第 5 章　数据绑定与高级编码

本章要点：

- 录入数据方法与编码
- 查询数据方法与编码
- 删除数据方法与编码
- 修改数据方法与编码

技能目标：

- 会编码获取 GridView 单元格的数据
- 会编码实现 GridView 基于单元格的更新
- 会编码实现 GridView 中的数据删除
- 会编码处理 GridView 的常用事件

5.1　项 目 导 入

【项目场景】

太仓某公司要开发一个小型系统，能够实现该公司信息的增删改查功能，运行结果如图 5.1～图 5.3 所示，请你为该公司开发该系统，实现相应功能。

图 5.1　添加公司信息

公司信息一览表

公司名称	Email	联系人	电话	删除
新科技有限公司	mm@126.com	高女士	0512-6972266	删除
奇奇科技有限公司	ms@163.com	张女士	0512-6972266	删除
新世纪集团	mm@163.com	李小丽	0512-6972266	删除
家化集团	mq@163.com	杨柳	0512-6978981	删除
明天公司	yu@163.com	李小同	0512-6978981	删除

图 5.2　查询、删除公司信息

公司信息一览表

公司名称	Email	联系人	电话		删除
新科技有限公司	my@126.com	gao女士	0512-6972266	更新 取消	删除
奇奇科技有限公司	ms@163.com	张女士	0512-6972266	编辑	删除
新世纪集团	mm@163.com	李小丽	0512-6972266	编辑	删除
家化集团	mq@163.com	杨柳	0512-6978981	编辑	删除

图 5.3　修改公司信息

【问题引导】

（1）如何添加数据。
（2）如何查询数据。
（3）如何删除数据。
（4）如何修改数据。

5.2　技术与知识准备

5.2.1　录入数据方法与编码

【示例 5.1】　新建一个学生管理系统，能够实现学生信息的添加功能，如图 5.4 所示。

【步骤 1】搭建系统框架，添加各层之间的依赖关系，如图 5.5 所示。

图 5.4　添加学生信息

图 5.5　搭建系统框架

【步骤 2】创建数据库 dbStu，添加数据表 tbStuInfos，如图 5.6 所示，并配置 Web.config：＜connectionStrings＞

```
    <add name =" DefaultConnection " providerName =" System. Data. SqlClient "
    connectionString="Data Source=.;Initial Catalog=dbStu;Integrated Security
    =True" />
  </connectionStrings>
```

图 5.6　学生信息表

【步骤 3】根据数据表，在实体层添加类 tbStuInfo.cs，编写代码如下：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace StuModel
{
  public class tbStuInfo
    {
        private string sno;
        public string Sno
        {
            get { return sno; }
            set { sno=value; }
        }
        private string sname;
        public string Sname
        {
            get { return sname; }
            set { sname=value; }
        }
        private string sex;
        public string Sex
        {
            get { return sex; }
            set { sex=value; }
        }
```

```
        private string zhuanYe;
        public string ZhuanYe
        {
            get { return zhuanYe; }
            set { zhuanYe=value; }
        }
        private string jiGuan;
        public string JiGuan
        {
            get { return jiGuan; }
            set { jiGuan=value; }
        }
    }
}
```

【步骤 4】添加数据访问类 DBHelper.cs，并编写代码，见第 3 章内容。

【步骤 5】数据访问层添加类 tbStuInfoService.cs，编写代码如下：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using StuModel;
using System.Data;

namespace StuDAL
{
   public class tbStuInfoService
    {
       public bool AddStuInfo(tbStuInfo tbstu)
       {
           string sql = string.Format("insert into tbStuInfos(Sno,Sname,Sex,
           ZhuanYe,JiGuan)values('{0}','{1}','{2}','{3}','{4}')", tbstu.Sno,
           tbstu.Sname, tbstu.Sex, tbstu.ZhuanYe, tbstu.JiGuan);
           return DBHelper.ExecuteNonQuery(DBHelper.ConnectionString,
           CommandType.Text, sql)>0;
       }
    }
}
```

【步骤 6】业务逻辑层编写代码如下：

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;
using StuModel;
using StuDAL;

namespace StuBLL
{
    public class tbStuInfoManager
    {
        public bool AddStuInfo(tbStuInfo tbstu)
        {
            return new tbStuInfoService().AddStuInfo(tbstu);
        }
    }
}
```

【步骤 7】右击 StuMWeb，添加页面 Reg.aspx，源视图代码如下：

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Reg.aspx.cs" Inherits
="Reg" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
    <table>
        <tr><td colspan="2"  style="text-align:center; font-size:large;
        color:red">添加学生信息</td></tr>
         <tr><td>学号</td><td>
             <asp:TextBox ID="txtSno" runat="server"></asp:TextBox></td></tr>
         <tr><td>姓名</td><td><asp:TextBox ID="txtSname" runat="server">
         </asp:TextBox></td></tr>
         <tr><td>性别</td><td><asp:TextBox ID="txtSex" runat="server"></asp:
         TextBox></td></tr>
         <tr><td>专业</td><td><asp:TextBox ID="txtZhanYe" runat="server">
         </asp:TextBox></td></tr>
          <tr><td>籍贯</td><td><asp:TextBox ID="txtJiGuan" runat="server">
         </asp:TextBox></td></tr>
          <tr><td colspan="2" style="text-align:center">
              <asp:Button ID="Button1" runat="server" Text="添加" /></td></tr>
    </table>
```

```
    </div>
    </form>
</body>
</html>
```

【步骤 8】表示层后台代码如下：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using StuBLL;
using StuModel;
public partial class Reg : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        tbStuInfo tbs=new tbStuInfo();
        tbs.Sno=txtSno.Text;
        tbs.Sname=txtSname.Text;
        tbs.Sex=txtSex.Text;
        tbs.ZhuanYe=txtZhanYe.Text;
        tbs.JiGuan=txtJiGuan.Text;
        if(new tbStuInfoManager().AddStuInfo(tbs))
            Response.Write("<script>alert('添加成功')</script>");
        else
            Response.Write("<script>alert('添加失败')</script>");
    }
}
```

5.2.2　查询数据方法与编码

ASP. NET 中有以下两种数据绑定方式：

1. 编码指定数据源

所谓的指定数据源的方式，就是编写代码在程序运行中动态绑定数据源，例如：

```
GridView1.DataSource=new UserManager().GetUsers();
//业务逻辑层已编写方法 GetUsers(),获取所有用户
GridView1.DataBind();
```

GridView1 就是数据绑定控件 GridView。

2. 使用数据源控件

数据源控件用于实现从不同数据源(数据库、XML 文件或业务逻辑层对象)获取数据的功能,它可以设置连接信息、查询信息、参数和行为,这样就可以把指定的数据绑定到数据绑定控件。常见的数据源控件有 SqlDataSource,XMLDataSource 等。

常见的数据绑定控件见表 5.1 所示。

表 5.1 常见的数据绑定控件

控件名称	用途
DropDownList	下拉列表框,可以使用下拉菜单的形式供用户选择
GridView	通过表格方式实现数据的展示,其中每列表示一个字段,每行表示一条记录

【示例 5.2】 用编码指定数据源方式实现姓名模糊查询功能,见图 5.7 所示。

图 5.7 查询学生信息

【步骤 1】数据访问层定义方法,实现数据查询功能,代码如下所示:

```
public List < tbStuInfo > GetStuBySname (string
name)
      {
          string sql=string.Format("select * from tbStuInfos where Sname like
          '%{0}%'", name);
          SqlDataReader dr=DBHelper.ExecuteReader(DBHelper.ConnectionString,
          CommandType.Text, sql);
          List<tbStuInfo>list=new List<tbStuInfo>();
          while(dr.Read())
          {
              tbStuInfo tbf=new tbStuInfo();
              tbf.Sno=Convert.ToString(dr["Sno"]);
              tbf.Sname=Convert.ToString(dr["Sname"]);
              tbf.Sex=Convert.ToString(dr["Sex"]);
              tbf.ZhuanYe=Convert.ToString(dr["ZhuanYe"]);
              tbf.JiGuan=Convert.ToString(dr["JiGuan"]);
              list.Add(tbf);
          }
          dr.Close();
          return list;
      }
```

【步骤 2】业务逻辑层编写代码如下所示:

```
public List<tbStuInfo>GetStuBySname(string name)
      {
```

```
        return new tbStuInfoService().GetStuBySname(name);
    }
```

【步骤 3】右击 StuMWeb，添加 StuCX.aspx 页面，添加一个文本框(TextBox)，一个按钮(Button)，一个数据绑定控件(GridView)，页面如图 5.8 所示，源视图代码如下所示：

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="StuCX.aspx.cs"
Inherits="StuCX" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
    <table><tr><td colspan="2">学生信息查询</td></tr>
        <tr><td colspan="2">请输入学生姓名(可实现模糊查询)</td></tr>
         <tr><td><asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
        </td><td>
            <asp:Button ID="btnCX" runat="server" Text="查询" />
           </td></tr>
         <tr><td colspan="2">
            <asp:GridView ID="GridView1" runat="server"></asp:GridView>
         </td></tr>
    </table>
    </div>
    </form>
</body>
</html>
```

图 5.8　学生信息查询页面

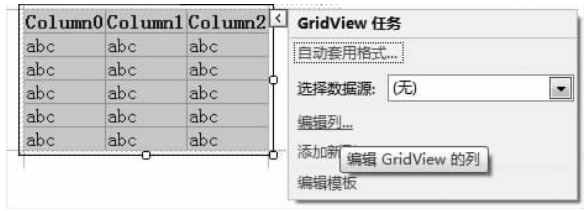

图 5.9　选中【编辑列...】命令

【步骤 4】选中 GridView，单击右上角的方块，弹出如图 5.9 所示的窗口，选中编辑列命令，弹出如图 5.10 所示的对话框。

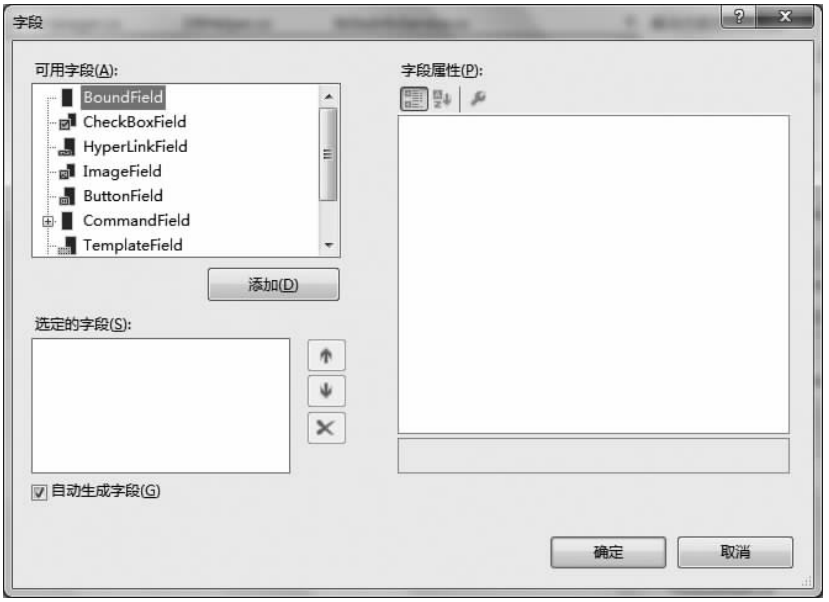

图 5.10 【字段】对话框

【步骤 5】添加 BoundField 字段，在 DataField 属性中设置显示的字段，HeaderText 显示列标题，如图 5.11 所示。这时会发现系统又为我们重复显示了一遍数据，如图 5.12 所示，解决方法是将 GridView 控件的 AutoGenerateColumns 属性设置为 False 即可。

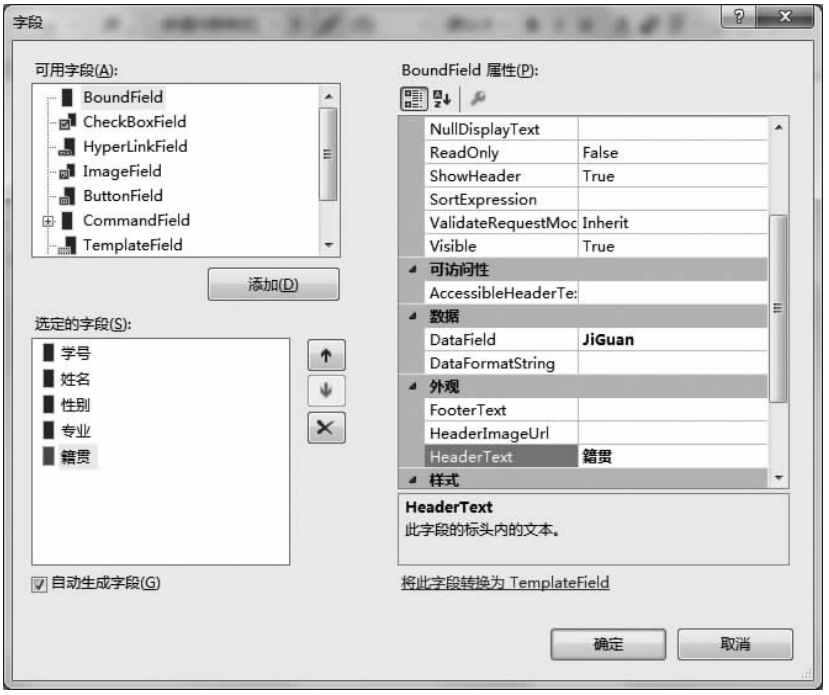

图 5.11 添加 BoundField 字段

AutoGenerateColumns 属性表示是否为数据源中的每一字段自动创建 BoundColumn 对象并在 GridView 控件中显示这些对象。

图 5.12　添加 BoundField 字段页面

【步骤 6】双击查询按钮，产生 Click 事件，编写代码如下：

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using StuBLL;
using StuModel;

public partial class StuCX : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }
    protected void btnCX_Click(object sender, EventArgs e)
    {
      GridView1. DataSource = new  tbStuInfoManager ( ). GetStuBySname (TextBox2.
      Text);
        GridView1.DataBind();
    }
}
```

5.2.3　删除数据方法与编码

我们通过单击 GridView 控件中的删除按钮，利用编码删除后台数据库中的相关数据。

【示例 5.3】　添加删除列，实现删除功能，如图 5.13 所示。

【步骤 1】选中 GridView，单击右上角的方块，在弹出的窗口中选中【编辑列...】命令，弹出【字段】对话框，添加 TemplateField 字段，并设置 HeaderText 属性为删除，如图 5.14 所示。