，其属性设置如图 3-3 所示。可以通过修改属性值更改控件属性。



(Expressions)	
(ID)	**lblDatetime**
AccessKey	
AssociatedControlID	
BackColor	
BorderColor	
BorderStyle	NotSet
BorderWidth	
ClientIDMode	Inherit
CssClass	
Enabled	True
EnableTheming	True
EnableViewState	True
Font	
ForeColor	
Height	
SkinID	
TabIndex	0
Text	**Label**
ToolTip	
ViewStateMode	Inherit
Visible	True
Width	

图 3-3 Label 标签控件的属性窗口

2. TextBox 控件

TextBox 控件用于输入文字信息，ASP.NET 网页中具有文本输入功能的只有 TextBox 控件，通过设置控件的 TextMode 属性来区分文本、密码和多行文本输入方式，控件的常用属性及方法见表 3-3。

⚠ 注意：文本框内容是字符串类型，如果要做计算，需要作类型转换。如 **convert.tosingle** 转成单精度，或 **single.parse()** 符号的中英文区别。

表 3-3　TextBox 控件的常用属性和方法

属性或方法	说　明
AutoPostBack	文本框内容发生变化，并且输入焦点离开文本框（TAB,ENTER），是否自动将文本框内容发回服务器。
Text	文本框中内容
TextMode	SingleLine 单行输入模式，默认 MultiLine 为多行模式，该模式时文本放不下时，会出现流动条 Password 密码输入
Columns	以字符为单位指明文本框的显示宽度
Rows	当 TextMode 为 MultiLine 时，指明文本框的行数
MaxLength	在单行文本方式下，文本框可以输入的字符数，最大长度可达 2147483647 个字符，当 Textmode 属性设置为 Multiline 时，该属性不适用
Wrap	用于获取或设置一个布尔值，该值指示文本框内的文本是否可以换行。当属性为 True（默认）时，文本框中的内容可以换行；只有 TextMode 属性设置为 MultiLine 时，该属性才适用
ReadOnly	输入框为只读，当值为 True 时，禁止更改，默认值为 False
DataBind()	将数据源绑定到被调用的服务器控件及其所有子控件上
TextChanged()	当文本框内容发生变化时触发该方法

TextBox 控件大部分属性设置和 Label 控件类似，下面主要介绍控件的 TextChanged 事件，改变了文本框的内容并按回车，或通过程序改变了 text 属性值，将会触发 textchange 事件，并执行相关的事件过程。TextBox 控件 AutoPostBack 属性是一个布尔值，用于指定当用户改变文本框的内容时，是否向服务器进行自动回送并引发 TextChanged 事件，所以要把 AutoPostBack 属性设置为 True。

⚠ 注意：更改内容后，只有当用户按了 <Enter> 键或当文本框失去焦点时才发生回送动作。事件触发的时候会重新载入页面，也就是还是会执行一遍 page_load，该操作有时候也会造成错误。

【实例 3-1】　触发 TextBox 控件的 TextChanged 事件，当文本内容修改或改变后弹出提示框。

首先，通过工具箱将 TextBox 控件拖放到页面相应位置上，修改控件 ID 属性为“txtDemo”，将控件的 AutoPostBack 属性设置为 True，选择控件属性窗口中的⚡符号，打开如图 3-4 所示的控件对应的事件窗口，在【TextChanged】后的空白处双击鼠标左键，此时将在页面的 .cs 文件中生成 txtDemo_TextChanged 事件方法，在方法中输入如下代码：

```
protected void txtDemo_TextChanged(object sender, EventArgs e)
{
    Response.Write("<script> alert(' 你修改了文本框的值！ ')</script>");
}
```

网页运行后，在文本框中输入内容后按【回车】键，将弹出如图 3-5 所示

的对话框，表示触发了 TextChanged 事件。

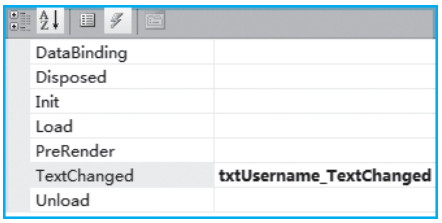


图 3-4 TextBox 控件的事件设置

图 3-5 TextChanged 事件运行效果

⚠ 注意：软件开发是团队成员协作的结果，而编码规范是程序员之间沟通的桥梁。如果每个成员遵循一致的编码风格，则可以减少沟通所需的工作量。减轻程序员的负担。命名规范是一种约定，减少了编码的自由度，对于减少人员之间通信工作量，提高工作效率意义重大。具体编码规范，详见本书附录 A。

3.1.4 按钮类型控件

1. Button 控件

Button 控件是 ASP.NET 开发时最常见的控件之一，Button 控件常见的属性和事件见表 3-4。

表 3-4 Button 控件常见的属性和事件

属性或方法	类 型	说 明
CausesValidation	bool	指示在单击 Button 控件时是否执行验证
CommandArgument	string	该参数会传递到 Command 事件
CssClass	string	该参数指示控件在客户端呈现的级联样式表 (CSS) 类
Enabled	bool	指示是否启用 Web 服务器控件
OnClientClick	string	Button 控件的 Click 事件时所执行的客户端脚本
Text	string	在 Button 控件中显示的文本标题
ValidationGroup	string	Button 控件回发到服务器时要进行验证的控件组
Click	EventHandler	单击 Button 控件触发的事件
Command	CommandEventHandler	单击 Button 控件触发的事件

微课 3.1.3 按钮类控件

本节主要介绍 Button 控件的 CommandArgument、CssClass、OnClientClick 属性、Click 事件及 Command 事件。

CssClass 用于设置控件的 CSS 属性，如字体颜色、背景颜色、边框等，其值应与样式表中的类名相对应。

OnClientClick 用于设置客户端单击所发生的事件，如某些重要操作需要用户进行确认，可以通过 javascript 的 confirm() 函数来实现。

CommandArgument 用于设置传递到 Command 事件的参数。

Click 和 Command 事件都是单击 Button 按钮时发生的事件，但处理的委托类型不同，触发 Click 事件由 EventHandler 委托进行处理，触发 Command 事件后由 CommandEventHandler 委托进行处理，其声明格式如下：

```
public delegate void EventHandler (    Object sender, EventArgs e)
public delegate void CommandEventHandler ( Object sender,    CommandEventArgs e )
```

Click 和 Command 事件都有两个参数，第一个参数表示由哪个控件触发了事件，第二个参数表示发生事件时的一些事件数据。这两个委托第一个参数都是相同的，第二个参数不同，EventArgs 类不带有任何事件数据（该委托很常见，不关心事件数据的事件都由该委托处理），而 CommandEventArgs 可以附带事件数据，其包括两个重要属性：CommandArgument 和 CommandName。CommandArgument 属性可以附带一些参数信息，CommandName 用于设置命令的名称。

【实例 3-2】 创建一个页面命名为“Button.aspx”，实现按钮的三种提交功能。

前台代码如下：

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Button.aspx.cs"
Inherits="_Button" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"  "http://www.
w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head id="Head1" runat="server">
    <title>ASP.NET 基本控件 Button 控件 </title>
    <style type="text/css">
    .btn
    {
        color:#ffffff;
        font-weight:bold;
        background-color:#cc3300;
        border:none;
    }
    </style>
</head>
<body>
    <form id="form2" runat="server">
    <div>
        <asp:Button ID="btnOK" runat="server" Text=" 提交 " OnClick="btnOK_Click"
OnClientClic.k="javascript:return confirm(' 确认提交 ');" OnCommand="btnOK_Command"
CommandArgument="1" CssClass="btn" />
    </div>
    </form>
</body>
</html>
```

后台程序代码如下：

```
public partial class _Button : System.Web.UI.Page
    {
```

```
        protected void btnOK_Click(object sender, EventArgs e)
        {
            btnOK.Text = " 发生了 Click 事件 ";
        }
        protected void btnOK_Command(object sender, CommandEventArgs e)
        {
            Response.Write("<script>alert(' 发生了 Command 事件事件的数据是：" +
e.CommandArgument.ToString() + "');</script>");
        }
    }
```

浏览器中的 HTML 代码如下：

```
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.
w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
    <html xmlns="http://www.w3.org/1999/xhtml" >
    <head><title>
        ASP.NET 基本控件 Button 控件
    </title>
        <style type="text/css">
        .btn
        {
            color:#ffffff;
            font-weight:bold;
            background-color:#cc3300;
            border:none;
        }
        </style>
    </head>
    <body>
        <form name="form1" method="post" action="Button.aspx" id="form2">
    <div>
    <input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE" value="/wEPDwU
KMTQ2OTkzNDMyMWRk8oOKWA57ZbXtacIpYdP9o2tVOhc=" />
    </div>
        <div>
            <input type="submit" name="btnOK" value=" 提交 " onclick="javascript:return
confirm(' 确认提交 ');" id="btnOK" class="btn" />
        </div>

    <div>
        <input type="hidden" name="__EVENTVALIDATION" id="__EVENTVALIDATION"
value="/wEWAgLc1o/jAgLdkpmPAWUkmFpAVODohFDhiNfzIGu4jHdr" />
    </div></form>
    </body>
    </html>
```

可见默认情况下，Button 控件被解释成 <input type="submit" …/> 形式的提交按钮，且该按钮的客户端 id（即在 HTML 代码中的 id 属性）和服务器端指定的 id 是一致的。此外，在设计时表单的代码是“<form id="form1" runat="server">”，在客户端为“<form name="form1" method="post" action="Button.aspx" id="form1">”，其原因是表单发送到客户端时被设置成 post 提交方式，接收表单数据的页面是当前页面。运行效果分别如图 3-6、图 3-7、图 3-8 和图 3-9 所示。

图 3-6　运行初始效果图

图 3-7　单击提交后效果图

图 3-8　单击确定后效果图

图 3-9　最终效果图

文本类控件的使用

【实例 3-3】 文本类控件的应用——设计如图 3-10 所示的留言板。

页面设计代码如下：

```
    <span style="color: #009966"><span style="background-color: #ffccff"> 通过该页面添加新留言到数据库，点击添加按钮添加！ </span><br /> </span>
        </div> <!--asp.net 控件文本类按钮控件的简单使用 -->
          <asp:Panel ID="Panel1" runat="server" Height="221px" Width="557px" BorderColor="Red" BorderStyle="Dashed" BorderWidth="1px">
            <asp:Label ID="Label1" runat="server" Text=" 留言者："></asp:Label>
            <asp:TextBox ID="tbName" runat="server" Width="189px"></asp:TextBox>
            <br />
            <asp:Label ID="Label2" runat="server" Text=" 标题 " Width="63px"></asp:Label>
            <asp:TextBox ID="tbTitle" runat="server" Width="476px"></asp:TextBox><br />
            <asp:Label ID="Label3" runat="server" Text=" 内容 " Width="63px"></asp:Label><br />
             <asp:TextBox ID="tbContent" runat="server" TextMode="MultiLine" Height="80px" Width="539px"></asp:TextBox>
            <br />
            <asp:Label ID="lbHint" runat="server" ForeColor="Red"></asp:Label><br />
            <asp:Button ID="btnAdd" runat="server" BackColor="#FFC080" BorderColor="#C0FFC0"
             BorderStyle="Ridge" Font-Bold="True" Font-Names=" 宋体 " Font-Overline="False" Font-Size="Larger" OnClick="btnAdd_Click" Text=" 留言 " Width="112px" />
            </asp:Panel>
            <br />
```

程序运行效果如图 3-10 所示。

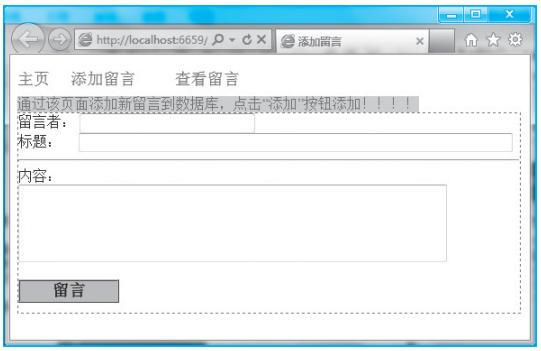


图 3-10 留言板的设计

2. LinkButton 控件与 HyperLink 控件

LinkButton 控件用于创建超链接样式的按钮，HyperLink 控件用于创建超链接。

⚠ 注意：该控件的外观与 HyperLink 控件相同，但其功能与 Button 控件一样。

HyperLink 控件只是产生一个有 URL 指向的超级链接，而 LinkButton 控件属于 Button 类控件。LinkButton 控件本身支持事件处理，并没有 HyperLink 的 NavigateUrl 属性，其 URL 链接功能主要由事件处理完成，LinkButton 控件支持 OnClick、OnCommand 等服务器端方法。LinkButton 的外观和风格与 HyperLink 相同，其优势还包括点击时能够返回同一个网页且便于使用的 OnClick 方法。 LinkButton 控件的常用属性见表 3-5。

表 3-5 LinkButton 控件的常用属性

属　性	描　述
CommandArgument	有关所执行命令的附加信息
CommandName	与 Command 事件相关的命令
OnClientClick	当 LinkButton 控件被单击时被执行的函数的名称
PostBackUrl	当 LinkButton 控件被单击时从当前页面进行回传的目标页面的 URL
Text	LinkButton 上的文本

本节应用 LinkButton 的属性 CommandName 和方法 Command 来实现。

首先，了解 LinkButton 的几个比较重要的属性和方法。

CommandName 属性：取得或设定与 LinkButton 控制项相关的命令名称。该值与 CommandArgument 属性一起传给 Command 处理事件。

CommandArgument 属性：包含有关命令的补充资讯，如 Ascending 排序顺序，与 CommandName 一起使用。

Click 事件：该事件一般在没有命令名与 LinkButton 控制关联时（如“提交”按钮）使用。

Command 事件：当单击 LinkButton 控件时会触发 Command 事件。当命令名（如 Sort）与 LinkButton 控件关联时，通常使用该事件。这样可以实现在一个网页上创建多个 LinkButton 控件，并以编程方式确定单击了哪个 LinkButton 控件。

了解 LinkButton 的属性和方法后，可以在程序中给 LinkButton 的 CommandName 属性和 CommandArgument 属性赋值，通过 Command 事件，即可从 CommandEventArgs 类中得到数据，进而判断是哪个 LinkButton 被触发了。其中，CommandEventArgs 类存储了和按钮 (Button) 事件相关的数据，并且可以在事件处理中通过 CommandEventArgs 类的属性访问这些数据。

【实例 3-4】 LinkButton 控件的应用。

```
<Form Id="Form2" Runat="Server">
<ASP:LinkButton Id="LB1" Text=" 点击我 " OnClick="LB1_Click" Runat="Server"/><p>
<ASP:Label Id="L1" Text=" 现在是 A" Runat="Server" />
</Form>

void LB1_Click(Object Sender, Eventargs e)
    {
        L1.Text=" 现在是 B" ;
}
```

HyperLink 控件的常用属性见表 3-6。

表 3-6　HyperLink 控件的常用属性

属　性	描　述
ImageUrl	显示此链接的图像的 URL
NavigateUrl	该链接的目标 URL
Target	URL 的目标框架
Text	显示该链接的文本

【实例 3-5】 HyperLink 控件的应用。

```
<html>
    <body>
        <form id="Form2" runat="server">
         <asp:HyperLink  ID="HyperLink1"  ImageUrl="/images/qtc.gif"
NavigateUrl=http://www.qtc.edu.cn Text=" 青岛职业技术学院 "  Target="_blank"
runat="server" />
        </form>
    </body>
</html>
```

3. ImageButton 控件

ImageButton 控件用于在用户界面中建立一个图片按钮。ImageButton 控件的常用属性见表 3-7。

表 3-7 ImageButton 控件的常用属性

属 性	说 明
AlternateText	获取或设置当图像不可用时，ImageButton 控件中显示的替换文本。
ImageUrl	获取或设置在 ImageButton 控件中显示的图像的位置。
ImageAlign	获取或设置 ImageButton 控件相对于 Web 页上其他元素的对齐方式。
PostBackUrl	获取或者设置单击 ImageButton 控件时从当前页发送到网页的 URL
Click	在单击 ImageButton 时发生的事件。

【实例 3-6】 ImageButton 控件的应用。

```
<%@ Page Language="C#" AutoEventWireup="True" %>
<html>
<head>
<script language="C#" runat="server">
void ImageButton_Command(object sender, CommandEventArgs e)
{
    if (e.CommandName == "Sort" &&e.CommandArgument==" 闪烁 ")
        Label1.Text = " 你点击了闪烁图片 ";
    else
        Label1.Text = " 你点击了工作图片 ";
}
</script>
</head>
<body>
<form id="Form2" runat="server">
<h3> 控件示例 </h3> 点击图片：<br><br>
<asp:ImageButton ID="ImageButton1" runat="server" AlternateText=" 闪 烁 "
ImageUrl="images/blank.gif"
OnCommand="ImageButton_Command" CommandName="Sort" CommandArgument=
" 闪烁 "/>
<asp:ImageButton ID="ImageButton2" runat="server" AlternateText=" 工作 " ImageUrl=
"image/workplan.gif"
OnCommand="ImageButton_Command" CommandName="Sort" CommandArgument=
" 工作 "/>
<br><br>
<asp:label ID="Label1" runat="server"/>
</form>
</body>
</html>
```

文本类、控制类控件的使用

任务实施

3.1.5 网站登录模块的实现

1. 设计网页布局

在 Index.aspx 页中的 ContentPlaceHolderID 为“ContentPlaceRight”的控件

中设计登录网页的布局，在 div 中设计一个 Table，要求五行两列，并进行相应的布局设计。

2. 设计控件

从工具箱拖放两个 TextBox 控件，分别放在第 2 行第 2 个单元格和第 3 行第 2 个单元格中，在第四行第 1 个单元格，和第 4 行第 2 个单元格中分别拖放两个按钮，在第五行中拖放一个 LinkButton 控件，具体代码如下：

微课 3.1.4　网站登录模块的实现

```
<div align="right">
<table style="width: 100%;">
        <tr>
            <td colspan="2" align="center">
                用户登录
            </td>
        </tr>
        <tr>
            <td>
                用户名：
            </td>
            <td>
                <asp:TextBox ID="txtUsername" runat="server" Columns="20"
    MaxLength="10" Rows="5"></asp:TextBox>
            </td>
        </tr>
        <tr>
            <td>
                密码：
            </td>
            <td>
                <asp:TextBox ID="txtPWD" runat="server" Columns="20"
    TextMode="Password" ></asp:TextBox>
            </td>
        </tr>
        <tr>
            <td align="center">
                   <asp:Button ID="btnOK" runat="server" Text=" 登录 "
onclick="btnOK_Click"
    AccessKey="R" ToolTip=" 后台登录 "    CssClass="btn"    />
            </td>
            <td align="center">
                <asp:Button ID="btnCancle" runat="server" Text=" 取消 " CssClass="btn" />
            </td>
        </tr>
        <tr>
            <td colspan="2" align="center">
                <asp:LinkButton ID="lbtnRegister" runat="server" Text=" 新用户注册 "
    PostBackUrl="~/Register.aspx"></asp:LinkButton>
```

```
                </td>
            </tr>
        </table>
    </div>
```

生成视图如图 3-1 所示。

3. 代码设计

双击【登录】按钮，触发按钮“Click”事件，实现用户模拟验证功能，编写代码如下：

```
protected void btnOK_Click(object sender, EventArgs e)
    {
        if (txtUsername.Text == "xzp" && txtPWD.Text == "123")
        {
            Response.Write("<script language='javascript'> alert(' 登录成功！ ')</script>");
        }
      else
      {
        Response.Write("<script> alert(' 登录失败，请重新注册！ ')</script>");
      }
}
```

当输入正确的用户名和密码时，登录模块运行效果如图 3-11 所示。

4. 设置主页

从工具箱拖放一个 ImageButton 和一个 LinkButton 控件放在 Main.Master 母版页的最上部，为 ImageButton 控件的 ImageUrl 属性指定要显示的主页的图标，为 LinkButton 控件的 onclientclick 设置设为主页相关的指令，效果如图 3-12 所示，代码如下：

```
    <asp:ImageButton ID="ibtnMainPage"  runat="server"  ImageUrl="~/images/
mainpage.ico" Width="20px" Height="20px" />
    <asp:LinkButton ID="lbtnMainPage" runat="server" Text=" 设为主页 " onclientclick="this.style.
behavior='url(#default#homepage)';this.setHomePage('http://www.qtc.edu.cn')" ></asp:LinkButton>
```

图 3-11 登录成功效果图

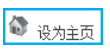


图 3-12 设为主页

5. 制作网站的“联系我们”

从工具箱拖放一个 HyperLink 控件放在 Main.Master 母版页的最底部，设置 HyperLink 控件的 NavigateUrl 属性值为“mailto:xuzhp@qtc.edu.cn”，当点击【联系我们】超链接控件时，将打开 Outlook 窗口发送邮件。代码如下：

```
<asp:HyperLink ID="HyperLink1" runat="server" NavigateUrl="mailto:xuzhp@qtc.edu.cn" Text=" 联系我们 "> 联系我们 </asp:HyperLink>
```

任务拓展

微课 3.1.5　认识 Page 对象

3.1.6　认识 Page 对象

1. Page 对象概述

在 ASP.NET 中每个页面都派生自 Page 类，并继承该类所有的公开方法和属性。Page 类与扩展名为 .aspx 的文件相关联，这些文件在运行时被编译为 Page 对象，并被缓存在服务器内存中。Page 类常用的属性见表 3-8。

表 3-8　Page 类常用的属性和方法

属性与方法	说　明
IsPostBack	该属性可以检查 .aspx 页是否为传递回服务器的页面，常用于判断页面是否是首次加载
IsValid	该属性用于判断页面中的所有输入的内容是否已经通过验证，是一个布尔值的属性。当需要使用服务器端验证时，可以使用该属性
IsCrossPagePostBack	该属性判断页面是否使用跨页提交，是一个布尔值的属性
SetFocus	将浏览器焦点设置为指定控件
OnLoad	触发 Load 事件
FindControl	在页面容器中搜索指定的服务器控件
MapPath	搜索虚拟路径（绝对的或相对的）或应用程序相关的路径映射到的物理路径
Eval	为在运行时根据对象分析和计算数据绑定表达式提供支持

Page 页面第一次执行时，有一个编译的过程，即在页面执行之前需要首先将 .aspx 页及其后台代码编译成页面类，然后才执行页面中的处理，而第二次执行时，由于页面类已存在，不需经过编译过程，所以执行时间比第一次要短。

Page 对象是指向页面自身的方式，在整个页面的执行期内，都可以使用。每个 Page 页的头部显示如下：

```
<% @ Page Language="C#" CodeFile="default.aspx.cs" Inherits="_default" trace="true"%>
```

@Page 指令定义了 ASP.NET 页用于编译和解析的属性，每个 aspx 页面只能有一个 @Page 指令。参数代码含义如下。

① Language: 指定页面代码和后置代码使用的语言，但只支持微软 .NET 框架中的语言。

② AutoEventWireup: 设置页面是否自动调用网页事件，默认为 true。

③ CodeFile: 指定代码后置文件名，后置代码，该后置代码与页面是局部关系。

④ Inherits: 页面类。

⑤ trace: 调试跟踪，如：<trace enabled="true" pageOutput="true" traceMode="">。

其中，设置调试跟踪后，程序运行结果如图 3-13 所示。

跟踪信息

类别	消息
aspx.page	Begin PreInit
aspx.page	End PreInit
aspx.page	Begin Init
aspx.page	End Init
aspx.page	Begin InitComplete
aspx.page	End InitComplete
aspx.page	Begin PreLoad
aspx.page	End PreLoad
aspx.page	Begin Load
aspx.page	End Load
aspx.page	Begin LoadComplete
aspx.page	End LoadComplete
aspx.page	Begin PreRender
aspx.page	End PreRender
aspx.page	Begin PreRenderComplete
aspx.page	End PreRenderComplete
aspx.page	Begin SaveState
aspx.page	End SaveState
aspx.page	Begin SaveStateComplete
aspx.page	End SaveStateComplete
aspx.page	Begin Render
aspx.page	End Render

图 3-13 调试跟踪

2. Page 页的生命周期

（1）单独的 Page 页面的事件执行顺序

当 Page 页执行时，其运行的事件顺序如图 3-14 所示。

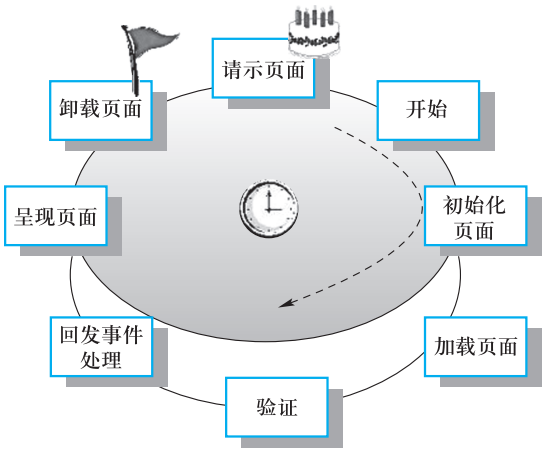


图 3-14 Page 页的生命周期

① Page.PreInit 在页初始化开始时发生。

② Page.Init 当服务器控件初始化时发生，初始化是控件生存期的第一步。（继承自 Control 类。）

③ Page.InitComplete 在页初始化完成时发生。

④ Page.PreLoad 在页 Load 事件之前发生。

⑤ Page.Load 当服务器控件加载到 Page 对象中时发生。（继承自 Control 类）。

⑥ Page.LoadComplete 在页生命周期的加载阶段结束时发生 。

⑦ Page.PreRender 在加载 Control 对象之后、呈现之前发生。（继承自 Control 类）。

⑧ Page.PreRenderComplete 在呈现页内容之前发生。

【实例 3-7】 Page 页的生命周期。

```
public partial class _Default : System.Web.UI.Page
{
    protected int i = 1;
    protected void Page_PreInit(object sender, EventArgs e)
    {
        Response.Write(i + ".PreInit: 当页面初始化开始时发生 ");
        i++;            Response.Write("<br>");
    }
    protected void Page_Init(object sender, EventArgs e)
    {
        Response.Write(i + ".Init: 当服务器空间初始化时发生 ");
        i++;            Response.Write("<br>");
    }
    protected void Page_InitComplete(object sender, EventArgs e)
      {
        Response.Write(i + ".InitComplete: 当页初始化完成时发生 ");
        i++;
        Response.Write("<br>");
    }
    protected void Page_PreLoad(object sender, EventArgs e)
    {
        Response.Write(i + ".PreLoad: 当页 load 事件前发生 ");
        i++;
        Response.Write("<br>");
    }
    protected void Page_Load(object sender, EventArgs e)
    {
        Response.Write(i + ".Load: 当服务器控件加载到 Page 对象中时发生 ( 加载
页面时 )");
        i++;
        Response.Write("<br>");                    // 在 Load 和 LoadComplete 之间会
                                                      执行 D 控件 t 事件 t.
        // 如 Button 控件 t 的 Click 事件 t 或 TextBox 控件 t 的 TextChanged 事件
    }
        protected void Page_LoadComplete(object sender, EventArgs e)
        {
            Response.Write(i+".LoadComplete: 当页面生命周期的加载阶段结束时发生 ");
            i++;
            Response.Write("<br>");
        }
    protected void Page_PreRender(object sender, EventArgs e)
    {
        Response.Write(i + ".PreRender: 当页加载控件之后 , 呈现之前发生 , 使用该
事件对页或其控件的内容进行最后更改 ( 预呈现 )");
        i++;
        Response.Write("<br>");
```

```
        }
        protected void Page_PreRenderComplete(object sender, EventArgs e)
        {
            Response.Write(i + ".PreRenderComplete: 呈现内容前发生 .( 预呈现完成 )");
            i++;
            Response.Write("<br>");
        }
        protected void Page_SaveStateComplete(object sender, EventArgs e)
        {   /* 网页控件的状态信息是在 PreRenderComplete 事件后保存的在 SaveStateComplete 事件之前用 Trace 查看页时其实在该事件前还有
             * 一个 SaveState 事件 , 只是这里无法演示
             * 注意 :SaveStateComplete 事件在将页和页上控件的视图状态和控件状态保存到持久性介质之后引发
             * 这是在页被呈现到请求浏览器之前引发的最后一个事件 */
            Response.Write(i + ".SaveStateComplete: 在页已完成对页和页上控件的所有视图状态和控件状态信息的保存后发生 .");
            i++;
            Response.Write("<br>");
            /* 最后一部操作呈现 :Render: 它不是事件；在处理的这个阶段，Page 对象会在每个控件 t 上调用此方法 Render()。 所有 ASP.NET Web 服务器控件都有一个用于写出发送给浏览器的控件标记的 Render 方法。*/
            /* 当你关闭页 3 面的时候就会引发 UnLoad( 当服务器控件从内存中卸载时发生 )
             * 和 Disposed 两事件 ( 当从内存释放服务器控件时发生，这是服务器控件 t 生存期的最后阶段 )
             * 首先加载 UnLoad 事件 : 卸载页面：完全呈现页、将页发送至客户端并准备丢弃时，
             * 将调用卸载。此时，将卸载页属性（如 Response 和 Request）并执行清理
             * 最后加载 Disposed 事件 , 释放资源 , 生命周期结束 ....          */
        }
    }
```

（2）使用 MasterPage 时，MasterPage 与 ContentPage 的事件执行顺序

① ContentPage.PreInit
② MasterPage.Init
③ ContentPage.Init
④ ContentPage.InitComplete
⑤ ContentPage.PreLoad
⑥ ContentPage.Load
⑦ MasterPage.Load
⑧ ContentPage.LoadComplete
⑨ ContentPage.PreRender
⑩ MasterPage.PreRender
⑪ ContentPage.PreRenderComplete

（3）使用继承自 BasePage 的 Page，BasePage 与 Page 的事件执行顺序

对应（1）中的执行顺序，先执行 BasePage 的事件，再执行 Page 的事件。

（4）使用继承自 BasePage 的 Page 作为 MasterPage 的 ContentPage 时事件的执行顺序

对应（2）中的执行顺序，ContentPage 的执行顺序是先 BasePage 后 Page。

项目实训

【实训目的】

掌握使用 ASP.Net 基本控件。

【实训内容】

设计 SelectOkShop 电子商务网站的用户登录、设为主页、联系我们等页面功能。

任务 3.2　使用基本服务器控件设计用户注册页面

基本服务器控件的使用

任务陈述

任务构思与目标：根据需求新建 SelectOKShop 网站的用户注册功能页面，页面效果如图 3-15 所示，并模拟网站后台用户添加功能流程在当前页面（Register.aspx）显示正确添加的用户账户相关信息。

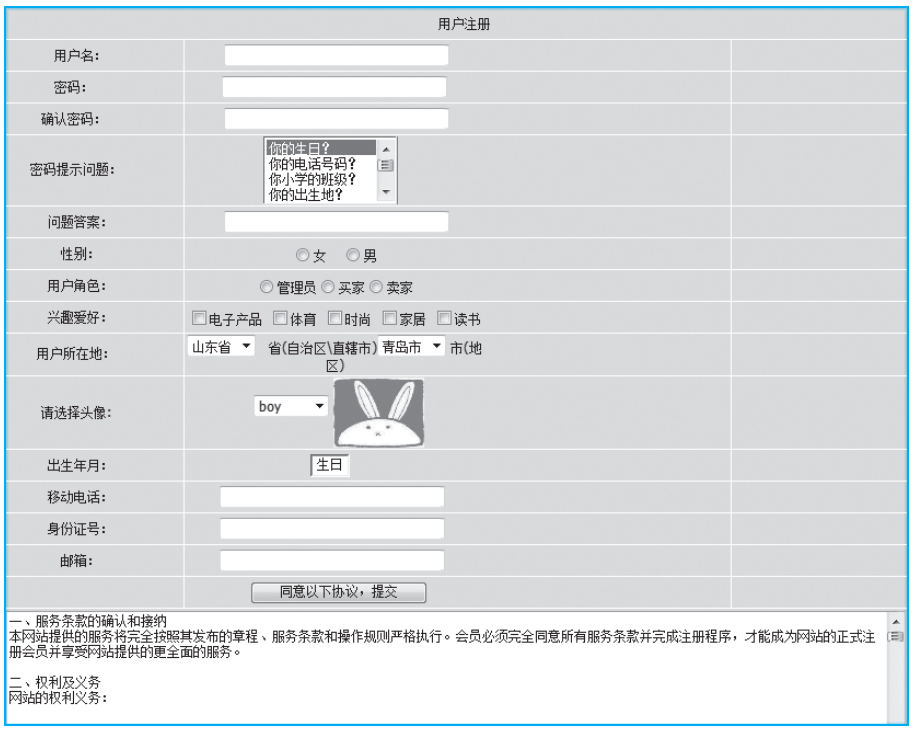


图 3-15　用户注册页面

任务设计：添加 Register.aspx 页面，根据服务器控件功能设计用户注册功能页面，运用页面布局知识实现控件的布局，并在当前页面（Register.aspx）显示正确添加的用户账户相关信息。

知识准备

3.2.1 选择类型控件

1. ListBox 控件

ListBox 控件显示一个选项列表，用户可从中选择一项或多项。如果选项总数超出可以显示的项数，则自动向 ListBox 控件添加滚动条。ListBox 控件的常用属性见表 3-9。

表 3-9 ListBox 控件的常用属性及说明

属　性	说　明
Items	泛指列表框中的所有项，每一项的类型都是 ListItem
Selected	检测条目是否被选中
SelectionMode	组件中条目的选择类型，即单选 (Single)、多选 (Multiple)，当您选择的是多选时，按 Ctrl 键或者 Shift 键可以实现多选
SelectedIndex	列表框中被选择项的索引值
SelectedItem	返回的类型是 ListItem，获得列表框中被选择的条目
SelectedValue	列表框中被选择项的值
Rows	列表框中显示总共多少行
Count	列表框中条目的总数

下面主要介绍 ListBox 控件的 Items 属性及相关方法。

在 ToolBox 中将 ListBox 控件拖放到页面的相应位置，打开属性面板，单击 Items 属性后的[...]按钮，会打开如图 3-16 所示的 ListItem 集合编辑器窗口。

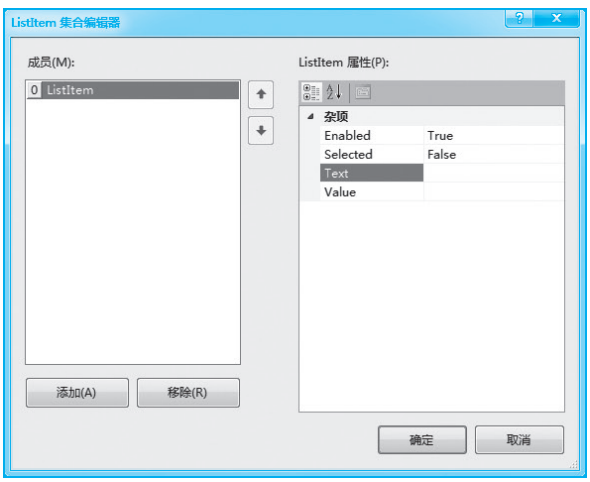


图 3-16 ListItem 集合编辑器窗口

微课 3.2.1 选择类控件 1

微课 3.2.2 选择类控件 2

在“ListItem 集合编辑器”窗口中，通过单击【添加】按钮，为 ListBox 控件添加列表项，可选中项，在属性面板中修改该项的属性值。还可以选中列表项，单击和按钮更改列表项的位置，单击【移除】按钮可以将该项从列表中删除。单击【确定】按钮后，控件中即添加了列表项 ListItem，生成如下代码：

```
<asp:ListBox ID="lstLeft" runat="server" Height="130px" SelectionMode="Multiple">
    <asp:ListItem> 星期一 </asp:ListItem>
    <asp:ListItem> 星期二 </asp:ListItem>
    <asp:ListItem> 星期三 </asp:ListItem>
    <asp:ListItem> 星期四 </asp:ListItem>
</asp:ListBox>
```

也可以在 .aspx 页中直接输入代码，完成添加或在 .cs 文件中添加，下面介绍 ListBox 控件的主要方法和应用。

（1）动态添加列表框中的项

```
ListBox.Items.Add(" 所要添加的项 ");
```

若在服务器端实现 , 为避免每次加载时执行添加列表项 , 上述代码包含在下面代码中。

```
if (!IsPostBack)
    {
      }
```

提示：**Page.IsPostBack** 是用来检查目前网页是否为第一次加载，当使用者第一次浏览这个网页时 **Page.IsPostBack** 会传回 **False**，不是第一次浏览这个网页时就传回 **True**。

（2）移出指定项

```
    If(ListBox.Items.Count > 0 )    // 首先判断列表框中的项是否大于 0
{
        ListBox.Items.Remove(ListBox.SelectedItem); // 移出选择的项
}
```

（3）清空所有项

```
    If(ListBox.Items.Count > 0 ) // 首先判断列表框中的项是否大于 0
{
        ListBox.Items.Clear();// 清空所有项
}
```

（4）指定项移动

```
i. 移至首条：
ListBox.SelectIndex=0; // 将被选中项的索引设置为 0
ii.移至尾条：
```

```
ListBox.SelectIndex=ListBox.Items.Count-1; // 将被选中项的索引设置为 ListBox.
                                              Items.Count-1
iii.          上一条：
ListBox.SelectIndex=ListBox.SelectIndex - 1;        // 用当前被选中的索引去减 1
iv.           下一条：
ListBox.SelectIndex=ListBox.SelectIndex + 1;        // 用当前被选中的索引去加 1
```

（5）插入项

```
ListBox1.Items.Insertat(3,new    ListItem(" 插入在第 3 行之后项 ",""));
ListBox1.Items.Insertat(index,ListItem)
ListBox1.Items.Insert(0,new    ListItem("text","value"));
```

【实例 3-8】 实现列表框单选和多选操作。

运行示例如图 3-17 所示，在左侧 ListBox 控件中选择部分项，通过单击“ > ”按钮后，将会把左侧选择的项移到右侧的列表框中，其他按钮类似，但是在移动后顺序变得混乱，为排序设置了【上移】和【下移】按钮，具体思路是如果是向上移位，就是把当前选定项的上一项的值赋给当前选定的项，然后把刚才新加入的对象的值赋给当前选定项的前一项。

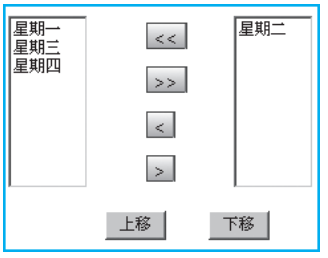


图 3-17 两级联动效果图

Aspx 页的实现代码如下：

```
<Table style="width: 328px">
        <tr><td class="style1" rowspan="4">
         <asp:ListBox ID="lstLeft" runat="server" Height="130px" SelectionMode=
"Multiple">
            <asp:ListItem> 星期一 </asp:ListItem>
            <asp:ListItem> 星期二 </asp:ListItem>
            <asp:ListItem> 星期三 </asp:ListItem>
            <asp:ListItem> 星期四 </asp:ListItem>
            </asp:ListBox>
        </td><td class="style4"><asp:Button ID="btnAllLeft" runat="server" Text="<<"
                CssClass="btn" onclick="btnAllLeft_Click" /></td>
            <td class="style3" rowspan="4">
            <asp:ListBox ID="lstRight" runat="server" Height="130px" SelectionMode=
"Multiple"></asp:ListBox>
            </td></tr>
        <tr><td class="style4"><asp:Button ID="btnAllRight" runat="server" Text=">>"
                CssClass="btn" onclick="btnAllRight_Click" /></td></tr>
        <tr><td class="style4"><asp:Button ID="btnLeft" runat="server" Text="<"
                CssClass="btn" onclick="btnLeft_Click" /></td></tr>
          <tr><td class="style4"><asp:Button ID="btnRight" runat="server" Text=">"
                CssClass="btn" onclick="btnRight_Click" /></td></tr>
```

```
            </Table>
             </br>
            <asp:Button ID="btnUp" runat="server" Text=" 上移 " onclick="btnUp_Click"
/> <asp:Button ID="btnDown"    runat="server" Text=" 下移 " />
```

部分 .cs 文件如下：

```
    protected void btnRight_Click(object sender, EventArgs e)
        {
            try
            {
                lstRight.Items.Add(lstLeft.SelectedItem.Text);// 注意顺序
                lstLeft.Items.Remove(lstLeft.SelectedItem.Text);
            }
            catch (Exception ex)
            {
                Response.Write("<script>alert(' 有异常，请检查！ ')</script>");
            }

        }
        protected void btnAllLeft_Click(object sender, EventArgs e)
        {
            int count = lstRight.Items.Count;
            int index = 0;
            for (int i = 0; i < count; i++)
            {
                ListItem item = lstRight.Items[index];
                lstRight.Items.Remove(item);
                lstLeft.Items.Add(item);
            }
            index++;
        }
        protected void btnLeft_Click(object sender, EventArgs e)
        {
          int count = lstRight.Items.Count;
            int index = 0;
            for (int i = 0; i < count; i++)
            {
                ListItem item = lstRight.Items[index];
                if (lstRight.Items[index].Selected == true)
                {
                    Response.Write("<script>alert(' 触发！ ')</script>");
                    lstRight.Items.Remove(item);
                    lstLeft.Items.Add(item);
                    index--;
                 }
```

```
                index++;
            }
        }
        protected void btnUp_Click(object sender, EventArgs e)
        {
            string name=lstLeft.SelectedItem.Text;
            int index=lstLeft.SelectedIndex;
            lstLeft.SelectedItem.Text=lstLeft.Items[index-1].Text;
            lstLeft.Items[index - 1].Text = name;
    }
```

思考：上例中 lstRight.Items.Add(lstLeft.SelectedItem.Text);lstLeft.Items.Remove(lstLeft.SelectedItem.Text); 两行代码如果调换一下顺序，结果会怎么样，为什么？

2. RadioButton 和 RadioButtonList 控件

RadioButton 控件用于显示单选按钮，该控件的常用属性见表 3-10。

表 3-10 RadioButton 控件的常用属性

属　性	描　述
AutoPostBack	布尔值，规定在 Checked 属性被改变后，是否立即回传表单。默认值是 false
Checked	布尔值，规定是否选定单选按钮
GroupName	该单选按钮所属控件组的名称
OnCheckedChanged	当 Checked 被改变时，被执行的函数的名称
Text	单选按钮旁边的文本
TextAlign	文本水平对齐方式（左对齐还是右对齐）

RadioButton 服务器控件在 Web 窗体页上创建一个单选按钮的意义不大，一般至少需要两个选项。通过设置 Text 属性指定要在控件中显示的文本。该文本可显示在单选按钮的左侧或右侧。设置 TextAlign 属性以控制该文本的水平对齐方式。如果为每个 RadioButton 控件指定了相同的 GroupName，则可以将多个单选按钮分为一组，将单选按钮分为一组将只允许从该组中进行互斥的选择。

【实例 3-9】 RadioButton 控件的应用，实现性别选择功能，如图 3-18 所示。

图 3-18 RadioButton 控件实现性别的选择

控制类控件的使用

页面设计代码如下：

```
    <%-- 使用 GroupName 属性设置组，同一组单选钮互相排斥 --%>
    <asp:RadioButton ID="rbMale" runat="server" GroupName="group1" Text=" 男 " />
    <asp:RadioButton ID="rbFemale" runat="server" GroupName=" group1" Text=" 女 " />
```

操作代码如下：

```
        string gender = "";
            if (rbMale.Checked)
                gender = " 男 ";
```

```
        else if (rbFemale.Checked)
            gender = " 女 ";
```

⚠ 注意：通过使用 GroupName="group1" 属性，定义单选按钮的组，具有相同组名的单选按钮作为一组互斥的控件来使用。若不使用 GroupName="group1" 属性，Web 窗体会将同一容器中的所有的 RadioButton 控件视为不同的组，可以多选。

RadioButtonList 控件用于创建单选按钮组，RadioButtonList 控件中的每个可选项是通过 ListItem 元素来定义的，该控件的常用属性和方法见表 3-11。

表 3-11　RadioButtonList 控件的常用属性和方法

属　性	描　述
CellPadding	单元格边框与内容之间的像素数
CellSpacing	表格单元格之间的像素数
RepeatColumns	用于指定在 RadioButtonList 控件中显示选项占用几列。默认值为 0，表示任意多列
RepeatDirection	用于指定 RadioButtonList 控件的显示方向。Vertical 时，列表项以列优先排列的形式显示；Horizontal 时，列项以行优先排列的形式显示
RepeatLayout	单选按钮组的布局。Table（默认）时，以表结构显示，属性值为 Flow 时，不以表结构显示
TextAlign	文本水平对齐方式（左对齐或右对齐）
Selected	布尔值，规定是否选定单选项
Items	表示列表中各个选项的集合，如 RadioButtonList1.Items(i) 表示第 i 个选项，i 从 0 开始。每个选项都有以下 2 个基本属性： Text 属性：表示每个选项的文本 Value 属性：表示每个选项的选项值
Count	通过 Items.Count 属性可获得 RadioButtonList 控件的选项数
Add 方法	通过 Items.Add 方法可以向 RadioButtonList 控件添加选项
Remove 方法	通过 Items.Remove 方法，可从 RadioButtonList 控件中删除指定的选项
Insert 方法	通过 Items.insert 方法，可将一个新的选项插入到 RadioButtonList 控件中
Clear 方法	通过 Items.clear 方法可以清空 RadioButtonList 控件中的选项
SelectedIndex	用于获取或设置列表中选定项的最低序号索引值。如果列表控件中只有一个选项被选中，则该属性表示当前选定项的索引值
SelectedItem	用于获取列表控件中索引值最小的选定项。如果列表中只有一个选项被选中，则该属性表示当前选定项。通过该属性可获得选定项的 Text 和 Value 属性值
SelectIndexChange 事件	当用户选择了列表中的任意选项时，都将触发 SelectedIndexChange 事件

RadioButtonList 控件使您能够创建单项选择的单选按钮组，可以通过绑定到数据源动态生成该组。若要指定要在 RadioButtonList 控件中显示的项，需要针对每项在 RadioButtonList 控件的开始标记和结束标记之间放置一个 ListItem 元素，使用代码如下所示：

```
<asp:RadioButtonList ID="rdoList" runat="server" RepeatDirection="Horizontal">
        <asp:ListItem Selected> 男 </asp:ListItem>
        <asp:ListItem> 女 </asp:ListItem>
</asp:RadioButtonList>
```

若要确定 RadioButtonList 控件中的选定项，请循环访问 Items 集合并测试该集合中每一项的 Selected 属性，具体实现方式请参考 ListBox 控件。

可以使用 RepeatLayout 和 RepeatDirection 属性指定如何呈现列表。如果将 RepeatLayout 设置为 Table（默认设置），将以表格形式呈现列表。如果将它设置为 Flow，则不会使用任何表格结构呈现列表。默认情况下，RepeatDirection 设置为 Vertical，若将此属性设置为 Horizontal 可以水平呈现该列表。

RadioButton 和 RadioButtonList 这两类控件都有各自的优点。使用单个的 RadioButton 控件相对于使用 RadioButtonList 控件，可以更好地控制单选按钮组的布局。例如，您可以在各单选按钮之间包含非单选按钮文本。如果您想要基于数据源中的数据创建一组单选按钮，则 RadioButtonList 控件是更好的选择。在编写代码以检查所选定的按钮方面，Radio ButtonList 控件也较为简单。

【实例 3-10】 RadioButtonList 控件的应用—实现性别的选择，实现性别选择功能，如图 3-18 所示。

操作代码如下：

```
string gender = "";
    if (rdoList.Items[0].Selected)
        gender = " 男 ";
    if (rdoList.Items[1].Selected)
    gender = " 女 ";
    Response.Write(" 性别 : " + gender);
```

提示：RadioButton 和 RadioButtonList 控件应用区别：使用 RadioButton 控件时，Web 窗体会将同一容器中的所有的 RadioButton 控件视为不同的组，可以多选；通过 GroupName 属性，可以定义单选按钮的组，具有相同组名的单选按钮作为一组互斥的控件来使用。RadioButtonList 是封装了一组单选按钮控件的列表控件，不需要设置其他属性，通过设置 RepeatColumns、RepeatDirection、TextAlign 属性，可以灵活对按钮进行排列。

3. CheckBox 和 CheckBoxList 控件

CheckBox 控件用于显示复选框，用户可以从一组 CheckBox 控件中选择一项或者多项。该控件的常用属性见表 3-12。

表 3-12 CheckBox 控件的常用属性

属 性	描 述
AutoPostBack	规定在 Checked 属性已改变后，是否立即向服务器回传表单。默认是 false
CausesValidation	规定点击 Button 控件时是否执行验证
Checked	规定是否已选中该复选框
InputAttributes	该 CheckBox 控件的 Input 元素所用的属性名和值的集合
LabelAttributes	该 CheckBox 控件的 Label 元素所用的属性名和值的集合
Text	与 CheckBox 关联的文本标签

续表

属　性	描　述
TextAlign	与 CheckBox 控件关联的文本标签的对齐方式
ValidationGroup	在 CheckBox 控件回发到服务器时要进行验证的控件组
OnCheckedChanged	当 Checked 属性被改变时，被执行函数的名称

CheckBoxList 控件用于创建多选的复选框组。每个 CheckBoxList 控件中的可选项都是由 ListItem 元素定义的，具体属性与 RadioButtonList 控件类似，在此不再赘述。

【实例 3-11】　CheckBoxList控件的应用——实现如图 3-19 所示的“多项爱好选择功能”。

图 3-19　CheckBoxList 控件的使用

页面设计代码如下：

```
<asp:CheckBoxList ID="chkHobby" runat="server" RepeatDirection="Horizontal">
                        <asp:ListItem> 音乐 </asp:ListItem>
                        <asp:ListItem> 体育 </asp:ListItem>
                        <asp:ListItem> 美术 </asp:ListItem>
</asp:CheckBoxList>
```

操作代码如下：

```
StringBuilder hobby1 = new StringBuilder();
        foreach (ListItem item in chkHobby.Items)
        {// 判断某项复选框是否被选中
            if (item.Selected)
                hobby.Append(item.Value + ",");
        }
        Response.Write(hobby);
```

4. DropDownList 控件

DropDownList 控件的常用属性、方法及事件与 ListBox 控件类似，但是前者只允许用户每次从列表中选择一项。

【实例 3-12】　列表控件的应用——实现如图 3-20 所示的“学历单项选择功能”。

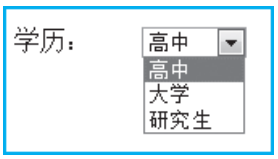


图 3-20　DropDownList 下拉列表框控件的使用

页面设计代码如下：

```
<asp:DropDownList ID="drpXueli" runat="server">
                        <asp:ListItem> 高中 </asp:ListItem>
                        <asp:ListItem> 大学 </asp:ListItem>
                        <asp:ListItem> 研究生 </asp:ListItem>
</asp:DropDownList></td>
```

操作代码如下：

```
string xueli = drpXueli.SelectedItem.Value; // 取出 dropdownlist 控件选中的列表项
```

列表类控件的使用

提示：DropDownList 控件与 ListBox 列表框控件的区别是：ListBox 可以通过设置 SelectionMode 属性值为“Multiple”属性即：SelectionMode="Multiple"，允许用户从后选项中选择多项，而 DropDownList 控件不允许多选。

3.2.2 Image 控件

Image 控件用于显示图像，该控件的常用属性见表 3-13。

表 3-13 Image 控件的常用属性

属 性	描 述
AlternateText	图形的替代文本
DescriptionUrl	对图像进行详细描述的位置
GenerateEmptyAlternateText	规定该控件是否创建一个作为替代文本的空字符串
ImageAlign	规定图像的排列方式，主要的值有：NotSet、AbsBottom、AbsMiddle、BaseLine、Bottom、Left、Middle、Right、TextTop、Top
ImageUrl	要使用的图像的 URL，可以是相对地址或绝对地址

Image 控件的大部分属性和 Label 控件类似，在此主要讲解其 ImageUrl 属性设置。

ImageUrl 属性用于获取 Image 控件中要显示图像的地址，在设置该属性时，单击 ImageUrl 属性文本框后面的...图标按钮，弹出如图 3-21 所示的“选择图像”对话框，用户可以选择要显示的图像。

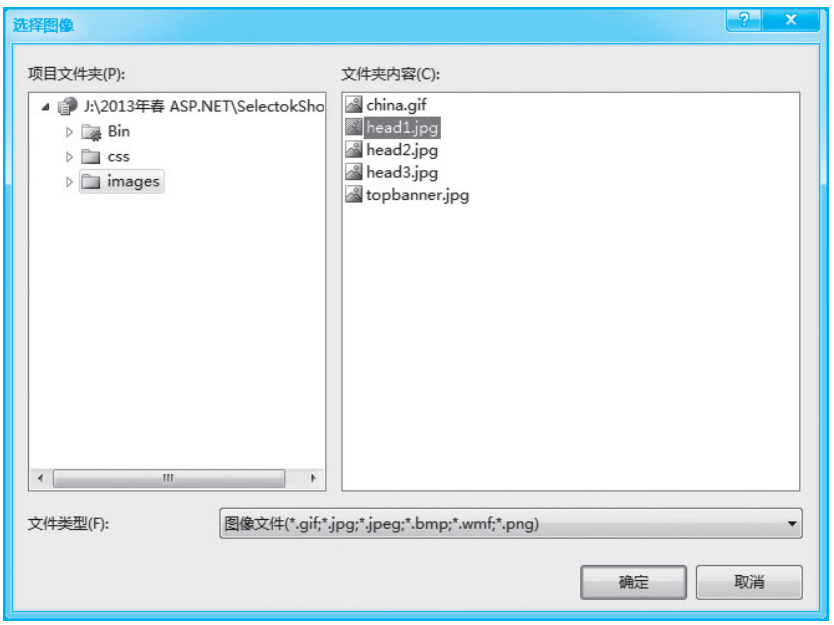


图 3-21 “选择图像”对话框

选择图片后，执行程序，运行结果如图 3-22 所示。

图 3-22　Image 控件示例

3.2.3　Calendar 控件

1. Calendar 控件概述

Calendar 控件用于在浏览器中显示日历，该控件可显示某个月的日历，允许用户选择日期，也可以跳到前一个或下一个月，允许用户选择一定范围内的日期并以编程方式控制选定日期的显示。Calendar 控件控件的常用属性见表 3-14。

表 3-14　Calendar 控件的常用属性及说明

属　性	说　明
Caption	日历的标题
CaptionAlign	日历标题文本的对齐方式
CellPadding	单元格边框与内容之间的空白，以像素计
CellSpacing	单元格之间的空白，以像素计
DayHeaderStyle	显示一周中某天的名称的样式
DayNameFormat	显示周中各天的名称格式
DayStyle	显示日期的样式
FirstDayOfWeek	哪天是周的第一天
NextMonthText	显示下一月链接的文本
NextPrevFormat	下一月和上一月链接的格式
NextPrevStyle	显示下一月和上一月链接的样式
OtherMonthDayStyle	显示不在当前月中的日期的样式
PrevMonthText	显示上一月链接的文本
SelectedDate	选定的日期
SelectedDates	选定的日期（复数）
SelectedDayStyle	选定日期的样式
SelectionMode	允许用户如何选择日期，如果需要让用户可以选择一天、一周或一个月，则必须设置 SelectionMode 属性。属性枚举成员如下： Day：允许用户选择单个日期，为默认值 DayWeek：允许用户选择单个日期或整周 DayWeekMonth：允许用户选择单个日期、周或整个月 None：不能选择日期

续表

属　性	说　明
SelectMonthText	显示为月份选择链接的文本
SelectorStyle	月份和周的选择链接的样式
SelectWeekText	显示为周的选择链接的文本
ShowDayHeader	布尔值，该值指示是否显示一周中各天的标头
ShowGridLines	布尔值，规定是否显示日期之间的网格线
ShowNextPrevMonth	布尔值，规定是否显示下一月和上一月链接
ShowTitle	布尔值，规定是否显示日期的标题
TitleFormat	日期标题的格式
TitleStyle	日期标题的样式
TodayDayStyle	当天的日期的样式
TodaysDate	获取或设置今天的日期的值
UseAccessibleHeader	规定是否使用 <th> 来代替 <td> 元素用于日的头部
VisibleDate	获取或设置指定要在 Calendar 控件上显示的月份的日期
WeekendDayStyle	周末的样式
OnDayRender	当每一天的单元格被创建时，所执行的函数的名称
OnSelectionChanged	当用户选择天、周或月时，所执行的函数的名称
OnVisibleMonthChanged	当用户导航到不同的月时，所执行的函数的名称

2. SelectionChanged 事件

当用户控件选择一天、一周或整月时触发该事件。

【实例 3-13】 日历控件的应用。

```
protected void claDemo_SelectionChanged(object sender, EventArgs e)
    {
        Response.Write(" 您选中的日期是："+ claDemo.SelectedDate.ToShortDateString());
        Response.Write("datetime 的最大值是：" + DateTime.MaxValue);
        Response.Write("datetime 的最大值是：" + DateTime.MinValue);
        lblTodaysDate.Text = "Today's Date is " + claDemo.TodaysDate.Day;
        if (claDemo.SelectedDate != DateTime.MinValue)
            lblSelected.Text = "The date selected is " + claDemo.SelectedDate.ToShortDateString();
            lblCount.Text = "Count of Days Selected:    " + claDemo.SelectedDates.Count.ToString();
    }
```

微课 3.2.3
Calendar 控件

其中，TodaysDate 属性获取或设置今天的日期的值（System.DateTime 类型），读取日期时间后根据具体显示方式或习惯的不同需要对 DateTime 类型值进行转换，DateTime 转换到字符串的形式如下。

① ToFileTime：转换到本地文件系统的格式。
② ToLongDateString：转换到长日期字符串。
③ ToLongTimeString：转换到长时间字符串。
④ ToShortTimeString：转换到短时间字符串。
⑤ ToString：转换到一个字符串。

获取出所选日期内部的具体值需要用到 DateTime 类型的只读属性，主要包括以下属性。

① Date：返回日期部分。
② Day：返回月份中的日期。
③ DayOfWeek：返回一周中的日期，如 Friday，Saturday 等。
④ DayOfYear：返回年份中的日期。
⑤ Hour：返回小时的值。
⑥ Millisecond：返回毫秒的值。
⑦ Minute：返回分钟的值。
⑧ Month：返回月的值。
⑨ Second：返回秒的值。
⑩ Ticks：返回表示日期和时间的以 100 毫微秒为间隔的间隔数。
⑪ TimeOfDay：返回当天的时间。
⑫ Year：返回年的值。
⑬ MaxValue：表示 DateTime 的最大可能值。此字段为只读，值为 12/31/999911:59:59 PM CE。
⑭ MinValue：表示 DateTime 的最小可能值。此字段为只读，值为 1/1/0001 12:00:00 AM。

任务实施

3.2.4　创建 SelectOKShop 用户注册页面

1. 新建 Register.aspx 页面

在网站的解决方案下，右击网站名称，在弹出的快捷菜单中选择【添加新项】命令。打开“添加新项”对话框，选择【Web 窗体页】，命名为“Register.aspx”页面。单击【添加】按钮就可以创建一个新的页面。

2. 实现页面功能

首先，使用“TABLE”布局元素设计良好的页面布局，根据相应功能，在页面上添加 ASP.NET 服务器控件，设置控件的相关属性，页面设计代码如下。

```
<body>
<form id="form2" runat="server">
<ucTitle:TitleUC ID="TitleUC1" runat="server" Title=" 注册新用户 " />
      <table width="100%" height="89%" border="0" cellspacing="0" cellpadding="0">
<tr valign="top">
  <td height="100%" colspan="3" bgcolor="#FFFFFF" class="l-r-space">
    <table width="100%" border="0" cellpadding="1" cellspacing="1" bgcolor="#009DE9">
      <tr>
```

```
                <td bgcolor="#FFFFFF">
                  <table class="Table" cellpadding="2" cellspacing="0" border="0">
                    <tr>
                      <td colspan="2"> 下面是必填内容：</td>
                    </tr>
                    <tr>
                      <td colspan="2"><hr /></td>
                    </tr>
                    <tr>
                      <td width="150" height="30" class="LeftTD" align="right"> 用户名称 :</td>
                      <td valign="middle">
                    <asp:textbox id="tbUsername" runat="server" Width="300px" > </asp:textbox>
                    </td>
                    </tr>
                    <tr>
                      <td width="150" height="30" class="LeftTD" align="right"> 用户密码 :</td>
                      <td valign="middle">
      <asp:textbox id="tbPassword" runat="server" Width="150px"    TextMode="Password">
</asp:textbox>
                      </td>
                    </tr>
                    <tr>
                     <td width="150" height="30" class="LeftTD" align="right"> 确认密码 :</td>
                     <td valign="middle">
      <asp:textbox id="tbPasswordStr" runat="server" Width="150px"    TextMode= "Password">
</asp:textbox>
                     </td>
                    </tr>
                    <tr>
                     <td width="150" height="30" class="LeftTD" align="right"> 电子邮件 :</td>
                     <td valign="middle">
                  <asp:textbox id="tbEmail" runat="server" Width="300px" ></asp:textbox>    </td>
                  </tr>
                  <tr>
                    <td width="150" height="30" class="LeftTD" align="right"> 所属角色 :</td>
                     <td valign="middle"><asp:DropDownList ID="RoleList" runat="server"
Width="153px"></asp:DropDownList>
                    </td>
                  </tr>
                  <tr>
                    <td colspan="2"> 下面是选填内容：</td>
                  </tr>
                  <tr>
```

```
                <td colspan="2"><hr /></td>
              </tr>
              <tr>
                <td width="150" height="30" class="LeftTD" align="right"> 真实名称 :</td>
                <td valign="middle">
                  <asp:textbox id="tbRealname" runat="server" Width="300px" ></asp:textbox>
                </td>
                </tr>
                    <tr>
                    <td width="150" height="30" class="LeftTD" align="right"> 性别 :</td>
                    <td valign="middle">
                      <asp:RadioButtonList ID="rdoList" runat="server" RepeatDirection=
"Horizontal"></asp:RadioButtonList>
                    </td>
                 </tr>
              <tr>
                <td width="150" height="30" class="LeftTD" align="right"> 联系地址 :</td>
                <td valign="middle">
                  <asp:textbox id="tbAddress" runat="server" Width="400px" ></asp:textbox>
                </td>
              </tr>
              <tr>
                <td width="150" height="30" class="LeftTD" align="right"> 联系电话 :</td>
                <td valign="middle">
                  <asp:textbox id="tbPhone" runat="server" Width="300px" >0</asp:textbox>
                </td>
              </tr>
              <tr>
                  <td width="150" height="30" class="LeftTD" align="right"> 移动电话 :</td>
                  <td valign="middle">
                   <asp:textbox id="tbMobile" runat="server" Width="300px" >13</asp:textbox>
                  </td>
               </tr>
              <tr>
                <td width="150" height="30" class="LeftTD" align="right"> 备注信息 :</td>
                <td valign="middle">
                  <asp:textbox id="tbRemark" runat="server" Width="400px"
                                        Height="150px" TextMode="MultiLine"></asp:textbox>
                </td>
              </tr>
               <tr>
                  <td width="150" height="30" valign="middle" class="LeftTD" align="right">
                    <asp:Label ID="Label1" runat="server" Width="150px"></asp:Label></td>
```

```
                    <td valign="middle">
                        <asp:ImageButton ID="ibtAdd" runat="server" ImageUrl="~/App_
Themes/Default/Images/badd.gif" AlternateText=" 添加 " OnClick="ibtAdd_Click" />
                        <asp:ImageButton ID="ibtReturn" runat="server" ImageUrl="~/App_
Themes/Default/Images/breturn.gif" CausesValidation="false" AlternateText=" 返回 "
OnClick="ibtReturn_Click" />
                    </td>
                </tr>
                </table>
            </td>
        </tr>
        </table>
        </td>
        </tr>
      </table>
    </form>
    </body>
```

3. 动态添加列表类控件

添加“角色列表候选项”到下拉列表框控件（RoleList）、RadioButtonList 控件多个候选项，添加“性别”到 RadioButtonList 控件（rdoList）等。程序运行，自动响应页面加载事件“Page_Load”，通过控件的选项集合属性“Items”，动态完成“控件初始化数据”功能。

```
protected void Page_Load(object sender, EventArgs e)
    {
        if (!Page.IsPostBack)
        {
            RoleList.Items.Add(" 超级管理员 ");// 添加角色列表候选项
            RoleList.Items.Add(" 卖家 ");
            RoleList.Items.Add(" 普通客户 ");
            rdoList.Items.Add(" 男 ");// 添加性别到 RadioButtonList 控件
            rdoList.Items.Add(" 女 ");
      }
       }
```

⚠ 注意：**Page.IsPostBack** 用于检查目前网页是否为第一次加载，当使用者第一次浏览这个网页时 **Page.IsPostBack** 会返回 **False**，不是第一次浏览这个网页时则返回 **True**。

4. 问题回答的判断

```
protected void btnOK_Click(object sender, EventArgs e)
    {
        if (this.IsValid) // 判断页面验证是否全部通过
        {
```

```
                Response.Write( "<Script>alert(' 页面验证成功 ')</script>");
                // 提交到服务器的后续操作
            }
            else
            {
                Response.Write("<Script>alert(' 页面验证不成功 ')</script>"); // 取消提交到
服务器的操作
            }
            // 验证通过后插入到数据库
            //if (txtUserName.Text!=""&&txtPwd.Text==txtPwdOK.Text)
            //{
            //      Response.Write("<script>alert(' 提交成功 !')</script>");
            //}
            //string str = lstPwdTip.SelectedItem.Text;
            //string strAnswer = txtAnswer.Text;
            ////Response.Write("<script>alert(' 你选的 '+str+' 问题答案是：'+strAnswer)</script>");
            //Response.Write(" 你选的 "+str+" 问题答案是："+strAnswer);
        }
        protected void lstPwdTip_SelectedIndexChanged(object sender, EventArgs e)
        {
            Response.Write(lstPwdTip.SelectedItem);
            Response.Write(lstPwdTip.SelectedIndex);
            Response.Write(lstPwdTip.SelectedValue);
        }
```

5. 头像选择

```
    protected void ddlImage_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (ddlImage.SelectedIndex==0)
            {
                imgHead.ImageUrl="~/images/head1.jpg";
            }
            else if (ddlImage.SelectedIndex==1)
            {
                imgHead.ImageUrl = "~/images/head2.jpg";
            }
            else
            {
                imgHead.ImageUrl = "~/images/head3.jpg";
            }
        }
```

6. 出生日期的选择

```
    protected void btnBirthday_Click(object sender, EventArgs e)
        {
            calBirthday.Visible = true;
        }
```

```
protected void calBirthday_SelectionChanged(object sender, EventArgs e)
{
    btnBirthday.Text = calBirthday.SelectedDate.ToShortDateString();
    calBirthday.Visible = false;
}
```

7. 任务运行

运行 Register.aspx 页面如图 3-15 所示。

任务拓展

3.2.5 ASP.NET 路径的使用

通常，路径的表达有三种方式：绝对路径、相对路径和基于根目录的路径。例如，假设当前应用程序的结构如图 3-23 所示，其网址为“http://www.asp.net”，下面介绍路径的表示方法。

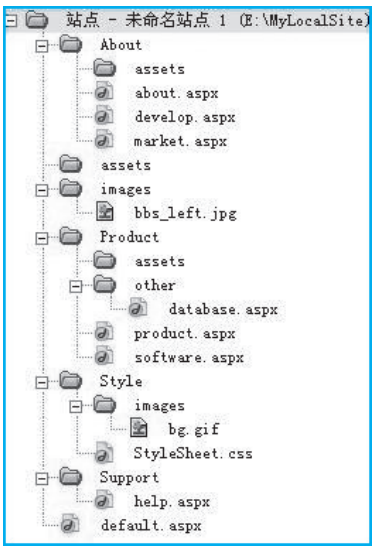


图 3-23 MyLocalSite 网站目录结构图

1. 绝对路径

使用完整的 URL 地址，这种链接路径就是绝对路径，其特点是路径同链接站点的源点无关。

在图 3-23 中，如要建立指向 about 目录下的 about.aspx，则链接地址为：

```
< a href="http://www.asp.net/about/about.aspx"></a>
```

建立指向 database.aspx 文档的路径为：

```
< a href="http://www.asp.net/product/other/database.aspx"></a>
```

⚠ 注意：使用绝对路径的最大缺点是不利于移植，例如，现在站点的地址为“**http://www.asp.net**”，如果有一天站点更改为“**http://www.asp.com**”，则所有的链接地址都失效，需要进行相应的更改，难以维护。

2. 相对路径

为了避免绝对路径的缺陷，可以使用相对路径。相对路径可以表达源端点和目标端点之间的相互位置关系。具体可以范围如下两种情况：

如果链接中源端点和目标端点在同一个目录下，则在链接中只需要指明目标端点的文档名称即可。例如，在图 3-23 中，如果希望在 about.aspx 文档中创建指向 develop.aspx 的链接，则可以编写代码如下：

```
<a href="develop.aspx">
```

如果在链接中，源端点和目标端点不位于同一个目录下，则只需要将目录

的相对关系表达出来即可。如果链接指向的文档没有位于当前目录的子级目录中，则可以利用“..”符号来表示当前的父目录，多个“..”符号可以表示更高的父级目录，从而构建出目录的相对位置。例如，如果希望在 about.aspx 文档中创建指向位于 product 目录中的 software.aspx 文档的链接，则可以用如下的代码：

```
<a href="../product/software.aspx">
```

如果希望在 database.aspx 文档中创建指向 about.aspx 文档的链接，则可以使用如下的路径：

```
<a href="../../about/about.aspx">
```

⚠ 注意：利用相对目录的好处在于只要站点的结构和文档不变，链接就不会出错，然而如果你移动了文件，或者相对关系发生变化，则就会发生错误。

3. 基于根目录的路径

基于根目录的路径可以看成是绝对路径和相对路径之间的一种折中，在这种表达方式下，所有的路径都是从站点的根目录开始的，同源端点位置无关。

例如，在图 3-23 中，建立指向 About 文件夹下的 about.aspx 的链接为：

```
<a href="/about/about.aspx">
```

建立指向 database.aspx 的链接为：

```
<a href="/product/other/database.aspx">
```

如果指向的是主页则可以写为：

```
<a href="/"> 或者 <a href="/default.aspx">
```

⚠ 注意：判断一个路径是相对路径还是相对于根目录的路径就看链接地址是否是从 "/" 开始。

4. ASP.NET 的“～”路径

在 ASP.NET 里增加了一个新的路径表达方法“～”，“～”表示的路径是当前应用程序的根目录。“～”和上面介绍的“/”最大的区别是由服务器进行动态解释。由于“~”是相对于应用程序的根目录，所以可以简化路径的设置。

⚠ 注意：asp.net 中有一些地方是不支持相对路径的，如 web.config；“~”只可以在服务器端使用。

5. 在样式中使用地址应注意的问题

可以在样式中引用图像，该图像的引用使用相对于样式文件的引用，仍然以图 3-23 为例，我在 Style 下建立了一个 StyleSheet.css 文件用于存放样式，该样式的内容如下：

```
td.mybg{background:url(images/bg.gif)}
```

其中，将 td 元素重新定义增加了背景图片“bg.gif”。“bg.gif”文件来自于 Style/images 文件夹下的“bg.gif”，这样在首页引用时，可以正确的显示，代码如下：

```
<link rel="stylesheet" type="text/css" href="Style/StyleSheet.css">
<table height="86">
<tr>
<td width="146" class="mybg">this is a test</td>
</tr>
</table>
```

然而这与主题中使用图像时正好相反，主题还可以包含外观文件中的控件定义所引用的图像。外观中对图像的引用应使用 Theme 目录下的图像文件夹的相对路径，以便外观文件和图像可以一起移动到其他应用程序中。在运行时，会重新设置图像的路径，以引用相对于目标页中的控件，而不是相对于外观文件。

6. 确保路径指向文件的唯一性

在用 ASP.NET 处理路径时，需要确保文件的唯一性，否则将发生错误。特别是在使用用户控件时，如果路径引用的过于复杂，尽管理论上应该正确，但是实际运行时可能出现预想不到的错误。

例如，在图 3-24 中，在 UI 的目录学有一个 Control 文件夹，在该文件夹下有一个 Calendar.ascx 用户控件，Page.ascx 用户控件引用了该 calendar.ascx 的代码如下：

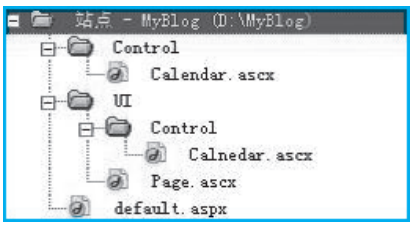


图 3-24 MyBlog 结构图

```
<%@ Register TagPrefix="uc1" TagName="Calendar" Src="Control/Calendar.ascx" %>
... ...
  <uc1:Calendar id="Calendar1" runat="server"></uc1:Calendar>
```

在根目录下有一个 default.aspx 首页，其又引用了 Page.ascx 用户控件，代码如下：

```
<%@ Register TagPrefix="uc1" TagName="Page" Src="UI/Page.ascx" %>
... ...
  <uc1:Page id="Page1" runat="server"></uc1:Page>
```

同时根目录下也有一个 Control 文件夹和 Calendar.ascx 文件。尽管上面代码表面上看没有问题，但是由于用户控件层层嵌套引用，使得实际运行时，系统区分不了到底使用哪一个 Calendar.ascx 而发生错误，此时一个较为简单的解决方法是，所有的路径都使用相对于根目录的引用。

3.2.6 ImageMap 控件

ImageMap 控件可以创建一个图像，使其包含许多可由用户单击的区域，这些区域称为“热点”。每一个热点都可以是一个单独的超链接或回发事件。

微课 3.2.4 图像类控件

将【ImageMap】控件从【工具箱】任务窗格拖放到网页上，设置 ImageMap 的外观和行为的属性。ImageMap 属性继承自 Image 控件，并添加了许多属性和 Click 事件，使该类有了图像映射的功能。ImageMap 控件的常用属性见表 3-15。

表 3-15　ImageMap 控件的常用属性

属　性	描　述
NavigateUrl	指定的 Url 上，PostBack 表示会产生一个服务器回发操作
HotSpots	ImageMap 控件中包含的所有热点对象的集合。有三种类型的热点区域： RectangleHotSpot：定义一个图片的矩形区域，包括有 Top、Bottom、Left 和 Right 属性，表示相对于图片的左上角以像素为单位取值 CircleHotSpot：包括指定圆心的 X、Y 属性，以及一个指定半径的 Radius 属性，都是以像素为单位 PolygonHotSpot：定义了一个多边形区域，由该区域轮廓线段端点的 X、Y 坐标组成的列表，端点坐标间以逗号隔开
AlternateText	图片无效时显示
HotSpotMode	热点模式，对应枚举类型 System.Web.UI.WebControls.HotSpotMode。其选项及说明如下： NotSet：未设置项。虽然名为未设置，但其实默认情况下会执行定向操作，定向到指定的 URL 位址去。如果未指定 URL 位址，那默认将定向到自己的 Web 应用程序根目录 Navigate：定向操作项。定向到指定的 URL 地址去。如果未指定 URL 地址，那默认将定向到自己的 Web 应用程序根目录 PostBack：回发操作项。点击热点区域后，将执行后部的 Click 事件 Inactive：无任何操作，即此时形同一张没有热点区域的普通图片
NavigateUrl	指定该热点的 Url 地址
PostBackValue	被点击对象的值，由事件参数 ImageMapEventArgs 传递
Target	指定浏览器要目标页显示在哪种类型的窗口中。有以下几个值： _blank：将内容显示在一个无框架的新建的未命名窗口中； _new：未列入标准文档，类似 _blank； _parent：将内容显示在父级窗口或框架中或者显示在该链接指向的框架页中； _self：在当前框架或窗口中显示内容； _top：在当前没有框架的整个窗口中显示页面内容

ImageMap 控件主要由两部分组成。第一个是图像，图像可以是任何标准 Web 图形格式的图形，例如 .gif、.jpg 或 .png 文件。第二个元素是一个热点控件集。每个热点控件都是一个不同的元素。对于每个热点控件，都需要定义其形状（圆形、矩形或多边形），还要定义用于指定热点位置和大小的坐标。例如，如果创建了一个圆形热点，则应定义圆心的 x 和 y 坐标以及圆的半径，可以根据需要为图像定义任意数量的热点，但不需要定义足以覆盖整个图形的热点。

⚠ 注意：可以定义重叠的作用点。每个热点都包含一个 z- 索引值，如果用户单击由两个或更多重叠热点定义的区域，将选中 z 顺序值最高的热点。

在【设计】视图中，右键单击 ImageMap 控件，选择快捷菜单中的【属

性】，单击 HotSpots 属性旁的省略号按钮，打开【HotSpot 集合编辑器】对话框。单击【添加】按钮右边的箭头，再单击要添加的热点类型：CircleHotSpot、RectangleHotSpot 或 PolygonHotSpo 在【属性】区域中，设置热点的属性。

【实例 3-14】 RectangleHotSpot 的应用。

```
<asp:ImageMap ID="ImageMap1" runat="server"
        HotSpotMode="PostBack" ImageUrl="~/Image/yesnomaybe.gif"  OnClick=
"firstImage_Click">
          <asp:RectangleHotSpot Bottom="60" Top="21" Left="17" Right="103"
           PostBackValue="Yes" AlternateText="Yes" />
           <asp:RectangleHotSpot Bottom="60" Top="21" Left="122" Right="208"
           PostBackValue="No" AlternateText="No" />
           <asp:RectangleHotSpot Bottom="122" Top="83" Left="17" Right="103"
           PostBackValue="Maybe" AlternateText="Maybe" />
           <asp:CircleHotSpot HotSpotMode="Navigate" X="165" Y="106"
           Radius="25" NavigateUrl="http://www.baidu.com" Target="_blank"
AlternateText=" 百度一下 " />
    </asp:ImageMap>
```

3.2.7 Calendar 控件的高级应用

1. Calendar 控件的高级属性与方法

使用 Calendar 控件的 VisibleDate 属性可以设置显示的月份，VisibleDate 属性是 DateTime 类型需要 3 个整型参数：year、month 和 day。例如：

```
Calendar1.VisibleDate = new DateTime(Calendar1.VisibleDate.Year,
                        Int32.Parse(ddl.SelectedItem.Value), 1);
```

其中，Int32.Parse(ddl.SelectedItem.Value) 能把 Value 值转换为整型

DateTime 对象的 DaysInMonth 属性获取该月份中的天数，例如：

```
System.DateTime.DaysInMonth(currentYear, currentMonth)
```

判断当前日期是星期几，可用 DayOfWeek 对象，例如：

```
DateTime date = new DateTime(currentYear, currentMonth, i);
if (date.DayOfWeek == DayOfWeek.Friday)
   //date 是星期五
```

使用 SelectedDates 集合以编程方式选择 Calendar 控件上的日期。使用 Add、Remove、Clear 和 SelectRange 方法在 SelectedDates 集合中的选定日期。例如：

```
Calendar1.SelectedDates.Add(date);
```

SelectRange 方法将指定的日期范围添加到 SelectedDatesCollection 集合中需要两个参数：开始日期和结束日期。例如：

```
Calendar1.SelectedDates.SelectRange(StartDate, EndDate);
```

2. DayRender 事件

当为 Calendar 控件在控件层次结构中创建每一天时发生，DayRender 事件处理程序接收两个 DayRenderEventArgs 类型的参数。该对象有两个属性，可以通过编程方式读取。

① Cell 表示要呈现的单元格的表格单元格对象。

② Day 表示呈现在单元格中日期的 CalendarDay 对象。

例如，添加事件：

```
OnDayRender="Calendar1_DayRender"
```

事件处理程序：

```
protected void Calendar1_DayRender(object sender, DayRenderEventArgs e)
{
    //  注意这将会覆盖 WeekendDayStyle
    if (!e.Day.IsOtherMonth && e.Day.IsWeekend)
      e.Cell.BackColor = System.Drawing.Color.LightGreen;
    //  在单元格中显示 "Happy New Year!"
    if (e.Day.Date.Month == 1 && e.Day.Date.Day == 1)
      e.Cell.Controls.Add(new LiteralControl("<br/>Happy New Year!"));
}
```

其中参数表示意义如下：

DayRenderEventArgs 包含 Day 和 Cell 的属性。

Day 是一个 CalendarDay 类型属性。CalendarDay 类的属性主要有：

① Date：由 Day 表示的日期，只读。

② DayNumberText：该日期的日编号的等效字符串，只读。

③ IsOtherMonth：指示该日期是否显示当前月份以外的月份，只读。

④ IsSelectable：指示该日期是否可以被选择，非只读。

⑤ IsSelected：指示该日期是否被选择。

⑥ IsToday：指示该日期是否是今天。

⑦ IsWeekend：指示该日期是否是周末。

3. VisibleMonthChanged 事件

当用户单击标题标头上的下个月或上个月导航控件时发生，添加事件：

```
OnVisibleMonthChanged="Calendar1_VisibleMonthChanged"
```

事件处理程序代码如下：

```
protected void Calendar1_VisibleMonthChanged(object sender, MonthChangedEventArgs e)
{
    if ((e.NewDate.Year > e.PreviousDate.Year) || ((e.NewDate.Year == e.PreviousDate.Year) &&
```

```
            (e.NewDate.Month > e.PreviousDate.Month)))
            lblMonthChanged.Text = "My future's so bright...";
        else
            lblMonthChanged.Text = "Back to the future!";
        Calendar1.SelectedDates.Clear(    );
        lblSelectedUpdate(    );
        lblCountUpdate(    );
        txtClear(    );
    }
```

其中，MonthChangedEventArgs 类型的参数属性如下。

① NewDate：表示 Calendar 当前显示的月份。

② PreviousDate：表示 Calendar 以前显示的月份。

3.2.8 使用面向对象思想模拟操作新注册的用户

面向对象是一种对现实世界理解和抽象的方法，是计算机编程技术发展到一定阶段后的产物。早期的计算机编程是基于面向过程的方法，例如，实现算术运算 1+1+2 = 4，通过设计一个算法就可以解决当时的问题。随着计算机技术的不断提高，计算机被用于解决越来越复杂的问题。通过面向对象的方式，将现实世界的事物抽象成对象，现实世界中的关系抽象成类、继承，帮助人们实现对现实世界的抽象与数字建模。通过面向对象的方法，以更易于理解的方式对于复杂系统进行分析、设计与编程。同时，面向对象能有效提高编程的效率，通过封装技术，消息机制可以像搭积木的一样快速开发出新的系统。C# 语言是一种典型的面向对象的语言，在 Web 开发中面向对象编程有助于提高代码的重用性，对于提高开发效率，降低开发成本意义重大。实现代码如下。

```
public partial class Admin_User_AddUser : System.Web.UI.Page
    {
        protected void ibtAdd_Click(object sender, ImageClickEventArgs e)
        {
            // 得到性别
            string gender = "";
            if (rdoList.Items[0].Selected)
                gender = " 男 ";
            if (rdoList.Items[1].Selected)
                gender = " 女 ";
            // 创建一个用户对象
            User user = new User(tbUsername.Text, tbPassword.Text, tbEmail.Text,
drpRoleList.SelectedValue.ToString(), tbRealname.Text, tbAddress.Text,
            tbPhone.Text, gender, tbRemark.Text);
            // 创建用户列表
```

```
                List<User> listUser = new List<User>();
                listUser.Add(user);
                Response.Write(" 用户信息如下所示 " + "\n");
                foreach (User usera in listUser)
                {
                    Response.Write(usera.UserName + "\n");
                    Response.Write(usera.Role + "\n");
                }
        }
    public class User// 定义用户类
    {
        string userName;    // 定义属性
          public string UserName
        {
            get { return userName; }
            set { userName = value; }
        }
        string     password;
        string     email ;
        string role;
          public string Role // 定义属性
          {
              get { return role; }
              set { role = value; }
          }
        string realName;
        string address ;
        string     phone ;
        string gender; // 性别
        string remark;
        public User(string userName, string password, string email, string role, string
realName, string address, string phone, string gender, string remark) // 构造函数
            {
                this.userName = userName;
                this.password = password;
                this.realName = realName;
                this.address = address;
                this.phone = phone;
                this.email = email;
                this.role= role;
                this.gender = gender;
                this.remark = remark;
            }
        }
```

提示：在 **SelectOKShop** 电子商务网站中，新注册的用户信息应该使用后边章节要讲到的数据库访问技术：**ADO.NET** 添加到后台数据中，本任务模拟其流程在当前页面输出成功注册的用户信息。

项目实训

【实训目的】

掌握使用 ASP.Net 基本控件的使用

【实训内容】

设计 SelectOk 电子商务网站用户服务功能的各个页面：用户登录、用户注册、用户查找等。

任务 3.3 使用 ASP.NET 验证控件检验用户注册信息

验证控件的使用

任务陈述

任务构思与目标：利用验证控件来验证新注册用户数据的有效性。

根据需求，在“任务 3-2 使用基本服务器控件设计的用户注册页面（Register.aspx）的基础上，需要对用户在控件中输入的数据进行有效性验证，以保证用户提交的数据是合法的。

在用户注册页 Register.aspx 上，根据功能添加相应的验证控件，设置验证控件的相关验证属性，结果如图 3-25 所示。

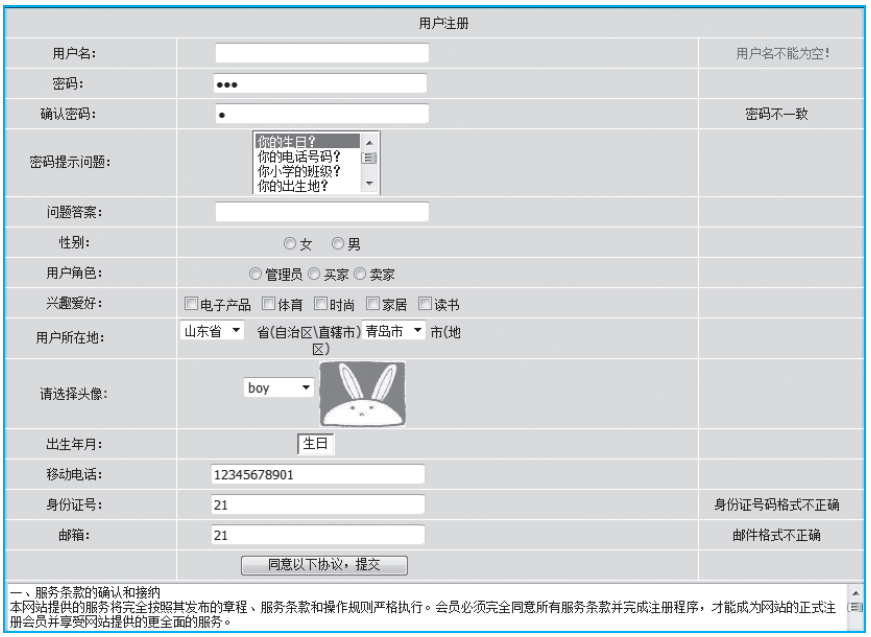


图 3-25 验证控件验证用户注册信息的有效性

任务设计：在“任务 3-2 用户注册页面（Registe.aspx）”的设计基础上，按照验证功能分类给 asp.net 控件绑定对应的验证控件，从而验证 asp.net 控件数据的有效性，若数据验证不合法，则在当前页面动态给出“验证错误信息”，如图 3-25 所示。

知识准备

验证控件的使用

PPT

3.3.1　验证控件

Web 应用程序开发者，特别是 ASP 应用程序开发者，一直对数据验证比较恼火，好不容易写出数据提交程序的主体以后，还得花大量时间去验证用户的每一个输入是否合法。如果开发者熟悉 JavaScript 或者 VBScript，可以用这些脚本语言实现验证，但是又要考虑用户浏览器是否支持这些脚本语言。通过 ASP.NET 验证控件可以轻松的实现对用户输入的验证，程序员们可以将重要精力放在主程序的设计上了。

在 ASP.NET 中，提供了 5 种基本的验证类型控件和一个验证总结控件（ValidationSummary），分别由不同的验证控件来实现。每个验证控件都引用页面上其他的输入控件（服务器控件）。在处理用户输入时，ASP.NET 页框架将用户输入传递到一个或多个适当的验证控件。验证控件将测试用户输入并设置表示输入是否通过测试的属性。在调用所有验证控件之后，该页面的 IsValid 属性被设置无效，如果任何一个控件显示验证检查失败，则该 IsValid 属性设置为无效。5 种基本的验证类型控件和验证总结控件见表 3-16。

表 3-16　验证服务器控件

验证服务器控件名称	说　明
RequiredFieldValidator	判定用户是否在某个输入框中输入了数据
CompareValidator	将用户输入的数据与指定的数据进行比较
RangeValidator	判定用户所输入的数据是否在某个规定的范围内
RegularExpressionValidator	判定用户所输入的数据是否符合某种规定的格式
CustomValidator	用于自定义验证规则
ValidationSummary	给出网页上所有 server 控件的错误信息 Validation

表 3-17 中列出的属性适用于所有验证控件。

表 3-17　基本验证控件的共用属性

属　性	说　明
ControlToValidate	获取或设置要验证的输入控件
Display	指定的验证控件的显示行为，此属性可以为下列值之一：None—验证控件从不内联显示，如果希望仅在 ValidationSummary 控件中显示错误信息，则使用此选项。Static—如果验证失败，验证控件显示错误信息，当验证控件显示其错误信息时，页面布局不变。Dynamic—如果验证失败，验证控件显示错误信息，当验证失败时，在页上动态分配错误信息的空间

续表

属　性	说　明
EnableClientScript	指示是否启用客户端验证。通过将 EnableClientScript 属性设置为 False，可在支持此功能的浏览器上禁用客户端验证
Enabled	指示是否启用验证控件。可通过将该属性设置为 False 以阻止验证控件验证输入控件
ErrorMessage	当验证失败时在 ValidationSummary 控件中显示的错误信息
ForeColor	指定当验证失败时用于显示内联消息的颜色
IsValid	指示 ControlToValidate 属性所指定的输入控件是否被确定为有效
Text	此属性设置后，验证失败时会在验证控件中显示此消息。如果未设置此属性，则在控件中显示 ErrorMessage 属性中指定的文本

3.3.2 客户端验证

在 ASP.NET 验证控件技术出现之前，控件输入数据后，需要开发者自己编写代码实现验证功能，如果开发者熟悉 JavaScript 或者 VBScript，可以用脚本语言在客户端轻松实现验证，同时减轻服务器的负担。

【实例 3-15】 JavaScript 脚本实现客户端验证。

如图 3-26 所示，在输入用户信息时，若输入的姓名为空或未选择性别，则提示错误信息，并取消“提交”功能。

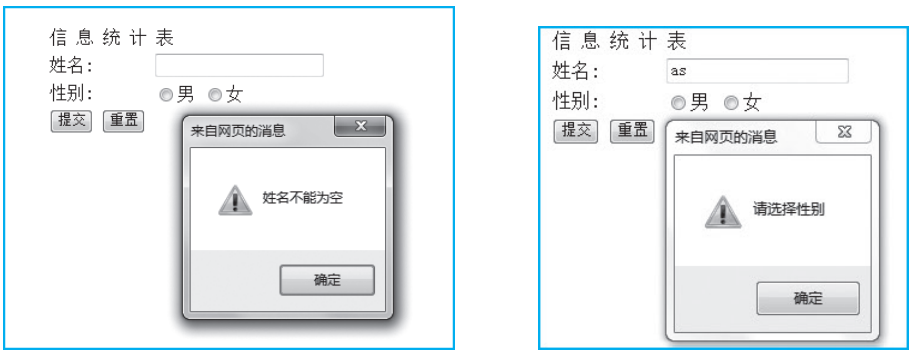


图 3-26 客户端验证

验证代码如下。

```
<html>
<head>
<link rel="stylesheet" type="text/css">
<title>javascript 示例 </title>
<script language="javascript">
    function check() {<!-- 提交表单时调用验证函数 -->
        var name = document.f1.name.value; // 验证姓名不能为空
        var age = document.f1.age.value;
        if (name == "") {
            alert(" 姓名不能为空 ");
            document.f1.name.focus();
            return false;
        }
         var gender = new Array();// 验证必须选择性别
```

```
            gender = document.f1.sex;
            var gendersum = 0;
            for (var i = 0; i < gender.length; i++) {
                if (gender[i].checked) {
                    gendersum++;
                }
            }
            if (gendersum == 0) {
                alert(" 请选择性别 ");
                return false;
            }
        }
    </script>
    </head>
    <body>

    <form action="" name="f1" method="get" onSubmit="return check()">
    <table   width="34%"    class="default" align="center">
    <tr class="title">        <td colspan="2"> 信   息     表 </tr></tr>
    <tr>
     <td width="21%" class="item"> 姓名 </td>
     <td width="79%"><input type="text" name="name" size="20"></td>
    </tr>
    <tr>
     <td    class="item"> 性别 </td>
     <td><input type="radio" name="sex" value="1" > 男
           <input type="radio" name="sex" value="0"> 女            </td>
    </tr>
    <tr>
     <td colspan="2"><input type="submit" name="sm" value=" 提交 " class="btn">
         <input type="reset" name="rs" value=" 重置 " class="btn"></td>
     </tr>
    </table>
    </form>
    </body>
    </html>
```

微课 3.3.1　客户端验证

微课 3.3.2　验证控件 1

微课 3.3.3　验证控件 2

微课 3.3.4　验证控件 3

3.3.3　ASP.NET 验证控件

1. RequiredFieldValidator 限制空字段控件

RequiredFieldValidator 控件用于在 Web 窗体页面上保证用户非空输入，RequiredFieldValidator 控件的常用属性见表 3-4。在页面提交前不允许用户名为空，否则显示“错误提示信息”，代码示例如下：

```
    <asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server" ControlToValidate=
"UserNameTxt" ErrorMessage="* 必须填写用户名 " Display="Static" ForeColor="Red"></
asp:RequiredFieldValidator>
```

⚠ 注意：验证控件只能验证输入类控件。

2. CompareValidator 比较两个字段控件

可使用 CompareValidator 控件与固定值比较，也可对两个控件进行比较，还可以用于检查数据类型。CompareValidator 控件的常用属性见表 3-18。

表 3-18 CompareValidator 控件的常用属性

属 性	属 性 值
Operator	获取或者设置严重使用的比较操作，如 Equal 等
ControlToValidate	表示要进行验证的控件的 ID
ControlToCompare	获取或者设置用于比较的输入控件的 ID
Type	获取或者设置比较的两个值数据类型
ValueToCompare	获取或者设置要比较的值

【实例 3-16】 JavaScript 脚本实现客户端验证

检查文本框中输入的日期是否符合要求，运行效果如图 3-27 所示。

```
<asp:TextBox ID="txtDate" runat="server"></asp:TextBox>
  <asp:CompareValidator ID="CompareValidator1" runat="server"
      ControlToValidate="txtDate" Display="Dynamic" ErrorMessage=" 日期格式有问题 "
      Operator="DataTypeCheck" Type="Date"></asp:CompareValidator>
<asp:Button ID="btn" runat="server" Text=" 提交 " />
```

【实例 3-17】 使用 CompareValidator 控件对两个控件的输入进行比较，下边对“密码”“确认密码”是否一致进行比较，运行效果如图 3-28 所示。

图 3-27 日期验证

图 3-28 比较字段检查

添加 ASP.NET 控件，设计界面如图 3-28 所示，从工具箱拖放 CompareValidator 控件，并设置其相关属性，见表 3-19。

表 3-19 CompareValidator 控件的属性

属 性	属 性 值
Errormessage	密码不一致！
Display	Static
Operator	Equal
ControlToValidate	表示要进行验证的控件的 ID。ConfirmPassword
ControlToCompare	获取或者设置用于比较的输入控件的 ID。PassWordTxt
Type	String

页面主要设计代码如下：

```
    <asp:TextBox ID="PassWordTxt" runat="server" TextMode="Password" Width="149px"> </
asp:TextBox>
    <asp:TextBox ID="ConfirmPassword" runat="server" TextMode="Password"
Width="149px"></asp:TextBox>
    <asp:CompareValidator ID="CompareValidator2" runat="server" ControlToCompare=
"PassWordTxt"
        ControlToValidate="ConfirmPassword" ErrorMessage=" 密码不一致 !"
        Width="166px" ForeColor="Red">
    </asp:CompareValidator>
```

3. RegularExpressionValidator 表达式验证控件

RegularExpressionValidator 验证控件用来验证输入控件的值，以确定该值是否与某个正则表达式所定义的模式相匹配，ValidationExpression 属性用来指定用于验证输入控件的正则表达式。正则表达式是一种文本模式，包括普通字符（例如，a 到 z 之间的字母）和特殊字符（称为“元字符”），该文本模式描述在搜索文本时要匹配的一个或多个字符串。正则表达式的结构与算术表达式的结构类似，即各种元字符和运算符的组合，常用正则表达式字符如下：

（1）匹配数字与非数字

\d：任何一个数字，等价于 [0-9] 或 [0123456789]。

\D：任何一个非数字，等价于 [^0-9] 或 [^0123456789]。

（2）匹配字母和数字与非字母和数字

字母（A-Z 不区分大小写）、数字、下划线是一种常用的字符集合，可用如下类元字符：

\w：任何一个字母（不区分大小写）、数字、下划线，等价于 [0-9a-zA-Z_]。

\W：任何一个非字母数字和下划线，等价于 [^0-9a-zA-Z_]。

（3）匹配空白字符与非空白字符

\s：任何一下空白字符，等价于 [\f\n\r\t\v]。

\S：任何一下空白字符，等价于 [^\f\n\r\t\v]。

正则表达式字符的组合即组成了正则表达式，常用的正则表达式见表 3-20。

表 3-20　常用的正则表达式

表　达　式	匹　　配
验证用户密码 : 以字母开头，长度在 6-18 之间，只能包含字符	^[a-zA-Z]\w{5,17}$
只能输入由数字、26 个英文字母或者下划线组成的字符串	^\w+$
验证身份证号（15 位或 18 位数字，最后一位或者为 x）	^(^\d{15}$|^\d{18}$|^\d{17}(\d|X|x))$
账号（字母开头，允许 6-20 字符，允许字母数字下划线）	^[a-zA-Z][a-zA-Z0-9_]{5,20}$
只能输入 m-n 位的数字	^\d{m,n}$

可以通过属性窗口设置正则表达式，如图 3-29 所示，单击...，可打开 ASP.NET 正则表达式编辑器，如图 3-30 所示，选择表达式类型，实现相应的验证。

关于复杂的正则表达式，可以查阅附录，并且可以根据正则表达式的表示

规则，自己编写正则表达式。

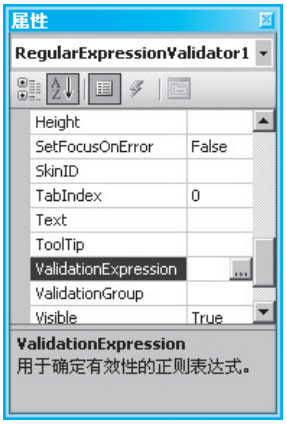


图 3-29 属性窗口

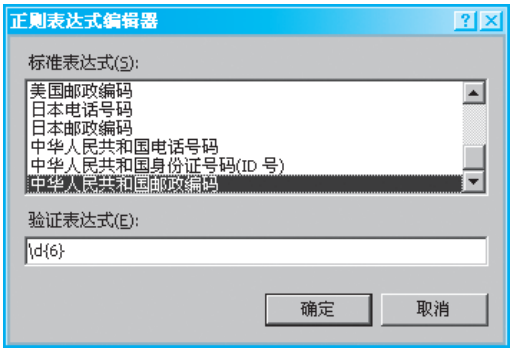


图 3-30 设置邮编的正则表达式

4. CustomValidator 自定义验证控件

ASP.NET 提供了不少验证控件，但需要做特殊的验证时仍无法满足需求，如果现有的 ASP.NET 验证控件无法满足需求，可以定义一个自定义的服务器端验证函数，然后使用 CustomValidator 控件来调用它。

```
protected void btnOK_Click(object sender, EventArgs e)
    {
        if (this.IsValid) // 判断页面验证是否全部通过
        {
            Response.Write( "<Script>alert(' 页面验证成功 ')</script>");
            // 提交到服务器的后续操作
        }
        else
        {
            Response.Write("<Script>alert(' 页面验证不成功 ')</script>"); // 取消提交到服务器的操作
        }
```

可以使用如下代码将事件处理程序绑定到方法。

```
<asp:TextBox ID="txtTel" runat="server" CssClass="input_text"></asp:TextBox>
<asp:CustomValidator ID="valxTel" runat="server" ControlToValidate="txtTel"
ErrorMessage="CustomValidator"onservervalidate="valxTel_ServerValidate"> 手机号格式不正确
</asp:CustomValidator>
```

5. 页面验证属性：IsValid

IsValid 属性是 Page 类的一个属性，用于指示整个页面是否全部验证通过，若验证通不过，则不提交到服务器端执行。

通常在页面拖放一个“提交”按钮，为其编写事件处理程序，代码如下。

```
protected void Button1_Click(object sender, EventArgs e)
    {
        if (this.IsValid) // 判断页面验证是否全部通过
        {
            ResultLbl.Text = " 页面验证成功 ";
            // 提交到服务器的后续操作
        }
        else
        {
            ResultLbl.Text = " 页面验证未成功 !"; // 取消提交到服务器的操作
        }
    }
```

任务实施

3.3.4　验证控件检验用户注册信息

1. 添加验证控件

在任务 3-2“Register.aspx”用户注册页面的基础上，按照功能分类，从工具箱拖放验证控件至页面上，用户名、密码：绑定 RequiredFieldVailidator 控件，不能为空，代码如下：

```
<asp:RequiredFieldValidator ID="rfvUserName" runat="server"
                            ControlToValidate="txtUserName" ErrorMessage="RequiredFieldValidator"
                            ForeColor="#CC0000"> 用户名不能为空！ </asp:RequiredFieldValidator>
<asp:RequiredFieldValidator ID="rfvPWD" runat="server"
                            ControlToValidate="txtPwd" ErrorMessage="RequiredFieldValidat
or"> 密码不能为空 </asp:RequiredFieldValidator>
```

确认密码：绑定 CompareValidator 控件，两次输入的密码一致，代码如下：

```
<asp:CompareValidator ID="cvPwdOk" runat="server" ControlToCompare="txtPwd"
                      ControlToValidate="txtPwdOK" ErrorMessage="CompareValidator">
密码不一致 </asp:CompareValidator>
```

身份证号、邮箱文本控件：绑定 RegularExpressionValidator 控件，输入的数据格式验证，代码如下：

```
<asp:TextBox ID="txtIDNumber" runat="server" CssClass="input_text"></asp:TextBox>

        </td>
        <td>
        <asp:RegularExpressionValidator ID="RegularExpressionValidator1" runat="server"
        ControlToValidate="txtIDNumber" ErrorMessage="RegularExpressionValidator"
        ValidationExpression="^(^\d{15}$|^\d{18}$|^\d{17}(\d|X|x))$"> 身份证号码格式不
正确 </asp:RegularExpressionValidator>
```

```
<asp:RegularExpressionValidator ID="revEmail" runat="server"
                    ControlToValidate="txtEmail" ErrorMessage="RegularExpression
Validator"
                    ValidationExpression="\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\
w+)*"> 邮件格式不正确 </asp:RegularExpressionValidator>
```

移动电话文本空间：通过 CustomValidator 实现自定义验证，选定定义的 CustomValidator 控件，切换到如图 3-31 所示的事件窗口。

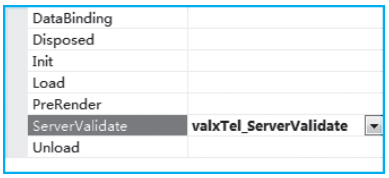


图 3-31 CustomValidator 验证控件的事件窗口

在“ServerValidate”属性后的对话框中双击生成 valxTel_ServerValidate 事件。

前台代码如下：

```
<asp:CustomValidator ID="valxTel" runat="server" ControlToValidate="txtTel"
                    ErrorMessage="CustomValidator" onservervalidate="valxTel_
ServerValidate"> 手机号格？不正确 </asp:CustomValidator>
```

完善后台 valxTel_ServerValidate 事件代码：

```
args.IsValid = (args.Value.Length == 11);
```

2. 设置验证控件的验证属性

所有验证控件都要设置的验证属性包括 ControlToValidate 属性：绑定到待验证的控件，Errormessage 属性：提示的错误信息，forecolor：错误信息的文本颜色，再根据控件设置自己的独有属性。

3. 使用页面验证属性 IsValid，提交按钮提交合法的数据

```
protected void ibtAdd_Click(object sender, ImageClickEventArgs e)
{
    if (this.IsValid) // 判断页面验证是否全部通过
    {
        // 提交到服务器的后续操作
    }
```

思考：如果通过 RegularExpressionValidator 控件验证“移动电话”项，其正则表达式是什么？

4. 任务运行

运行 Register.aspx 页面，效果如图 3-25 所示。

任务拓展

3.3.5 其他验证控件

1. 检查指定范围控件 RangeValidator

该控件用于检查用户输入数据的范围，数据类型可以是数字、字符串、

日期等。

使用 RangeValidator 控件对出生年月控件的输入进行范围验证，要求出生年月在 1989/1/1~2000/1/1，如图 3-32 所示。

图 3-32　范围验证

添加 ASP.NET 控件，设计界面如下，从工具箱拖放 RangeValidator 控件，并设置其相关属性见表 3-21。

表 3-21　RangeValidator 控件的属性

属　性	属 性 值
Errormessage	出生日期超出指定范围
Display	Dynamic
ControlToValidate	BirthdateTxt
MimimunValue	1989-1-1
MaximumValue	2000-1-1
Type	Date

页面主要设计代码如下：

```
<asp:RangeValidator ID="RangeValidator1" runat="server" ControlToValidate="BirthdateTxt"
  Display="Dynamic" ErrorMessage=" 出生日期超出指定范围 " MaximumValue="1989-1-1"
MinimumValue="2000-1-1" Type="Date" ForeColor="Red"></asp:RangeValidator>
```

2. 错误信息汇总控件 ValidationSummary

错误信息汇总控件用于汇总将页面中所有验证控件错误信息显示在一起，从而使页面内容更加紧凑。

将 ValidationSummary 控件拖放至窗体上即可。如图 3-33 所示，这时可以将页面上其他验证控件的 Display 属性设置为 None，即不显示出来，而直接汇总显示到 ValidationSummary 控件。

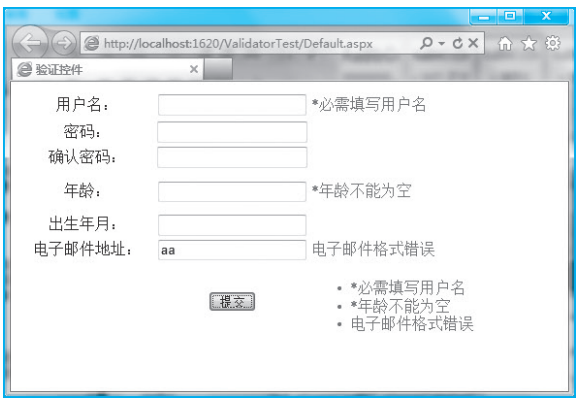


图 3-33　错误信息汇总

3.3.6 验证组属性 ValidationGroup

通过设置控件的 ValidationGroup 属性，可以将同一页面的验证控件分为不同的验证组。

如图 3-34 所示，设置添加按钮（ibtAdd）的 ValidationGroup 属性为“yanzheng”，返回按钮（ibtReturn）的 ValidationGroup 属性为空，页面上其他验证控件的 ValidationGroup 属性为“yanzheng”，使得当单击“添加”按钮时，首先进行输入数据的检验，检验通过再提交给服务器数据，而当单击“返回按钮”时，页面上验证控件不发挥作用，不进行数据验证。

图 3-34 按钮验证组

如验证控件（属于验证组“yanzheng”）：

```
<asp:RequiredFieldValidator id="rfvName" runat="server" ErrorMessage=" 名称不能为空！ "
ControlToValidate="tbUsername"  ValidationGroup="yanzheng"></asp:RequiredFieldValidator>
```

添加按钮（属于验证组“yanzheng”）：

```
<asp:ImageButton ID="ibtAdd" runat="server"
 ImageUrl="~/App_Themes/Default/Images/badd.gif" AlternateText=" 添加 "
 OnClick="ibtAdd_Click" ValidationGroup="yanzheng" />
```

而返回按钮（不属于任何验证组）

```
<asp:ImageButton ID="ibtReturn" runat="server" ImageUrl="~/App_Themes/Default/
Images/breturn.gif" AlternateText=" 返回 " OnClick="ibtReturn_Click" />
```

提示：通过验证组属性 **ValidationGroup**，将添加按钮与页面所有验证控件划为同一验证组，在触发添加按钮的提交信息到服务器功能前，首先由验证控件进行数据合法性，合法，才提交数据。

3.3.7 禁用数据验证

有可能用户在没有完全正确填写验证信息也要提交该页，此时就需要服务器控件避开客户端和服务器端验证，ASP.NET 提供了 3 种禁用验证的方式。

1. CausesValidation 属性

CausesValidation 属性设置相关控件提交给服务器数据前，是否进行数据数据验证。

设置【提交】按钮的 CausesValidation 属性为“false”，则在单击按钮时，不进行数据验证，页面上验证类控件不发挥作用。

```
<asp:ImageButton ID="ibtReturn" runat="server" ImageUrl="~/App_Themes/Default/
Images/breturn.gif" CausesValidation="false" AlternateText=" 提交 " OnClick="ibtReturn_Click" />
```

2. 禁用验证控件

将验证控件的 Enabled 属性设置为 false，页面在验证时将不会触发此验证控件。

3. 禁用客户端验证

将验证控件的 EnableClientScript 属性设置为“false”，页面在验证时将不会触发客户端验证。

思政小课堂
软件质量

项目实训

【实训题目】

① 掌握验证控件引入的意义。

② 掌握各类验证控件的应用场景。

③ 掌握验证控件与提交给服务器功能的结合使用，保证数据输入的合法性。

【实训内容】

① 网站中除了展示信息页面，定义其他页面输入数据的验证功能。

② 实现页面提交按钮的验证数据功能，取消按钮的不验证功能。

任务 3.4　设计产品分类导航及首页广告

用户控件的使用

任务陈述

任务构思与目标：网站首页通常都有新闻公告模块，展示最新信息，用以吸引顾客的注意，本案例利用用户控件设计 SelectOKShop 网站首页的产品导航栏，并显示广告，效果如图 3-35 所示。

图 3-35　电子商务网站导航、广告

任务设计：在 SelectOKShop 站点中使用 Panel、MultiView、Adator、UserControl 等控件实现网站首页的产品导航栏，并显示广告。

知识准备

3.4.1 Panel 控件

1. Panel 控件概述

Panel Web 服务器控件在 ASP.NET 网页内提供了一种容器控件，可以将 Panel 控件作为其他控件的容器或静态文本和其他控件的父级。Panel 控件具有以下优点。

（1）动态生成的控件的容器

Panel 控件可作为在运行时创建的控件的容器。

（2）对控件和标记进行分组

对于一组控件和相关的标记，可以把其放置在 Panel 控件中，通过操作此 Panel 控件将一组控件作为一个单元进行管理。例如，可以通过设置面板的 Visible 属性来隐藏或显示该面板中的一组控件。

（3）具有默认按钮的窗体

可将 TextBox 控件和 Button 控件放置在 Panel 控件中，通过将 Panel 控件的 DefaultButton 属性设置为面板中某个按钮的 ID 来定义一个默认的按钮。如果用户在面板内的文本框中进行输入时按 Enter，则与用户单击特定的默认按钮具有相同的效果。这有助于用户更有效地使用项目窗体。

（4）向其他控件添加滚动条

有些控件（如 TreeView 控件）没有内置的滚动条，可以在 Panel 控件中放置滚动条控件，若要向 Panel 控件添加滚动条，需要设置 Height 和 Width 属性，将 Panel 控件限制为特定的大小，然后再设置 ScrollBars 属性。

（5）页面的自定义区域

可使用 Panel 控件在页面中创建具有自定义外观和行为的区域，创建一个带标题的分组框：可设置 GroupingText 属性来显示标题。呈现页面时，Panel 控件的周围将显示一个包含标题的框。不能在 Panel 控件中同时指定滚动条和分组文本。如果设置了分组文本，其优先级高于滚动条。在页上创建具有自定义颜色或其他外观的区域：Panel 控件支持外观属性（例如 BackColor 和 BorderWidth），可以设置外观属性为页上的某个区域创建独特的外观。需要注意的是设置 GroupingText 属性将自动在 Panel 控件周围呈现一个边框。

2. Panel 控件的使用

① 在【设计】视图中，从工具箱的【标准】选项卡中，将 Panel 控件拖到页面上。

② 若要创建静态文本，请在控件中单击，然后键入文本。若要添加控件，

微课 3.4 广告控件

可以将其从【工具箱】拖到 Panel 控件中。

③ 拖动面板的边框可以调整控件的大小。

⚠ 注意：该控件会自动调整自身的大小以显示其所有的子控件（即使其超出了设置的高度）。

④ 设置 Panel 控件的属性，可以指定窗格与其子控件的交互方式。

Panel 控件的主要属性见表 3-22。

表 3-22　Panel 控件的主要属性

属性	描　述
HorizontalAlign	指定子控件在面板内的对齐方式（左对齐、右对齐、居中或两端对齐）
Wrap	指定面板内过宽的内容是换到下一行，还是在面板边缘处截断
Direction	指定控件的内容是从左至右呈现还是从右至左呈现。当在页面上创建与整个页面的方向不同的区域时，此属性非常有用
ScrollBars	如果已经设置了 Height 和 Width 属性以将 Panel 控件限制为特定的大小，则可以通过设置 ScrollBars 属性来添加滚动条
GroupingText	在 Panel 控件周围呈现边框和标题

⚠ 注意：设置 GroupingText 属性会导致滚动条不显示。

3.4.2　AdRotator 广告控件

网站的盈利模式主要有卖广告、卖产品（如淘宝网）和卖服务（如百度）等，其中，卖广告是主要的盈利模式。在 Web 应用开发过程中应该考虑广告模块的设计与实现。ASP.NET 为开发人员提供了 Adrotator 广告控件为页面在加载时提供一个或一组广告。广告控件可以从固定的数据源中读取（如 XML 或数据源控件），并从中自动读取出广告信息。当页面每刷新一次，广告显示的内容刷新一次。广告控件最好放置在 Panel 控件以及模板内。广告控件需要包含图像的地址的 XML 文件，且该文件用来指定每个广告的导航连接。广告控件最常用的属性是 AdvertisementFile，该属性用于配置相应的 XML 文件，所以必须首先按照标准格式创建一个 XML 文件。使用 AdRotator 服务器控件步骤如下。

1. 创建 XML 文件

创建一个包含着广告细节的 XML 文件，打开【添加新项】对话框，选择【XML 文件】，并命名为“Ad.xml”，如图 3-36 所示。

向 XML 文件中添加固定格式的 XML 代码，所有广告信息包含在一对 <Advertisements> </Advertisements> 中，每个广告包含在 <Ad></Ad> 中，具体解释示例如下。

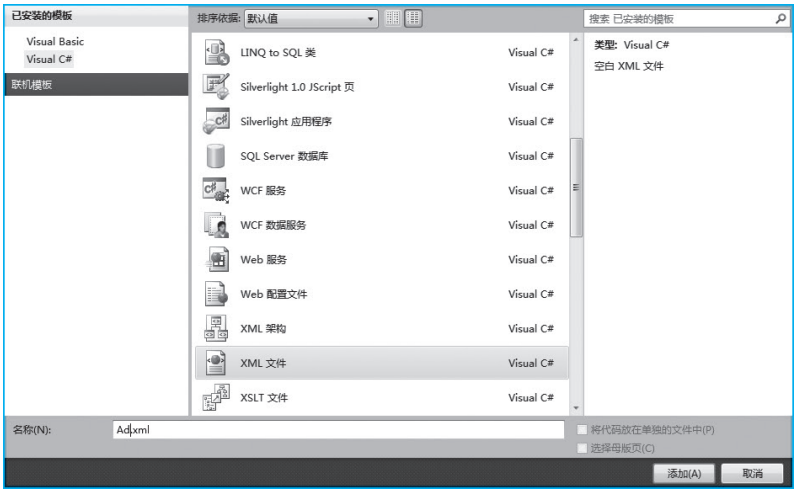


图 3-36 创建一个 XML 文件

```
<Advertisements>
    <Ad>
        <ImageUrl> 包含显示图像的 URL，可以是绝对路径，也可以是相对于显示广告的页面的相对路径。</ImageUrl>
        <NavigateUrl> 包含目标 Web 站 点的 URL </NavigateUrl>
        <AlternateText> 包含着一些文本，当鼠标移过图像时，作为提示信息显示出来。换句话说，这是广告图像中 ALT 元素的文本。 </AlternateText>
        <Keyword> 这个可选元素包含了广告所属的类别。这样就使各种类别的广告都在同一个 XML 中，然后使用 AdRotator 控件中的 Keywordfilter 属性在给定页面上对广告进行过滤。 </Keyword>
        <Impressions> 指出广告的相对开销，即广告出现的权重 </Impressions>
    </Ad>
</Advertisements>
```

⚠ 注意：创建广告文件时是区分大小写的。因此，其中所包含元素的大小写应该拼写正确，例如：应该使用 **<Ad>** 而不是 **<ad>**。

【实例 3-18】 创建广告 XML 文件，显示三个广告横幅的信息。

```
<Advertisements>
    <Ad>
        <ImageUrl>~/images/ad/huazhuangpin.jpg</ImageUrl>
        <NavigateUrl>http://www.sina.com</NavigateUrl>
        <AlternateText>SINA</AlternateText>
        <Keyword>huangzhuang</Keyword>
        <Impressions>50</Impressions>
    </Ad>
    <Ad>
        <ImageUrl>~/images/ad/ 衣服 .jpg</ImageUrl>
        <NavigateUrl>http://www.sohu.com</NavigateUrl>
```

```
        <AlternateText>SOHU</AlternateText>
        <Keyword>yifu</Keyword>
        <Impressions>10</Impressions>
    </Ad>
    <Ad>
        <ImageUrl>~/images/ad/ 电子产品 .jpg</ImageUrl>
        <NavigateUrl>http://www.china.com</NavigateUrl>
        <AlternateText>CHINA</AlternateText>
        <Keyword>Computer</Keyword>
        <Impressions>40</Impressions>
    </Ad>
</Advertisements>
```

2. AdRotator 服务器控件与 XML 文件的关联

在 ASP.NET 页面中创建一个 AdRotator 服务器控件，将广告 XML 文件链接到该控件。使用以下服务器控件标记来完成：

```
<asp:AdRotator ID="adMain" runat="server" AdvertisementFile="~/Ad.xml"  />
```

AdvertisementFile 属性指示广告文件即包含广告信息的 XML，为了显示图象，必须要引用 AdRotator 控件。由于 KeywordFilter 属性没有设置，所以当刷新页面时会看到所有这三个广告图像以随机方式显示。

⚠ 注意：广告控件本身并不提供点击统计，所以无法计算广告是否被用户点击或者统计用户最关心的广告。

3.4.3　MultiView 和 View 视图切换控件

ASP.NET 中的 MultiView 和 View 可以作为其他控件的容器，可方便地显示信息的替换视图。通常，MultiView 和 View 搭配使用，可以让用户在同一页面中通过切换到每个选项卡看到要看的内容，而不用每次都重新打开一个新的窗口。

MultiView 控件用作一个或多个 View 控件的外部容器。View 控件又可包含标记和控件的任何组合。MultiView 控件一次显示一个 View 控件及该 View 控件内的标记和控件。通过设置 MultiView 控件的 ActiveViewIndex 属性，可以指定当前可见的 View 控件。

未选择某个 View 控件时，该控件不会呈现到页面中。但是，每次呈现页面时都会创建所有 View 控件中的所有 Web 服务器控件的实例，并且将这些实例的值存储为页面视图状态的一部分。

无论是 MultiView 控件还是各个 View 控件，除当前 View 控件的内容外，都不会在页面中显示任何标记。例如，这些控件不会以与 Panel 控件相同的方式来呈现 div 元素，但是也不可以作为一个整体应用于当前 View 控件的外观属性。可以将一个主题分配给 MultiView 或 View 控件，控件将该主题应用于当前 View 控件的所有子控件。

可以使用 ActiveViewIndex 属性或 SetActiveView 方法定义活动视图。如果 ActiveViewIndex 属性为空，则 MultiView 控件不向客户端呈现任何内容。如果活动视图设置为 MultiView 控件中不存在的 View，则会在运行时引发 ArgumentOutOfRangeException 异常。

ActiveViewIndex 属性：用于获取或设置当前被激活显示的 View 控件的索引值。默认值为 -1，表示没有 View 控件被激活。

Views ：获取 MultiView 控件的 View 集合。

NextView ：CommandName，定位到下一个视图。

PrevView ：CommandName，定位到前一个视图。

SwitchViewByID ：CommandName，定位到对应的视图 ID。

SwitchViewByIndex ：CommandName，定位到对应的索引。

SetActiveView 方法：将制定的 View 控件设置为 MultiView 控件的活动视图，该方法参数为 View 控件的 ID。

提示：每个 View 控件都支持 Controls 属性，该属性包含该 View 控件中的控件集合。但是，可以在代码中单独引用 View 控件中的控件。

【实例 3-19】 MultiView 和 View 实现公司简介和欢迎加盟选项卡效果。

```
<asp:Button ID="btnIntroduction" runat="server" BorderWidth="0" Text=" 公司简介 "
        onclick="btnIntroduction_Click1" />
    <asp:Button ID="btnWelcome" runat="server" BorderWidth="0" Text=" 欢迎加盟 "
        onclick="btnWelcome_Click"  />
<table style="border: 1px ridge #0000FF; width: 100%;">
    <tr valign="top">
        <td style="width: 300; height: 250">
            <asp:MultiView ID="mvIntroduce" runat="server" ActiveViewIndex="1">
                <asp:View ID="vIntroduce" runat="server">
                    <asp:Label ID="lblIntroduce" runat="server"></asp:Label>
                </asp:View>
                <asp:View ID="vInjoin" runat="server">
                    <asp:Label ID="lblInjoin" runat="server"></asp:Label>
                </asp:View>
            </asp:MultiView>
        </td>
    </tr>
</table>
protected void Page_Load(object sender, EventArgs e)
    {
        lblIntroduce.Text = @" 本公司为一家集软件开发、传统网络开发以及无线网络开发的大型科技公司。公司自成立以来经过长期发展，取得了 80 多项国家专利！ "+"\r\n";
        lblInjoin.Text = @" 欢迎有志之士加盟本公司！ 软件开发工程师要求：
        1. 热爱软件行业；2. 熟悉、net 框架，能熟练使用 C# 和 JavaScript 进行编码；
        3. 熟悉 SQL Server 等常用的数据库 4. 有良好的编码规范习惯；
```

```
            5. 做事认真负责，服从公司的工作安排；6. 吃苦耐劳，敬业。 ";
        }
        protected void btnIntroduction_Click1(object sender, EventArgs e)
        {
            mvIntroduce.SetActiveView(vIntroduce);
        }
        protected void btnWelcome_Click(object sender, EventArgs e)
        {
                mvIntroduce.SetActiveView(vInjoin);
        }
```

【实例 3-20】 MultiView 控件实现页码切换。

```
    <asp:multiview ID="Multiview1" ActiveViewIndex="0" runat="server">
            <asp:View ID="firstView" runat="server">
                <span> 第一页 </span>
                 <asp:Button ID="btnNext" runat="server" Text=" 下一页 " CommandName=
"NextView" />
            </asp:View>
            <asp:View ID="secondView" runat="server">
                <span> 第二页 </span>
                <asp:Button ID="btnPrev" runat="server" Text=" 上一页 "CommandName=
"PrevView" />
                <asp:Button ID="btnNext2" runat="server" Text=" 下一页 " CommandName=
"NextView" />
            </asp:View>
            <asp:View ID="thirdView" runat="server">
                <span> 第三页 </span>
                <asp:Button ID="btnFirst" runat="server" Text="第一页" CommandName="Switch
ViewByID" CommandArgument="firstView" />
                <asp:Button ID="btnSecond" runat="server" Text="第二页" CommandName="Switch
ViewByIndex" CommandArgument="1" />
            </asp:View>
        </asp:multiview>
```

任务实施

3.4.4　创建并使用电子商务网站的广告栏

1. 将广告图片复制到网站项目中

2. 创建 XML 文件

在项目中添加一个 XML 文件，将文件命名为“Ad.xml”，将实例 3-17 的代码添加到文件中。

3. 创建 Panel 控件

从【工具箱】中，将 Panel 控件拖放到 Index.Aspx 页中的中上部位置，生

成代码如下：

```
<asp:Panel ID="Panel1" runat="server" Height="180px" Width="350px">
</asp:Panel>
```

4. 创建 AdRotator 控件

从【工具箱】中，将 AdRotator 控件拖放到 Index.Aspx 页的“Panel1”中，并与“Ad.xml”文件进行关联，生成代码如下：

```
<asp:AdRotator ID="adMain" runat="server" AdvertisementFile="~/Ad.xml"  />
```

5. 运行

刷新浏览器，广告图片将根据各自权值显示，运行效果如图 3-37 所示。

图 3-37 AdRotator 控件运行效果图

3.4.5 创建并使用电子商务网站的产品导航栏

1. 设置导航按钮

将 3 个 Button 控件拖放到 Index.Aspx 页的左侧 ID 为“ContentPlaceHolderLeft”的容器中，将按钮的“Text”属性分别设置为“电子产品”“化妆品”“服饰”，并修改控件的样式，代码如下。

```
<asp:Button ID="btnEle" runat="server" BorderWidth="0" Text="电子产品" onclick="btnEle_Click" />
<asp:Button ID="btnFashion" runat="server" BorderWidth="0" Text="化妆品" onclick="btnFashion_Click" />
<asp:Button ID="btnDress" runat="server" BorderWidth="0" Text="服　饰" onclick="btnDress_Click" />
```

2. 设置 MultiView 控件

设置 MultiView 控件相应的代码如下。

```
<table style="border: 1px ridge #0000FF; width: 100%;">
  <tr valign="top">
    <td style="width: 300; height: 250">
```

```
<asp:MultiView ID="mvNavigation" runat="server" ActiveViewIndex="1">
    <asp:View ID="vEle" runat="server">
        <table style="width: 100%;">
            <tr>
                <td>
<asp:LinkButton ID="lnkbtnComputer" runat="server"> 台式计算机 </asp:LinkButton>
                </td>
                <td>
 <asp:LinkButton ID="lnkbtnPad" runat="server"> 平板电脑 </asp:LinkButton>
                </td>
                <td>
  <asp:LinkButton ID="lnkbtnPhone" runat="server"> 手机 </asp:LinkButton>
                </td>
                </tr>
                <tr>
                  <td>
   <asp:LinkButton ID="lnkbtnNote" runat="server"> 笔记本 </asp:LinkButton>
                  </td>
                  <td>
   <asp:LinkButton ID="lnkbtnTV" runat="server"> 电视 </asp:LinkButton>
                  </td>
                  <td>
    <asp:LinkButton ID="lnkbtnComera" runat="server"> 相机 </asp:LinkButton>
                  </td>
                </tr>
</table>
```

3. 运行

运行效果如图 3-38 所示。

电子产品	化妆品	服饰
台式计算机	平板电脑	手机
笔记本	电视	相机

图 3-38　产品导航

任务拓展

3.4.6　第三方控件的使用

日常生活中很多电子产品、汽车等虽然是同一型号，但是所带配件除了标准配件以外，还有许多不同厂家提供的相应配件，如手机、专业相机、汽车等。ASP.NET 提供了 90 多种标准的内置控件，但是仍然不能满足程序员的编程需求，为此一些专门的机构开发了一些可以添加到 ASP.NET 中的控件，称之为第三方的控件。

常用的第三方控件有弹窗控件（PopupWin）、文本编辑器控件（FreeTextBox）、皮肤控件（DYLIKEBUTTON）等。第三方控件虽然来自不同的机构，但是均须遵循 ASP.NET 的规范，第三方控件除了自身的属性和方法外，在部署、使用上均有共同的地方。下面以第三方控件 PopupWin 为例，介绍第三方控件的使用。

PopupWin 控件用于弹出提示框效果，此控件支持多种弹出效果，同时也可以设置锚点等，该控件的下载地址是：http://www.codeproject.com/KB/custom-controls/asppopup.aspx。

1. PopupWin 弹出窗口控件的添加

（1）打开 Visual Studio2010，在左侧的【工具箱】的【常规】选项卡上点击右键，选择【重命名选项卡】，如图 3-39 所示，将“常规”重命名为“第三方控件”。

（2）打开【第三方控件】，在空白位置处单击右键选择【选择项】，如图 3-40 所示，打开如图 3-41 所示的【选择工具箱项】对话框，选择【.NetFrameWork 组件】选项卡，单击【浏览】按钮，选择需要添加的控件文件“EeekSoft.Web.PopupWin.dll”。

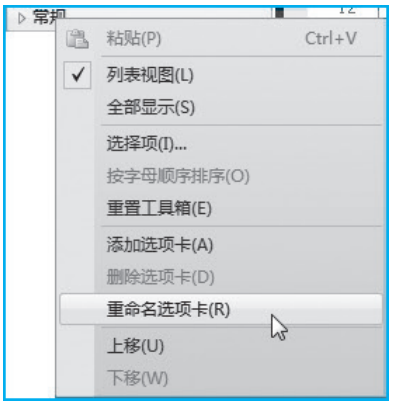


图 3-39 重命名工具箱选项卡

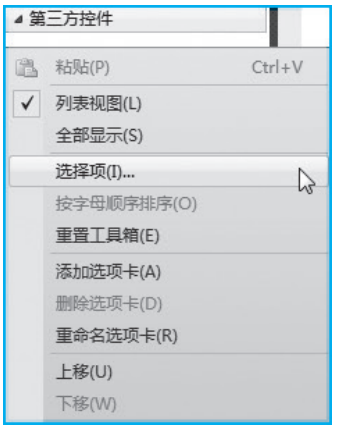


图 3-40 工具箱选项卡中添加控件

（3）单击【确定】按钮后，会在工具箱【第三方控件】选项卡出现两个控件，分别是 PopupWin 和 PopupWinAnchor，如图 3-42 所示。

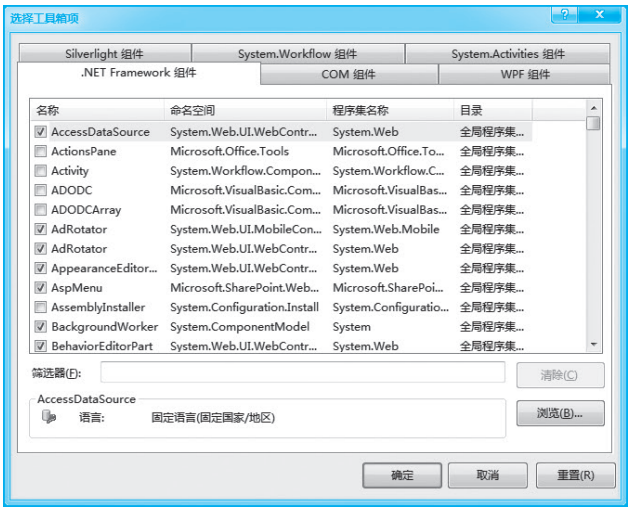


图 3-41 选择工具箱项

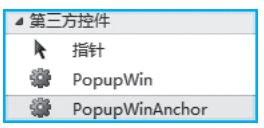


图 3-42 控件添加效果图

（4）将【PopupWin】控件拖放的 Web 页的相应位置后，即可与普通控件一样使用，控件生成代码如下。

```
<%@ Register Assembly="EeekSoft.Web.PopupWin" Namespace="EeekSoft.Web"
TagPrefix="cc1" %>
                    <cc1:PopupWin ID="PopupWin1" runat="server" />
```

PopupWinAnchor 控件属于事件响应控件，可实现当单击某个控件或者是单击某些内容的时候自动弹出窗口。

2. 使用 PopupWin 控件的常用属性

ActionType：动作类型（点击连接后），返回 PopupAction 枚举。（注意：如果要使用相关的点击事件，如 OnLinkClicked 和 OnPopupClosed，此处必须设为 RaiseEvents）。

Text: 设置或获取新窗口里要显示的文本。

Link：设置或获取点击连接时打开的地址或脚本，如：Link="http：//www.zzwater.com.cn" LinkTarget="_blank" ActionType="OpenLink"，其中 actiontype 用于说明是否弹出大页面窗口等。

LinkTarget：设置或获取连接打开的目标方式。

ColorStyle：设置或获取颜色样式，返回 PopupColorStyle 枚举。

Message：设置或获取弹出窗口显示的信息。

Title：设置或获取弹出窗口和新窗口的标题。

GradientLight：设置或获取亮度的颜色。

GradientDark：设置或获取暗度的颜色（在 Mozilla 里即背景色）。

TextColor：设置或获取文本颜色。

LightShadow：设置或获取亮度阴影的颜色。

DarkShadow：设置或获取暗度阴影的颜色。

Shadow：设置或获取阴影颜色。

DockMode：设置或获取弹出窗口的收缩状态，返回 PopupDocking 枚举，如：DockMode="BottomRight"，则显示的窗口从右侧弹出。

OffsetX：设置或获取 X 轴的偏移坐标（从左或右）。

OffsetY：设置或获取 Y 轴的偏移坐标（从底部）。

HideAfter：设置或获取窗口显示的时间，默认为 500 毫秒（-1 为无限时间）。

PopupSpeed：设置或获取弹出的速度，默认为 20。

ShowAfter：设置或获取显示弹出窗口之前的延迟时间，默认为 1000 毫秒。

AutoShow：页面加载时自动显示弹出窗口（在设置的 ShowAfter 属性之后）。

DragDrop：设置或获取是否允许拖动弹出窗口。

WindowSize：设置或获取打开窗口大小。

WindowScroll：设置或获取新窗口是否允许滚动条。

ShowLink：是否在弹出窗口中显示连接和启用动作。

【实例 3-21】 PopupWin 的应用。

运行效果如图 3-43 所示。

```
protected void Page_Load(object sender, EventArgs e)
    {
        PopupWin1.Message = " 欢迎您访问本网站，近期电子产品将开展推广优惠活动，敬请关注 " ;
        PopupWin1.AutoShow = true;
        PopupWin1.Title = " 欢迎 ";
    }
```

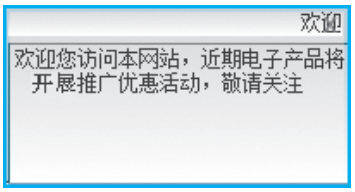


图 3-43 弹窗效果图

项目实训

【实训题目】

掌握用户控件的使用，提高代码的复用性。

【实训内容】

将 SelectOkShop 电子商务网站的可复用功能模块用用户控件实现，如：用户搜索功能模块、商品展示功能模块等。

任务 3.5 设计电子商务网站的新闻公告栏

用户控件的使用

任务陈述

任务构思与目标：网站首页通常都有新闻公告模块，展示最新信息，用以吸引顾客的注意，本任务利用用户控件设计 SelectOKShop 网站首页的新闻公告栏，滚动显示，效果如图 3-44 所示。

任务设计：在 SelectOKShop 站点根目录下添加文件夹“UserControl”，站点中创建的用户控件均放在该文件夹下，创建用户控件 News.ascx，实现新闻内容展示，将新闻用户控件应用到网站首页。

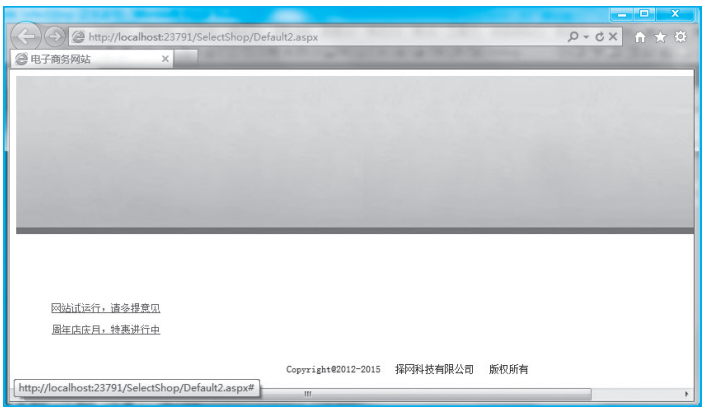


图 3-44　电子商务网站滚动新闻公告栏

知识准备

3.5.1　用户控件

1. 用户控件概述

ASP.NET 的服务端控件使得 Web 开发工作变得更为简单，功能更为强大。但是，如果服务端没有所要求的控件，则可以自定义控件来取代 .NET 提供的控件，这种控件就是用户控件。用户控件能够在多个 ASP.NET Web 应用程序之间划分和重复使用公共用户界面（UI）功能，提高了代码的重用性。与 Web 窗体页一样，用户控件可以在第一次请求时被编译并存储在服务器内存中，从而缩短以后请求的响应时间。与服务器控件不同的是，不能独立地请求用户控件，用户控件必须包括在 Web 窗体页内才能使用。

2. 创建用户控件

在 Visual Studio 2010 中创建用户控件的步骤如下。

（1）在解决方案资源管理器中添加【新项】，弹出【添加新项】窗口，如图 3-45 所示，选择【Web 用户控件】选项可创建用户控件 WebUserControl.ascx 文件。

（2）在 WebUserControl.ascx 文件中可添加任意的 Web 服务器控件创建用户控件。在 WebUserControl.ascx 代码视图中有如下页面指令。

```
<%@ Control Language="C#" AutoEventWireup="true" CodeFile="WebUserControl.ascx.cs" Inherits="userControl_WebUserControl" %>
```

用户控件的使用

微 课 3.5　用 户 控件

提示：用 @ Control 命令表明这是一个用户控件，表明用户控件继承 WebUserControl 类。

用户控件声明语法与创建 Web 窗体页所采用的语法类似，用户控件和网页之间的主要区别如下。

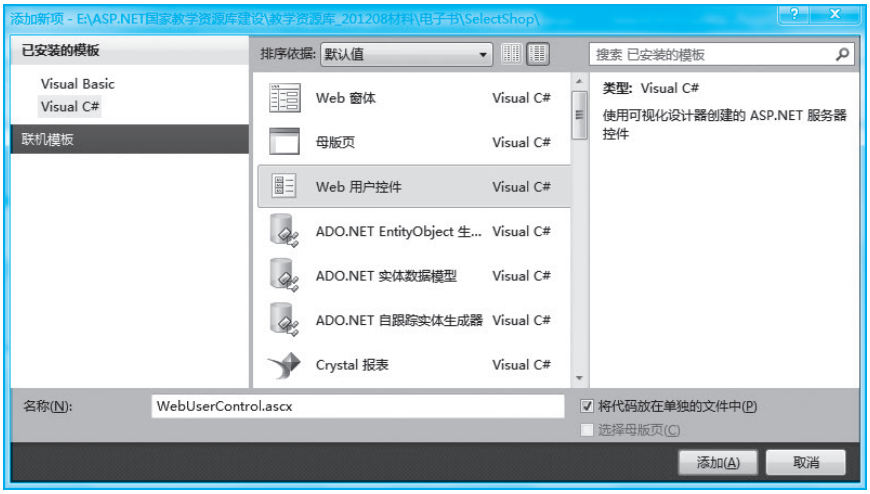


图 3-45 创建 Web 用户控件

① 用户控件以 Control 指令开头，而网页以 Page 指令开头。

② 用户控件使用的扩展名是 .ascx，而网页的扩展名为 .aspx 。

③ 用户控件后台代码从 System.Web.UI.UserControl 类继承。（其实 UserControl 类和 Page 类继承自同一个 TemplateControl 类，因此，二者共享多种方法和事件）

④ 用户控件不能被客户端浏览器直接请求，用户控件需要嵌入到其他网页中使用。

⑤ 用户控件中没有 html，body 和 form 元素，这些元素不允许位于宿主中。

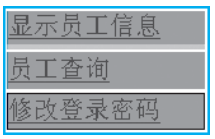


图 3-46 不同页面具有相同的局部元素

【实例 3-22】 定义用户控件。

网站的不同页面共享同一局部元素，如图 3-46 所示，因此将其定义成用户控件（单独的 .ascx 文件）。

页面设计代码如下。

```
<%@ Control Language="C#" AutoEventWireup="true" CodeFile="myControl.ascx.cs" Inherits="myControl" %>
        <table border="0" style="width: 160px;">
        <tr>
        <td style="height: 30px; background-color: darkgray; border-bottom-style: solid; 1px; #ffffff; width: 222px;"><a href="#"> 显示员工信息 </a> </td>
        </tr>
        <tr>
          <td  style="height: 30px; background-color: darkgray; text-align: left; border-bottom: white 1px solid; width: 222px;"><a href="#"> 员工查询 </a></td>
        </tr>
        <tr>
        <td style="height:30px; background-color: darkgray; text-align: left; border-bottom: #ffffff 1px solid; width: 222px;"><a href="#"> 修改登录密码 </a></td>
```

```
        </tr>
        </table>
```

⚠ 注意：C# 版本指 C# 语言规范的版本；Visual C# 版本是开发工具，是 Visual Studio 的一个组件。

3.5.2　在 Web 页面中使用用户控件

只有当包括在 Web 窗体页中时，用户控件才可以工作。当一个请求到达某一页而该页包含用户控件时，该用户控件的处理过程与 ASP .NET 服务器控件的处理过程一致。

在要包含用户控件的 Web 窗体页中，需声明一个 @ Register 指令，该指令包括如下属性。

① tagprefix 属性，该属性将前缀与用户控件相关联，该前缀将包括在用户控件元素的开始标记中，相当于 Web 服务器控件中 < asp:TextBox>< /asp:TextBox> 中的 asp。

② tagname 属性，该属性将名称与用户控件相关联，该名称将包括在用户控件元素的开始标记中，相当于 Web 服务器控件中 < asp:TextBox>< /asp:TextBox> 中的 TextBox。

③ Src 属性，该属性定义要包括在 Web 窗体页中的用户控件文件的虚拟路径。

⚠ 注意：Src 属性值可以是到应用程序的根目录中的用户控件源文件的相对或绝对路径。为方便使用，建议使用相对路径。符号“~”表示应用程序的根目录。

【实例 3-23】 在 Web 页面中使用用户控件，如图 3-47 所示。

图 3-47　用户控件使用

Web 页面代码如下。

```
        <%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default2.aspx.cs"
Inherits="Default2" %>
    <%@ Register Src="myControl.ascx" TagName="myControl" TagPrefix="uc1" %>
    <html xmlns="http://www.w3.org/1999/xhtml" >
    <head id="Head1" runat="server">
    </head>
    <body>
        <form id="form2" runat="server">
        <div>
```

用户控件的使用

```
            <table style="width: 458px; height: 226px" border="1">
                <tr>
                    <td rowspan="3" valign="top" style="width: 163px">
                        <uc1:myControl ID="MyControl1" runat="server" />    </td>
                    <td colspan="2" rowspan="3">    </td>
                </tr>
                 </table>
        </div>
    </form>
</body>
</html>
```

提示：<%@ Register Src ="myControl.ascx" TagName="myControl" TagPrefix="uc1" %>。其中 Src 指定用户控件的路径，TagPrefix 指定用户控所属的组，TagName 用于给用户控件指定名称。

页面具体引用控件指令：<uc1:myControl ID="MyControl1" runat="server" />，基本上和内置控件一样，只是开头用“自定义名称空间：控件名称”，而不是“asp: 控件名称”。

用户控件的使用

任务实施

3.5.3 创建并使用电子商务网站的新闻公告栏

任务运行效果如图 3-48 所示。

图 3-48 用户控件设计新闻公告栏

1. 创建用户控件

选择 SelectOKShop 站点根目录下文件夹 UserControl，选择【添加新项】命令，在给出的模板中选择【Web 用户控件】选项，用户控件命名为“News.ascx”。

2. 编辑用户控件

打开用户控件文件 News.ascx，编辑新闻内容页面代码如下。

```
<%@ Control Language="C#" AutoEventWireup="true" CodeBehind="newsUC.ascx.
cs" Inherits="Login1.UserControls.newsUC" %>
```

```
<p><a href="#"> 网站试运行，请多提意见 </a></p>
<p><a href="#"> 周年店庆月，特惠进行中 </a></p>
```

提示：用户控件文件 News.ascx 定义控件内容的方法与 Web 页面相同，可以放置除了 body、html、head、form 元素以外的其他标签元素和 ASP.NET 控件。

3. 在 Web 页面引用用户控件

打开 Default.aspx 设计页面，在 Visual Studio 2010 解决方案管理器中选中“News.ascx”，将其手动拖到应用了母版的网站首页 Index.aspx 的 Content PlaceHolderLef 左侧区域，如图 3-49 所示。

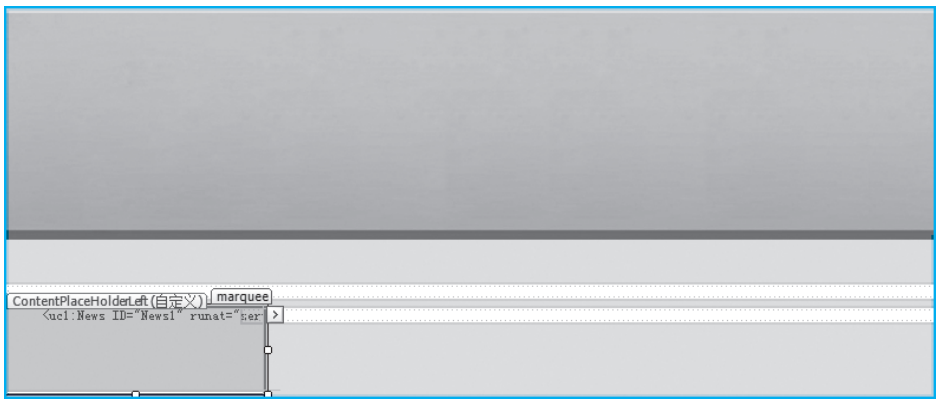


图 3-49　用户控件拖至 Web 页面

```
<%@ Page Title="" Language="C#" MasterPageFile="~/SelectOKShop/MasterPageQian.master" AutoEventWireup="false" CodeFile="Default.aspx.cs" Inherits="_Default" %><%--引用用户控件 --%>
<%@ Register src="~/userControl/News.ascx" tagname="News" tagprefix="uc1" %>
<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolderSitemap" Runat="Server">
<%--News.ascx 控件在内容控件 --%><%-- 跑马灯效果 --%>
    <marquee onMouseOver="this.stop()" onMouseOut="this.start()" scrollamount="1" scrolldelay="10" direction="up"
        style="text-align:center; height: 70px; width: 218px;">
        <uc1:News ID="News1" runat="server" /></marquee>
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolderLeft" Runat="Server">
</asp:Content>
<asp:Content ID="Content3" ContentPlaceHolderID="ContentPlaceHolderRight" Runat="Server">
</asp:Content>
```

4. 运行该页面

运行效果如图 3-44 所示。

任务拓展

3.5.4 访问用户控件的属性

标准服务器控件均有其自身的属性和方法，用户控件也可以定义自己的属性和方法，并在 Web 页中调用。

【实例 3-24】 在 Web 页面中使用用户控件，如图 3-50 所示。

```
public partial class Navigater : System.Web.UI.UserControl
{
    private string userName="xzp";
        public string UserName
        {
            get
            {
                return userName;
            }
            set
            {
                userName=value;
            }
        }
        public string UserID
        {
            get { return txtName.Text; }
            set { txtName.Text = value; }
        }
        public string Password
        {
            get { return txtPassword.Text; }
            set { txtPassword.Text = value; }
        }
    }
```

3.5.5 动态修改用户控件的内容

1. 设计用户控件

页面代码如下。

```
<%@ Control Language="C#" AutoEventWireup="true" CodeFile="Control.ascx.cs" Inherits=
"Control" %>
    <div>
        <table style="width: 539px; height: 192px" border="1">
            <tr>
```

微课 3.6 文件路径

```
                    <td> 用户名：</td>
                     <td> <asp:TextBox ID="txtName" runat="server" AutoPostBack="True"></
asp:TextBox></td>
                  </tr>
                   <tr>
                      <td> 密码：</td>
                      <td> <asp:TextBox ID="txtPassword" runat="server" TextMode=
"Password"></asp:TextBox></td>
                    </tr>
                    <tr>
                        <td> <asp:Button ID="Button10" runat="server" Text=" 确定 " /></td>
                        <td>                    </td>
                        <td>                    </td>
                    </tr>
                </table>
            </div>
```

2. 用户控件定义公用的属性

```
    public partial class Control : System.Web.UI.UserControl
        public String UserID
         {
              get { return txtName.Text; }
              set { txtName.Text = value; }
         }
        public string Password
        {
              get { return txtPassword.Text; }
              set { txtPassword.Text = value; }
        }
```

3. Web 页面引用用户控件并动态修改其内容

```
    <%@ Page Language="C#" AutoEventWireup="true"  CodeFile="Default.aspx.cs"
Inherits="_Default" %>
    <%@ Register Src="Control.ascx" TagName="Control" TagPrefix="uc1" %>
    <html xmlns="http://www.w3.org/1999/xhtml" >
    <head id="Head1" runat="server">
        <title> 无标题页 </title>
    </head>
    <body>
    <h2> 用户控件示例   </h2>
        <form id="form1" runat="server">
        <div>

    <uc1:Control ID="login" runat="server" />
    <p><asp:Label ID="message" runat="server" Text="Label"></asp:Label>
                </p>
```

```
            </form>
        </body>
    public partial class _Default : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
            {
             message.Text = " 用户名为：" + login.UserID + "<br>";
             Login. Password="123";
    }
```

项目实训

【实训题目】

掌握用户控件的使用，提高代码的复用性。

【实训内容】

将 SelectOkShop 电子商务网站的可复用功能模块用用户控件实现，如：用户搜索功能模块、商品展示功能模块等。

单元小结

本单元主要介绍了运行在服务器端的 HTML 服务器控件、ASP.NET 控件的使用（控件的主要属性、事件驱动机制等），使用其设计出较好交互功能的 Web 页面，使读者理解 Web 窗体页面服务器控件的功能、与传统 HTML 控件的区别，还重点介绍了用于服务器控件数据验证的验证控件，使用其能够方便地完成页面输入数据的验证功能，结合代码的重用思想理解用户控件的创建和使用。

评价体系表

专业能力测评表

（在□中打 √，A 理解，B 基本理解，C 未理解）

专业核心能力	评价指标	自测结果
运用服务器控件设计动态交互 Web 页面的能力	1. 能够区分服务器控件、客户端控件	□ A □ B □ C
	2. 能够使用服务器控件设计友好交互页面	□ A □ B □ C
	3. 能够使用服务器控件设计后台代码	□ A □ B □ C

续表

专业核心能力	评价指标	自测结果
运用验证控件验证服务器控件输入数据有效性的能力	1．能够理解验证控件的引入意义 2．能够按照验证功能分类正确选择验证控件进行服务器控件数据有效性的验证	□A □B □C □A □B □C
运用用户控件创建可重用控件的能力	1．能够使用用户控件创建可复用的功能模块 2．能够在 Web 页面中使用创建好的用户控件 3．能够通过代码动态修改用户控件的值	□A □B □C □A □B □C □A □B □C
学生签字：	教师签字： 年 月 日	

职业能力测评表

（在□中打 √，A 通过，B 基本通过，C 未通过）

职业核心能力	评价指标	自测结果
自我管理能力	1．培养自己的责任心 2．能够管理自己的时间 3．所学知识的灵活运用	□A □B □C □A □B □C □A □B □C
合作交流能力	1．知道如何尊重他人的观点 2．能与其他人有效沟通 3．能获取信息并反馈信息	□A □B □C □A □B □C □A □B □C
解决问题能力	1．能够有效使用信息资源 2．能客观分析事物的现状 3．能发现并解决常规问题	□A □B □C □A □B □C □A □B □C
设计创新能力	1．能够提出有价值的观点 2．能够运用不同的思维方式 3．能够制定创新方案并实施	□A □B □C □A □B □C □A □B □C
学生签字：	教师签字： 年 月 日	

单元练习题

一、单项选择题

1．现有一人口年龄调查输入框，年龄范围为 10 ～ 120，则最好使用（　　）验证控件。

A．RequiredFieldValidator　　B．CompareValidator

C．RangeValidator　　D．RegularExpressionValidator

2．某 ASP.NET 的 Web 项目，名称为“TestWeb”。在其根目录下建立两个目录 dic1 和 dic2，两目录下分别有文件 file1.aspx 和 file2.aspx。现在需要编写代码从 file1.aspx 跳转到 file2.aspx，正确的代码是（　　）。

A．Response.Redirect(“dic2/file2.aspx”)

B．Response.Redirect(“~\dic2\file2.aspx”)

C．Response.Redirect(“../dic2/file2.aspx”)

D．Response.Redirect(“..\dic2\file2.aspx”)

3．如需在 Calendar 控件的 SelectionChanged 事件发生时得到用户选择的日期，则需要用到 Calendar 的（　　）属性。

A．SelectDate　　B．SelectedDate

C．ChangeDate　　D．ChangedDate

4．定义一个验证控件如下：

<asp:RequiredFieldValidator id="RFValidator1" ControlToValidate="TextBox1" Text="Error" ErrorMessage=" 错误 " runat="server"/>

当验证没有通过时，显示的提示信息是（　　）。

A．Error　　B．错误

C．Error 错误　　D．错误 Error

5．已经定义好主题 myTheme，并定义了默认皮肤 button.skin，为了使用皮肤，需要在 Web.config 中添加下面哪行代码（　　）。

A．<pages theme="default"/>　　B．<pages theme="myTheme"/>

C．<pages theme="myTheme" skin="button"/>　　D．<%@ page Theme="myTheme" %>

6．在多层开发中使用的数据源是（　　）。

A．ObjectDataSource　　B．SqlDataSource

C．XmlDataSource　　D．StringDataSource

7．下面哪一行代码可以得到当前的日期和时间（　　）。

A．DateTime.Now　　B．DateTime.Today

C．DateTime.DateAndTime　　D．new DateTime()

8．dsStudents 数据集中包含一个名为 students 的表，该表的字段按顺序为：ld，Name，Age。如果要获得第 1 条记录中 Name 字段的值，应该使用以下哪个代码（　　）。

A．dsStudents.Tables("students").Rows(0).Item("Name")

B．dsStudents.students(1).Name

C．dsStudents.Tables[O].Rows[O][1]

D．dsStudents.Fields["Name"]

9．现有 TextBox 控件，需要验证其输入类型为整型，需要使用 CompareValidator 验证控件判断，需要设置 CompareValidator 那个属性（　　）。

A．Operator 属性为 DataTypeCheck，Type 属性为 Integer

B．Operator 属性为 Equal，Type 属性为 Integer

C．Operator 属性为 DataTypeCheck，Type 属性为 Currency

D．Operator 属性为 Equal，Type 属性为 Currency

二、填空题

1．用户控件的文件扩展名是 _____。

2．在 ASP.NET 中所有的自定义用户控件都必须继承自 _____。

3．如果希望控件内容变换后立即回传表单，需要在控件中添加 _____ 属性。

三、简答题

1．试说明 HTML 服务器控件、HTML 控件的区别；HTML 服务器控件与 Web 服务器控件的区别。

2．简要阐述使用服务器验证控件的一般步骤。

3．简述 ASP.NET 服务器控件的生命周期。

4．简要说明 ASP.NET 验证控件中的验证组的概念和应用场合。

四、编程题

1．试创建一个 Windows Application 应用程序：实现具有简单功能（能执行加、减、乘、

除运算）的计算器。

2．创建一个 WebApplication，实现简单的登录验证功能。

（1）要求用集合数据类型预先设定用户名、密码初始值见表 1。

（2）创建一登录界面如图 1 所示，单击【确定】按钮，实现登录验证功能。

表 1　用户名、密码初始集合值

序　　号	用　户　名	密　　码
1	admin	123
2	guest	Sys123

图 1　登录界面

3．要求使用 ASP.NET 验证控件，实现如图 2 所示 Web 页面的验证功能。

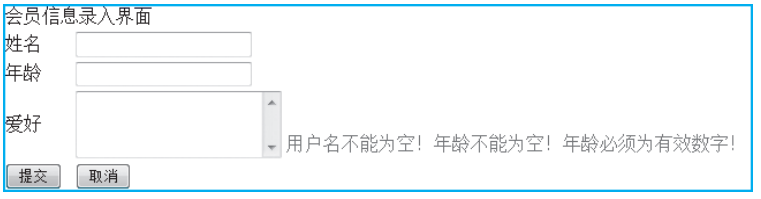


图 2　页面验证功能