第 22 章　注册模块设计

注册模块在网站开发中是一个必不可少的模块，注册模块让用户能够在网站上注册自己的信息，以便在以后的访问中可以直接登录，网站也可以通过注册模块保存用户信息，让用户能够在网站上随时查阅自己的信息和聚合内容。

22.1　学习要点

注册模块需要涉及到一些 ASP.NET 3.5 的基本知识，如果要仔细学习注册模块的开发，需要详细了解本书的一些章节知识，这些章节如下所示：

- ASP.NET 的网页代码模型。
- Web 窗体基本控件。
- 数据库基础。
- ADO.NET 常用对象。
- Web 窗体数据控件。

基本了解了以上章节的知识点后，就能够熟练学习和开发此模块。

22.2　系统设计

在进行系统开发时，无论是模块开发还是整体规划，都需要进行系统设计，系统设计不仅能够方便开发人员的系统开发，同样也节约了在后期维护中所需的时间和成本。系统设计就好像是一张软件制造计划书，通过计划书能够高效的进行软件开发和软件维护。

22.2.1　模块功能描述

注册模块是网站中最常用也是必不可少的模块，对于注册模块的开发，首先需要确定一个基本的用户流程图，如图 22-1 所示。

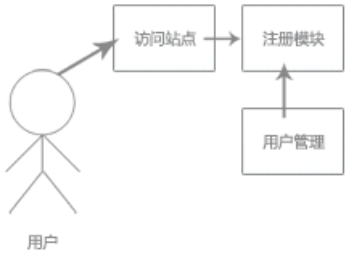

图 22-1　注册模块基本用户流程图

从注册模块的基本用户流程图可以看出，用户进行注册这个动作非常的简单。首先用户需要访问网站，访问网站后就会选择是否进行注册，如果需要注册则网站提供一个注册模块给用户，用户就能够进行注册。在用户完成注册后，用户信息还应该被管理员管理，管理员能够通过用户管理页面进行页面管理。从上述用户流程图可以基本规划以下几个页面：

- 注册页面：提供用用户注册操作。
- 管理页面：提供管理员管理页面。

在基本规划了 Web 应用中需要制作的模块，可以为这些模块进行模块的流程分析。

22.2.2 模块流程分析

在对业务进行了基本的划分之后，可以为模块进行基本的流程分析，包括这个模块中最基本的函数，以及这些函数在页面中是如何执行的。

对于注册页面而言，首先需要确定用户需要提供哪些注册内容，如果 Web 应用希望用户提供真实的信息，例如校内网这样的 SNS，那么就需要用户提供真实的信息，以及提供应用程序验证用户的真实性。如果 Web 应用无所谓用户提供的信息是真实的或者是虚假的，那么就无所谓应用程序的开发，那么应用程序的开发就只需要进行入库即可。

对于管理页面而言，管理人员需要对用户信息进行操作，包括修改和删除。在 ASP.NET 3.5 中，可以使用 SQL 数据源控件和 SQL 数据绑定控件完成功能。既然了解了基本的模块流程和制作，就可以模拟模块流程分析图，如图 22-2 所示。

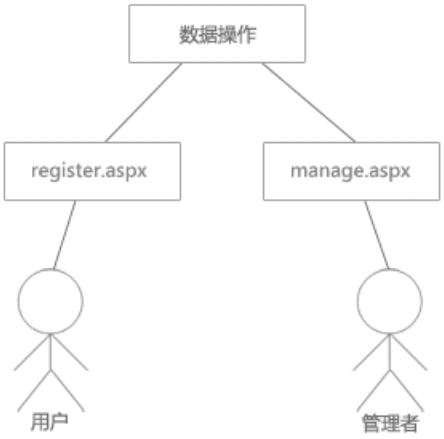

图 22-2　基本模块流程图

用户注册直接进入 register.aspx 页面进行注册，注册完成后进行数据操作，将用户信息加入到数据库中。管理人员进入 manage.aspx 对用户的注册信息管理进行数据操作即可。

22.3　数据库设计

数据库设计是软件设计中最为重要的一部分，当数据库的设计完成后，软件开发过程中如果对于数据库模型的更改则会引起很多的变动，如果对于数据库其中的一个字段的更改，很可能就需要将大部分代码中的 SQL 语句进行更改，良好的数据库设计是非常必要的。

22.3.1 数据库的分析和设计

用户在网站上进行登录，首先要确定对网站而言需要用户的哪些基本信息，这些基本信息可以暂时归纳如下：

- 用户名：用于保存用户的用户名，当用户登录时可以通过用户名验证。
- 密码：用于保存用户的密码，当用户使用登录时可以通过密码验证。
- 性别：用于保存用户的性别。
- 头像：用于保存用户的个性头像。
- QQ/MSN：用于保存用户的 QQ/MSN 等信息。
- 个性签名：用于展现用户的个性签名等资料。
- 备注：用于保存用户的备注信息。
- 用户情况：用于保存用户的状态，可以设置为通过审批和未通过等。

对数据库的基本分析完成后，就可以创建数据库表来存储用户注册的信息。这里需要创建一个 Register 数据库，创建完成后就能够在 Register 数据库中创建表。

22.3.2 数据表的创建

创建表可以通过 SQL Server Management Studio 视图进行创建也可以通过 SQL Server Management Studio 查询使用 SQL 语句进行创建，本书两者都介绍。这个模块的数据库设计比较简单，为了保存用户信息，可以创建一个 Register 表并为数据库分析中的基本信息创建字段，如图 22-3 所示。

列名	数据类型	允许空
id	int	☐
username	nvarchar(50)	☑
password	nvarchar(50)	☑
sex	int	☑
picture	nvarchar(MAX)	☑
IM	nvarchar(50)	☑
information	nvarchar(MAX)	☑
others	nvarchar(MAX)	☑
ifisuser	int	☑
		☐

图 22-3　数据库表结构

正如图 22-3 中所示，表为用户的基本信息创建了字段，这些字段的意义分别为：

- id：用于标识用户的 ID 号，并为自动增长的主键。
- username：用于标识用户名。
- password：用于标识用户密码。
- sex：用于标识用户性别。
- picture：用于标识用户头像。
- IM：用于标识用户的 IM 信息，包括 QQ/MSN 等。
- information：用于标识用户的个性签名。
- others：用于标识用户的备注信息。
- ifisuser：用于标识用户是否为合法用户。

创建数据表的 SQL 查询语句代码如下所示。

```
USE [Register]
GO
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Register](                                                    //创建数据库
[id] [int] IDENTITY(1,1) NOT NULL,
[username] [nvarchar](50) COLLATE Chinese_PRC_CI_AS NULL,
[password] [nvarchar](50) COLLATE Chinese_PRC_CI_AS NULL,
[sex] [int] NULL,
[picture] [nvarchar](max) COLLATE Chinese_PRC_CI_AS NULL,
[IM] [nvarchar](50) COLLATE Chinese_PRC_CI_AS NULL,
[information] [nvarchar](max) COLLATE Chinese_PRC_CI_AS NULL,
[others] [nvarchar](max) COLLATE Chinese_PRC_CI_AS NULL,
[ifisuser] [int] NULL,
 CONSTRAINT [PK_Register] PRIMARY KEY CLUSTERED
(
[id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```

上述代码创建了一个数据库并将 ID 设为自动增长的主键，在用户注册时，可以不向该字段进行数据操作。

22.4 界面设计

良好的界面设计是吸引用户的基本，在注册页面将页面设计的丰富多彩，可以吸引用户的注册和登录，并提高回头率。在进行页面设计时，可以使用 CSS 也可以使用表格进行页面布局，相比之下 CSS 具有更高的灵活性。

22.4.1 基本界面

在进行页面布局前，只需要创建一个基本页面以满足应用程序的需求即可。注册模块需要一些基本的控件，这些控件包括 TextBox 控件、Label 控件和按钮控件，示例代码见光盘中源代码\第 22 章\22-1\22-1\Default.aspx 所示。

上述代码创建了一个头部信息层、一个注册信息层和一个底部信息层，这三个层分别负责头部图片的显示、注册信息的样式控制和底部版权说明，在没有 CSS 控制时，其效果如图 22-4 所示。

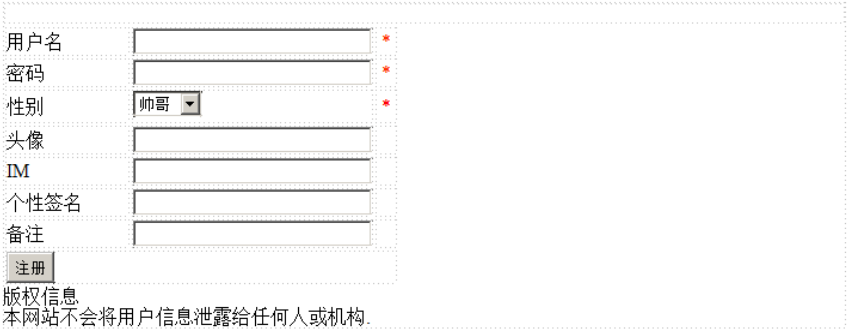

图 22-4　基本样式

在基本样式中，注册信息层使用表格进行排版，使用表格能够快速的进行页面的布局控制，表格同样可以使用 CSS 进行样式控制。

22.4.2　创建 CSS

使用 CSS 进行网页布局能够极大的加强网页布局的灵活度，同样在网页布局中也提高了代码的复用性并将 HTML 页面代码与 CSS 代码相分离，CSS 页面代码如下所示。

```
body                                                    //设置页面样式
{
    font-size:12px;
    font-family:Geneva, Arial, Helvetica, sans-serif;
    margin:0px 0px 0px 0px;
}
.top                                                    //设置头部样式
{
    background:white url(top.png) no-repeat top center;
    height:200px;
    margin:0px auto;
    width:800px;
}
.register                                               //设置注册样式
{
    margin:0px auto;
    width:800px;
}
.end                                                    //设置底部样式
{
    background:#f9fbfd;
    margin:0px auto;
    width:800px;
    text-align:center;
    padding:10px 10px 10px 10px;
}
```

在 CSS 页面文件样式编写完毕后，就需要在相应的页面进行引用，示例代码如下所示。

```
    <link href="css.css" rel="stylesheet" type="text/css" />
```

在使用了 CSS 文件后，页面样式如图 22-5 所示。

图 22-5　CSS 样式控制后的页面

上述页面在 CSS 的样式控制下显得非常的友好，用户在进行注册时，会感觉到应用程序是在用心制作的情况下上线的，提高了用户的回头率。

22.5　代码实现

在完成基本的控件布局和 CSS 样式布局之后，页面就能够呈现在客户端浏览器中。但是如果用户想要在页面中执行逻辑操作，就需要进行代码实现完成应用程序所需要执行的页面逻辑，以保证用户注册功能能够良好的运行。

22.5.1　验证控制

在用户进行注册操作时，需要对用户进行用户验证控制，例如用户没有输入密码的情况下单击了注册控件，数据是不应该被插入到数据库中的。如果没有对数据进行验证则会插入很多空数据，影响数据库功能。若要实现验证控制，可以使用现有的验证控件进行验证控制，示例代码见关盘中源代码\第 22 章\22-1\22-1\Default.aspx。

上述代码使用了 RequiredFieldValidator 控件进行了基本的验证，如果用户输入的用户名和密码以及性别为空，则会说明用户名和密码以及性别为空，请重新输入，如图 22-6 所示。

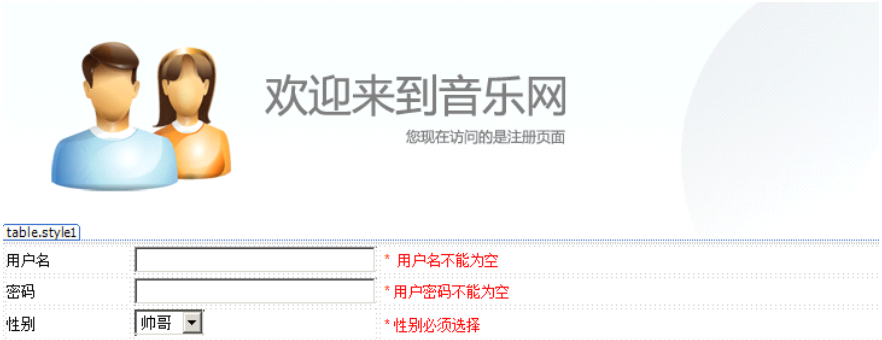

图 22-6　验证控制

进行验证控制后，就能够防止非法用户或用户疏忽所造成的空数据库问题，也方便了数据维护的进行。

22.5.2　过滤输入信息

在进行数据操作之前，并不能只凭用户输入的信息是否为空就能够判断用户是否是合法用户，在 Web 应用中包括很多的不好的信息，例如黄色淫秽名称或者是特殊的字符串，都有可能对网站造成危害。

注意：不仅仅是黄色淫秽的名称会对网站造成危害，特殊的字符串还有可能造成 SQL 注入等更大的危害。

在用户单击按钮控件时会执行数据插入操作，在数据插入之前就需要对信息进行过滤，示例代码如下所示。

```
protected void Button1_Click(object sender, EventArgs e)
{
    if (Check(TextBox1.Text) || Check(TextBox2.Text) || Check(TextBox4.Text) ||
    Check(TextBox5.Text) || Check(TextBox6.Text) || Check(TextBox7.Text))          //判断
    {
        Label8.Text = "用户信息中不能够包含特殊字符如<,>,',//,\\等,请审核";          //输出信息
    }
    else
    {
        //注册代码
    }
}
```

上述代码使用了 Check 函数对文本框控件进行了用户资料的判断，Check 函数的实现如下所示。

```
protected bool Check(string text)                                                  //判断实现
{
    if (text.Contains("<") || text.Contains(">") || text.Contains("'") ||
       text.Contains("//") || text.Contains("\\"))                                 //检查字串
    {
        return true;                                                               //返回真
    }
    else
    {
        return false;                                                              //返回假
    }
}
```

Check 函数定义了基本的判断方式，如果文本框信息中包含“<”，“>”，“'”，“/”，“\”等字符串时，该方法将会返回 true，否则会返回 false。这也就是说，如果字符串中包含了这些字符，则会返回 true。在 Button1_Click 函数中就会判断包含非法字符，并进行提示，否则会执行注册代码。对关键字的过滤是非常必要的，这样能够保证应用程序的完整性并提高应用程序健壮性，同时也对数据库中的完整性进行了保护。

22.5.3 插入注册信息

当用户单击按钮控件时，如果对用户进行了非空验证和关键字过滤后，就能够进行数据的插入，用户可以使用 ADO.NET 进行数据操作，示例代码如下所示。

```
protected void Button1_Click(object sender, EventArgs e)
{
    if (Check(TextBox1.Text) || Check(TextBox2.Text) || Check(TextBox4.Text) ||
       Check(TextBox5.Text) || Check(TextBox6.Text) || Check(TextBox7.Text))          //检查字串
    {
        Label8.Text = "用户信息中不能够包含特殊字符如<,>,',//,\\等,请审核";          //输出信息
    }
    else
    {
        try
        {
            SqlConnection con =
            new SqlConnection("server='(local)';database='Register';uid='sa';pwd='sa'"); //建立连接
            con.Open();                                                                 //打开连接
            string strsql =
            "insert into register (username,password,sex,picture,im,information,others,ifisuser) values
             ('" + TextBox1.Text + "','" + TextBox2.Text + "','" + DropDownList1.Text + "','" +
            TextBox4.Text + "','" + TextBox5.Text + "','" + TextBox6.Text + "','" + TextBox7.Text + "',0)";
            SqlCommand cmd = new SqlCommand(strsql,con);                                //创建执行
            cmd.ExecuteNonQuery();                                                      //执行 SQL
            Label8.Text = "注册成功,请牢记您的信息";                                   //提示成功
        }
        catch
        {
            Label8.Text = "出现错误信息,请返回给管理员";                               //抛出异常
        }
    }
}
```

上述代码通过 ADO.NET 实现了数据的插入，但是上述代码有一个缺点，如果用户注册了一个用户并且名称为 abc，当这个用户注销并再注册一个用户名称为 abc 时，如果依旧将数据插入到数据库则会出现错误。值得注意的是，这个错误并不是逻辑错误，但是这个错误会造成不同的用户可能登录了同一个用户信息并产生信息错误。为了避免这种情况的发生，在用户注册前首先需要执行判断，示例代码如下所示。

```
string check = "select * from register where username='" + TextBox1.Text + "'";
SqlDataAdapter da = new SqlDataAdapter(check,con);                              //创建适配器
DataSet ds = new DataSet();                                                     //创建数据集
da.Fill(ds, "table");                                                           //填充数据集
if (da.Fill(ds, "table") > 0)                                                   //判断同名
{
    Label8.Text = "注册失败,有相同用户名";                                      //输出信息
}
else
{
    SqlCommand cmd = new SqlCommand(strsql, con);                               //创建执行对象
```

```
            cmd.ExecuteNonQuery();                                          //执行 SQL
            Label8.Text = "注册成功,请牢记您的信息";                          //输出成功
        }
```

在用户注册时，首先从数据库查询出是否已经包含这个用户名的信息，如果包含则不允许用户注册，如果没有，则说明用户是一个新用户，可以进行注册。

22.5.4 管理员页面

管理员页面作为管理页面，其功能非常简单，只需要对数据进行删除和修改即可，无需进行任何的数据操作，使用 ASP.NET 本身的数据源控件和数据绑定控件就能够实现管理员页面的编写和制作。作为数据的呈现，可以使用 GridView 控件进行呈现，同时 GridView 控件还支持编辑和删除功能，示例代码见光盘中源代码\第 22 章\22-1\22-1\Manage.aspx 所示。

上述代码编写了 GridView 控件的样式并且为 GridView 控件配置了数据源，同时也配置 GridView 控件能够支持编辑和删除等操作，在数据源配置时，需要新建一个连接字串，如图 22-7 所示。

建立连接字串并保存连接字串到 Web.config 文件中，单击【下一步】按钮，可以生成 SQL 语句，在生成 SQL 语句时，为了方便管理，管理员通常都是对最新注册用户进行管理，如图 22-8 所示。

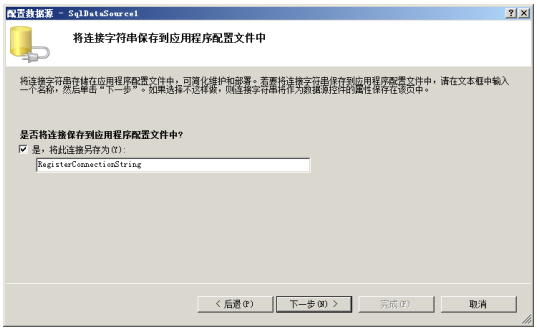

图 22-7 建立连接字串

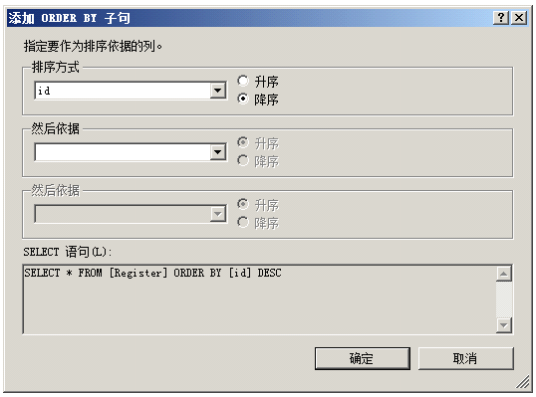

图 22-8 选择排序方式

选择按照 id 的方式进行倒序，能够让管理员快速的管理最新的注册用户，并进行编辑和删除等操作，为了能够让数据源自动支持编辑和删除操作，必须进行数据源高级配置，如图 22-9 所示。

图 22-9 生成数据操作语句

勾选【生成 INSERT、UPDATE 和 DELETE 语句】选项，以支持数据源控件自动进行编辑和删除等操作，单击【确定】按钮并完成，就将数据源控件配置完成，数据源控件配置后代码见光盘中源代码\第 22 章\22-1\22-1\Manage.aspx 所示。

从上述代码可以看出数据源控件中生成的 SQL 语句，使用数据源控件能够简化开发人员对数据的开发。

22.6 实例演示

编写完成页面代码和逻辑代码后，就可以进行注册和管理操作了，单击【运行】按钮运行模块，就能够实现用户的注册操作。用户在注册页面可以填写相应的用户注册信息，这些信息能够保存用户在相应网站上所需要的个人信息，同时网站还能够使用这些信息对用户数据进行归纳和整合，如图 22-10 所示。

用户可以在该页面填写用户信息，并进行注册。用户填写相关选项时，系统都为相关的选项进行了验证和关键字符的过滤，如果用户是一个非法用户，系统通过非法的方法（如 SQL 注入和字符注入）时，就能够验证当前用户注册方法是不正常的方法，就会提示用户注册中所填的项目是错误的。同样如果用户忘记填写相应的选项时，系统也会提示用户必须填写相应的项目，否则不予注册。

这种做法是基于 Web 应用的安全考虑的，不仅提高了网站的健壮性，也让数据库中避免了过多的非法信息，也保证了 Web 应用的其他模块能够正常的运行。当用户注册信息错误时，系统会提示信息错误，如图 22-11 所示。

图 22-10　注册模块运行示例

图 22-11　注册信息异常

如果用户注册成功，开发人员能够控制用户跳转到登录页面，登录页面会在后面的模块中讲解。对于 Web 应用而言，需要对现有的用户信息进行管理，所以管理员能够在后台管理页面中管理相应的用户，进行编辑和删除等操作，如图 22-12 所示。

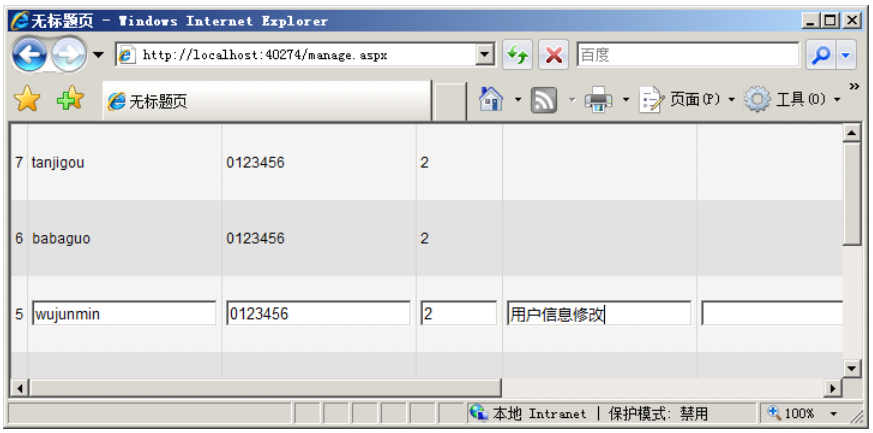

图 22-12　编辑相应用户

管理员可以在管理页面对用户数据库中的相应字段进行修改和增加，如果用户忘记了自己的密码，也可以通过联系管理员获取和修改自己的密码，对于恶意注册的用户，管理员可以轻易的删除相应的注册用户。

22.7　小结

本章介绍了注册模块的开发流程和核心代码，注册模块是网站应用中非常重要的模块，通过注册模块能够实现网站和用户的信息交流，本章从案例分析，数据库设计到代码编写都进行了讲解。本章还巩固了：

- ASP.NET 的网页代码模型。
- Web 窗体基本控件。
- 数据库基础。
- ADO.NET 常用对象。
- Web 窗体数据控件。

通过本章的学习能够巩固和强化对本书中的这些章节的理解。