数据绑定与导航在 Visual C#. Net 下的实现及应用

Study and Implementation of Data Binding and Navigation in Visual C#. Net

李林静 叶冬芬 蒋晓丹 (衢州学院 信息与电子工程系 浙江 衢州 324000)

摘 要：本文主要介绍了在 Visual C#. Net 中如何利用数据集访问模式提供的数据绑定与导航技术实现数据库中数据的显示。文中先总结了在. NET 平台下利用 Visual C#. Net 实现数据绑定与导航的方法,然后在 VS. NET 环境中通过创建 C# Windows 数据库应用程序实现了数据绑定与导航,对实现这种技术的关键问题进行了分析,最后将这种技术应用到数据维护和一对多的数据关系中。

关键词：NET 平台 数据绑定 数据导航 数据集访问模式 C#

1 引言

数据绑定与导航是面向对象程序设计中非常重要的技术。数据库应用程序最常见的功能就是从数据库中提取数据并显示在屏幕上,使用户能够方便地通过 Windows 控件来浏览、处理和修改数据,并把对数据的修改保存到数据库中,而要实现这一功能,就需要在程序中实现数据绑定与导航。

在. NET 框架下,数据集访问模式提供了 WinForm 窗体数据绑定结构,允许用户在记录间导航,支持对从数据库获取的每一条记录进行处理操作[1]。数据集访问模式是 ADO. NET 提供数据非连接访问的一种方法,该方法在 DataSet 断开与数据源的连接后,数据绑定与导航实现对加载到数据集对象中的数据进行处理,实现数据记录与可编辑控件之间的绑定,做到让数据库表中的数据在文本框、组合框、数据网格中随着记录指针移动而相应发生变化,这种方法不仅可以实现对数据表中数据的浏览,还可以实现对数据表数据的维护。

2 数据绑定及其分类

数据绑定是在数据源和控件之间建立一个同步、双向、互动的联系,以便更方便地访问和维护数据[1]。在 Windows 程序中,按控件绑定数据元素的能力分为以下三种[2,3]:

(1)一次只能在该控件上绑定一条记录中的某个字段的内容,实现这类控件如 Label、TextBox 、Button、CheckBox、RadioButton 等。C#为了实现动态绑定,专门开发了 Binding 类来负责创建和维护简单绑定。该类的构造函数如下:

Public Binding (string proname, object ds, string datamember);

借助该 Binding 类实现这类控件绑定的语法如下:

控件对象名称. DataBindings. Add (new Binding ("要绑定控件属性",数据源,"绑定的数据源字段"));

(2)在该控件上显示所有记录的某一个字段数据,实现这类控件如 ComboBox、ListBox,数据绑定的语法如下:

控件对象名称. DataSource = 数据源;

控件对象名称. DisplayMember = "数据源字段";

(3)在该控件上显示所有记录的所有字段,如 DataGrid,数据绑定的语法如下:

控件对象名称. DataSource = 数据源;

控件对象名称. DataMember = " 表格;

其中数据源可以是 DataSet、DataView、DataTable 等。

3 数据导航

实现数据表中前后数据记录的移动或将记录指针移到指定的位置,称为数据导航。在 Visual C#.

NET 中，ADO. NET 提供 BindingManagerBase 类来实现数据导航，保证数据的同步。它控制的数据源有个当前行的概念，控件一旦跟数据源绑定后，DataGrid 将显示数据源表的所有数据，用　　来指示在 DataGrid 中当前行的位置。简单绑定控件中显示的值将是数据源当前行的内容[4]。但该类是 CurrencyManager 类的基类，是一个抽象类，无法实例化，在程序中使用窗体的 BindingContext 方法可以建立 BindingManagerBase 对象，其中 BindingContext 方法必须传入数据源及数据成员，建立 BindingManagerBase 对象的语法如下：

mybm = this. BindingContext[数据源,"数据成员"];

BindingManagerBase 对象在实现数据导航时提供两个重要的属性 Count 和 Position 和事件 PositionChanged。其中 Count 记录所管理绑定的记录总数，Position 获取或设置绑定到该数据源的控件所指向的列表的位置，取值从 0 ~ Count - 1，对于 PositionChanged 事件在 position 位置更改时触发。

从上可以看出，在实现数据绑定时都需要传入数据源参数，指定该数据源内的数据成员。故在数据库应用程序中实现绑定与导航一般应遵循以下几个步骤：

(1) 实现对数据库的连接

(2) 建立数据适配器对象，填充数据集对象

(3) 对加载到数据集对象中的数据利用控件的数据绑定方法实施数据绑定与导航

下面按上述几个步骤，探讨数据绑定与导航实现的过程。

4　数据绑定与导航在 Visual C#. Net 下的实现

下面在 vs. net 中，创建基于 C#的 Windows 应用程序项目来实现数据绑定与导航。采用 SQL Server. NET 数据提供程序，利用数据集实现对数据库 xsgl 中的 student 表的访问。

4.1　数据绑定实现的效果

对 student 表访问的效果如图 1 和图 2 所示，当选图 1 中“学号”下拉列表框的学号 05010103 时，会将该条记录对应的字段内容分别放入姓名、性别、地址、入学成绩的文本框内，实现对文本框控件的绑定，同时 DataGrid 控件的记录位置也指示到学号 = 05010103 的位置。同样地，在图 2 中，选择 DataGrid 中如学号 = 05010104 记录，同时在学号下拉列表框，姓名、性别、地址、入学成绩的文本框内会显示学号 = 05010104 记录的字段内容，上述两种方式都实现了 DataGrid 控件浏览的记录与学号下拉列表框，4 个文本框显示内容的同步。这为数据的维护奠定了基础。

图 1　下拉列表框实现 DataGrid 与文本框显示内容的同步

图 2　选择 DataGrid 控件中任何一条记录实现 DataGrid 与文本框显示内容的同步

4.2　数据导航实现的效果

在图 1 中，当选择“下一条”按钮，浏览的效果如图 3 所示，DataGrid 控件浏览记录的位置指向学号 = 05010104 的记录，同时学号下拉列表框，姓名、性别、地址、入学成绩的文本框内显示的内容也是学号 = 05010104 记录的内容，通过数据导航实现了 DataGrid 与文本框显示内容的同步。

图 3　通过数据导航实现对数据的绑定

4.3 数据绑定与导航实现的关键技术

(1) 在学生教学管理 xsgl 数据库中增加学生入学基本信息表 (student), 该表主要包括学号、姓名、性别、籍贯和入学成绩字段, 在 SQL Server2000 中定义的结构如图 4 所示:

图 4　student 表的结构

(2) 定义一个全局的 BindingManagerBase 对象 bm

```
private BindingManagerBase bm;
```

(3) 在 Form1_Load 事件内添加的主要代码为:

```
//建立连接 xsgl 数据库代码
selectcmd = " select studID as 学号, studname as 姓名, studsex as 性别, studaddress as 地址, enterscore as 入学成绩 from student" ;
SqlDataAdapter adp = new SqlDataAdapter( selectcmd, conn) ;
DataSet ds = new DataSet( ) ; adp. Fill( ds," student" ) ;
//下面这 2 行代码实现对 DataGrid 控件的绑定
    dataGrid1. DataSource = ds;
    dataGrid1. DataMember = " student" ;
//下面这 2 行代码实现对组合框的绑定
    cbold. DataSource = ds;
    cbold. DisplayMember = " student. 学号" ;
//下面这 8 行代码实现对文本框的绑定
Binding bindname = new Binding ( " Text" , ds," student. 姓名" ) ;
Binding bindsex = new Binding ( " Text" , ds," student. 性别" ) ;
Binding bindaddress = new Binding ( " Text" , ds," student. 地址" ) ;
Binding bindscore = new Binding( " Text" , ds," student. 入学成绩" ) ;
    txtName. DataBindings. Add( bindname) ;
    txtSex. DataBindings. Add( bindsex) ;
    txtAddress. DataBindings. Add( bindaddress) ;
    txtScore. DataBindings. Add( bindscore) ;
    bm = this. BindingContext[ ds," student" ] ; //建立 bm 对象
    checkbm( ) ; //调用自定义 checkbm( ) 过程
```

在 Form1_Load 事件内, 主要完成对数据库 xsgl 的连接, 用数据适配器对象 adp 对数据集对象 ds 的填充, 生成了 "student" DataTable 对象, 从而产生了 ds 数据源, 接着使用数据绑定方法完成 dataGrid1、comboBox1 和 4 个文本框对象的数据绑定, 实现 DataGrid 控件指示记录位置与文本框显示内容的同步。另外建立 BindingManagerBase 对象 bm, 通过调用自定义 checkbm( ) 过程来设置 4 个按钮导航条位置的状态以及判断当前记录所在的位置。

(4) checkbm( ) 是自定义过程, 根据获取绑定到该数据源 ds 所指向的 student 的位置 Position 的值来判断记录的位置, 若 bm. Position = =0, 此时指向第一条数据记录, 若 bm. Position = = bm. Count -1, 此时指向最后一条数据记录, 所以用 4 个按钮来实现记录的定位。实现的主要代码为:

```
this. buttoncurrent. Text = string. Format( " {0} of {1} " , ( bm. Position +1) , bm. Count) ;
```

(5) "首记录" 导航条按钮实现的功能, 其代码:

```
bm. Position =0; checkbm( ) ;
```

(6)“上一条”导航条按钮实现的功能,其代码:

```
if(bm.Position>0) bm.Position--;
checkbm();
```

(7)“下一条”导航条按钮实现的功能,其代码:

```
if(bm.Position<bm.Count-1) bm.Position++;
checkbm();
```

(8)“末记录”导航条按钮实现的功能,其代码:

```
bm.Position=bm.Count-1; checkbm();
```

5 数据绑定与导航在数据维护中应用

上面示例通过 C#提供的类和方法实现了对数据表中数据记录的绑定和导航,在此基础上就可以实现对数据表中记录的添加、修改和删除等数据维护操作。下面对数据库 xsgl 中的 score 表的访问如图 5 所示,通过数据绑定与导航实现了 DataGrid 与文本框显示内容的同步[5]。当单击“修改”按钮,成绩文本框处于编辑状态,可以实现对学号 =05010101 同学所选修的课程号 =050103 的成绩进行修改,修改后,点击“保存”,以替换原有记录的内容。若误操作可以点击“取消”按钮,以取消当前操作。

若要添加某位同学选修某门课程的成绩,可以通过点击“添加”按钮,使三个文本框置空,操作完成后,点击“保存”,向该数据库添加一条记录。

图 5　借助绑定和导航实现对 score 表的维护

若要删除某位同学选修某门课程的成绩,首先通过导航条按钮定位到该记录,或通过选择 DataGrid 控件定位到该记录的位置,再点击“删除”按钮,由刚才的动作中获取要删除行的的位置,从该数据表中删除一条记录,实现删除的关键代码:

```
dsScore.Tables["score"].Rows[this.BindingContext[dsScore,"score"].Position].Delete();
sqlDataAdapter.Update(dsScore,"score");
```

在该应用中,主要通过建立数据集对象,利用数据适配器填充数据集,再采用数据绑定方法将文本框控件与数据源的各个字段绑定,数据网格 DataGrid 与数据源的绑定。添加、修改及删除记录时,先对数据集进行操作,再利用数据适配器的 Update()方法将数据保存到数据库。移动记录使用了数据导航技术。

6 数据绑定与导航在一对多关系中的应用

一对多关系是指在数据库中一个表的一项对应着另一个表的多项,这两个表之间的关联是通过它们的相同字段来联系的。在数据库应用中,存在许多一对多关系的数据源,比如学生表 student 和所修课程成绩 score 的数据关系,一个学生可以选修多门课程,获得多门成绩,这就形成了一个学生对应多门成绩的一对多的数据关系,其 student 的结构如图 4,在 xsgl 数据库中增加学生成绩表 score,该表主要包括学号、所修课程的课程号、所修课程所获得的成绩,在 SQL Server2000 中定义的结构如图 6 所示。为了实现这种数据关系的显示,ADO.NET 提供了与 DataSet 有关的对象 DataRelation,该 DataRelation 的主要功能就是允许你在一个 DataSet 中从一个 DataTable 导航至另一个 DataTable,在给定相关 DataTable 中的单个 DataRow 的情况下检索一个 DataTable 中所有相关 DataRow 对象。例如,当建立学生表和成绩表之间的 DataRelation 后,可以使用 DataRow.GetChildRows 检索特定学生所修课程的成绩。

下面对 xsgl 数据库访问,浏览学生成绩的效果如图 7 所示,在图 7 中,可以上下翻页,查看学生的学号和姓名,同时显示该学生所修课程的成绩。

图 6　score 表的结构

图 7 浏览学生成绩

在该应用中，通过建立一个数据集 ds，**用数据适配器向** ds **中填入两个表**：student **和** score，**同时以** studID **为关联字段，在数据集中创建** student **表和** score **表之间的** DataRelation，**当** student **中记录位置发生改变时，根据建立的关系使用** DataRow. GetChildRows **检索该学生所修课程的成绩，并把检索的结果重新绑定到** DataGrid **中，实现上述效果的关键技术**：

（1）**用数据适配器向** ds **中填入两个表**：student **和** score

（2）**建立**“student”**表中学号与姓名字段的绑定**

（3）**根据** studID **字段建立** student **与** score **表之间的关系** studscorel

```
studscorel = ds. Relations. Add( " studentscore" , ds. Tables[ " student" ]. Columns[ " studID" ] , ds. Tables[ " score" ]. Columns[ " studID" ] ) ;
```

（4）**建立**“student”**的** PositionChanged **事件的绑定**

```
this. BindingContext [ ds," student" ]. PositionChanged + = new EventHandler( BindingManagerBase_positionchanged) ;
```

（5）BindingManagerBase_positionchanged **事件实现的主要功能是**：**根据建立的关系** studscorel，**使用** DataRow. GetChildRows **检索该学生所修课程的成绩，并把检索的结果重新放到** ds1 **中。其代码**：

```
ds1. Tables[ " score" ]. Rows. Clear( ) ;
DataRow
dr = ds. Tables[ " student" ]. Rows[ this. BindingContext [ ds," student" ]. Position] ;
foreach( DataRow dr1 in dr. GetChildRows( studscorel) )
    { DataRow mydr = ds1. Tables[ " score" ]. NewRow( ) ;
        mydr[ " studID" ] = dr1[ " studID" ] ;
        mydr[ " courseID" ] = dr1[ " courseID" ] ;
        mydr[ " score" ] = dr1[ " score" ] ;
        ds1. Tables[ " score" ]. Rows. Add( mydr) ; }
```

（6）**上一页导航**：this. BindingContext [ds," student"]. Position - =1;

（7）**下一页导航**：this. BindingContext [ds," student"]. Position + =1;

7 结束语

本文对数据绑定与导航技术进行了分析，总结了利用这种技术在 Windows **窗体中向用户显示数据的方法，并用** Visual C#. Net **实现了这种技术，对其中的关键技术进行了分析，并把这种技术应用到数据维护和一对多数据关系中。在开发面向对象的数据库应用程序中，如何合理利用** WinForm **控件的数据绑定与导航，为快速开发便捷高效的数据库应用程序提供了思路**。

参考文献

1 杨树林，胡洁萍. C#程序设计与案例教程. 北京：清华大学出版社，2007：191 - 200.

2 曹祖圣，吴明哲，黄世阳，林义证. Visual c#. NET 程序设计经典. 北京：科学出版社，2006：382 - 397.

3 李岚，朱红高. 基于 C#的 ADO. NET 访问数据库技术. 电脑知识与技术（学术交流），2007，2（7）：28 - 29.

4 魏然，李新磊. ADO. NET 数据绑定及更新技术的研究. 科技信息（学术版），2007，（7）：159 - 160.

5 周世雄. NET 经典范例教程. 北京：清华大学出版社，2004：265 - 290