CHAPTER 5

第5章 Database类的建立及程序的跟踪调试

任务：

（1）建立 Database 类。

（2）获取数据库表中的数据并加以显示。

（3）跟踪调试代码。

（4）完成练习。

5.1 概　　述

5.1.1 C#中的自定义类及 Database 类

在先前的实验中，没有必要去定义一个类，因为在创建 Form 窗体时，Visual Studio 自动创建了一个相应的类，只需用户在自动创建的类中填写所用的代码就可以。而在实际应用中，需要自己创建功能类的情形也很多，比如做数据库应用，要对数据库进行增删改查等操作，而这些操作在创建的窗体界面上比比皆是。因而，我们设想有这么一个功能类，把对数据库表进行增删改查的操作都实现了，能够提供各种增删改查的方法，当需要的时候，就用该类中所提供的方法即可。那么，Database 类实际上就是刚刚说的这种功能类。

Database 类是一个采用 C++ 语言编写的用于处理数据库增删改查等操作的通用类。C++ 为操作数据库提供了基本的 API 支持，如创建连接、执行查询、执行修改等，它们大多用到了操作数据库的 SQL 语句作为字符串参数，在此基础上，通过 C++ 编程，定义并创建了处理 SQL 增删改查等操作的 Database 通用类。

以下是 Database 类实现的主要方法。

```
public Database()
    //Database()的构造函数,将所要连接的数据库赋予相应变量
public void Open()        //根据连接的数据库的变量值,进行连接
public void Close()       //关闭数据库连接
public int ExecuteSql(string sqlString)
    //根据 sqlString 所表示的 SQL 操作进行增删改查
```

```
public int TranExecuteSql(string[] sqlStringList)
    //根据 sqlStringList 字符串数组中所表示的各个 SQL 操作进行事务处理
```

本章给出关于 Database 的实验，包含两个实例，一个是在应用中创建 C++ 的类，如 Database 类，该 Database 类的具体内容参见附录 C“Database 类代码”，另一实例则是基于 Database 类的支持，实现“获取数据库表中的数据并加以显示”的操作响应。

5.1.2 程序调试的意义及其跟踪调试

调试的最初目的就是解决问题。程序调试的英文名是 Debug，其中，bug 为程序中存在的问题，Debug 就是解决这些问题。程序调试可以明确看到程序的执行过程以及每一步发生的变化，这样我们可以直观地体验每一步代码的运行结果，相比于想象代码运行的变量变化过程简单得多，有利于理解代码逻辑、检查语法错误等。当程序的运行结果与期望不符时，可以通过调试定位出哪步操作或者语句与预期结果不符，从而快速定位错误之处，再有针对性地分析代码，实现快速解决问题的目的。

调试是程序修正语法错误和逻辑错误的过程，是检验代码的正确性和有效性的必不可少的步骤。跟踪调试是检查和排除程序中存在错误的重要步骤，通过调试可以获取很多有用的信息，例如代码运行顺序、变量值的变化等。

在调试中程序的每一步的执行都是可控的，可以通过单步执行、设置断点等控制程序的运行，并且在每一步的执行之后可以查看当前有效变量的属性值。

Visual Studio 的调试器是查找、定位、排除程序错误的有效工具，开发人员既可以查看应用程序执行到某条语句时的变量以及对象属性的取值，也可以修改其值。程序既可以单步执行，也可以断点执行；断点可以设置在某条可执行语句上，也可以将变量断点设置在变量取值变化的语句上，暂停程序的运行。

本章的实验三中，设定了一个在学生表中插入一行数据的场景，有插入成功的样例，也给出插入失败的样例，此时，需要知道为什么失败了，就采用设置断点跟踪的方法来确定失败的根源。

特别提示：在附录中所提供的 Database 类代码，其构造函数中所表示的连接数据库，需要根据实际情况进行修改，学生可以根据自己的数据库名字以及所存放的目录位置，把代码中的相应内容替换掉，否则连接不会成功。

5.2 实　　验

实验一　创建 Database 类

(1) 给项目添加 Database 类。

在 Solution Explorer 视图中，右击 Stucour 项目，选择 Add→Class 命令，如图 5.1 所示。

(2) 在 Name 处输入“Database.cs”，如图 5.2 所示，然后单击 Add 按钮。

(3) 将 VS 2010 自动生成的代码全部删除(如图 5.3 所示，深色代码全部删除)。

(4) 将附录 C“Database 类代码”输入到 Database.cs 文件中，然后按 Ctrl＋S 组合键保存修改。

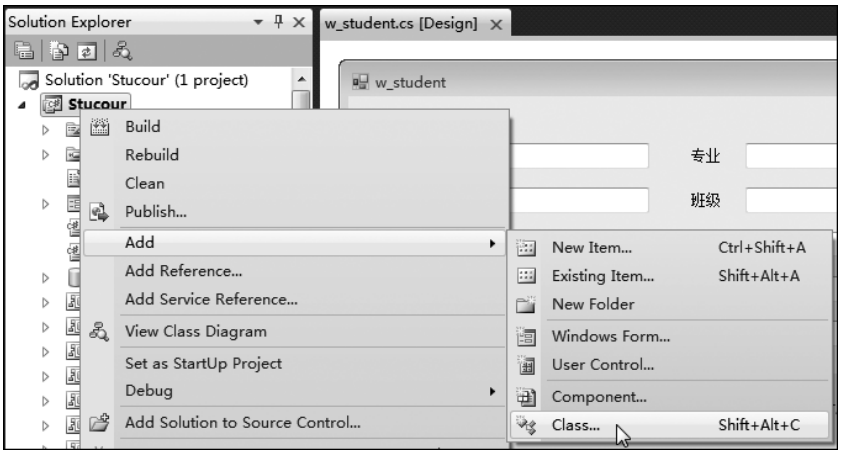

图 5.1　Class 选项界面示意图

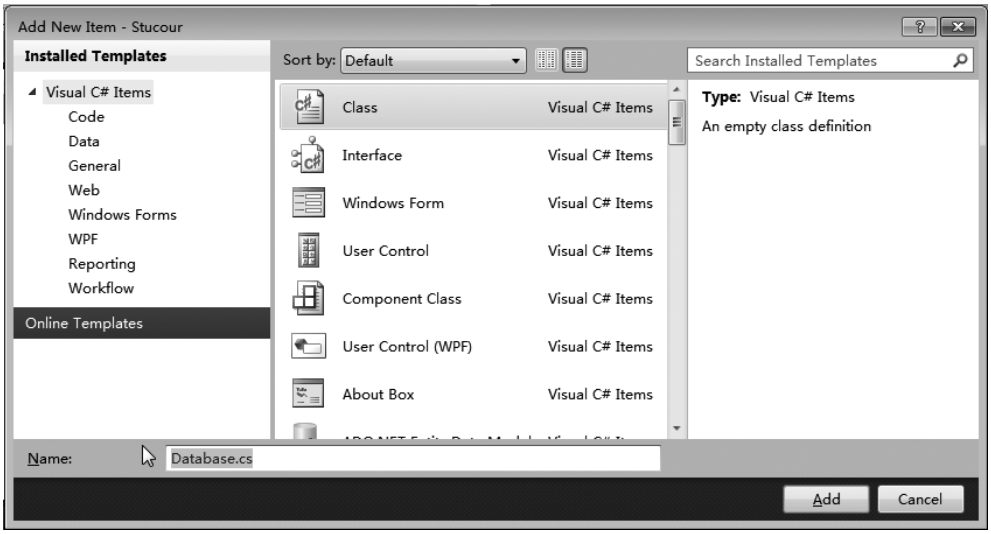

图 5.2　添加 Database.cs

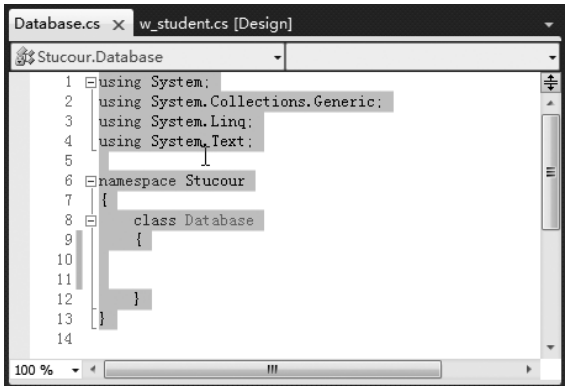

图 5.3　删除代码界面示意图

实验二　获取数据库表中的数据并加以显示

（1）以 w_student 窗体为例，实现 DataGridView 的选中行改变操作，即 DataGridView 被选择的行发生改变时触发的动作。将选择的行的数据填充到窗体上去。

首先打开 w_student 窗体，单击 DataGridView，然后在 Properties 视图中找到 Events 按钮单击，如图 5.4 所示。

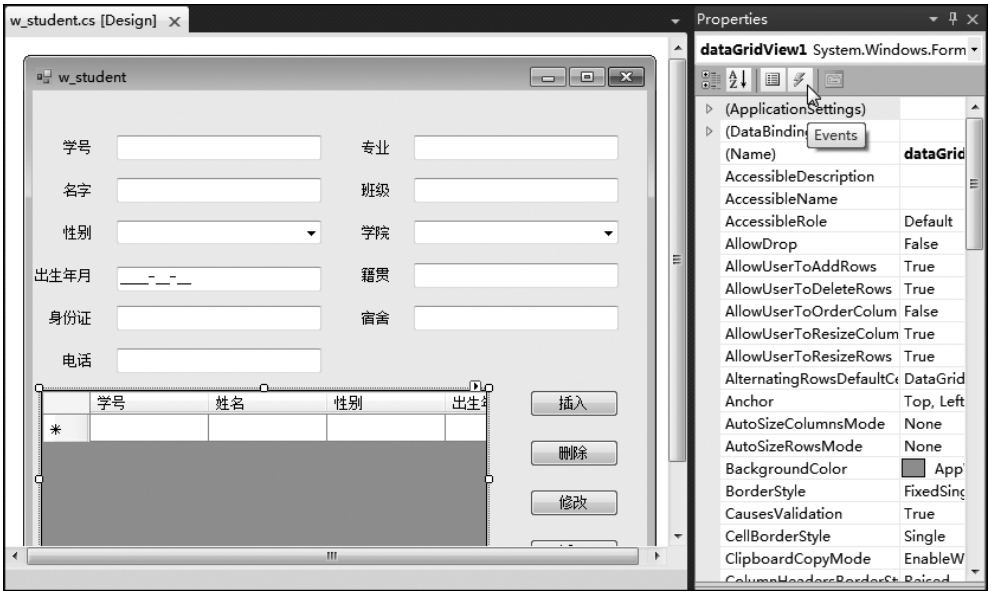

图 5.4　Events 位置示意图

（2）在单击了 Events 按钮后，找到 SelectionChanged 事件，在 SelectionChanged 事件右边的下拉框里双击，如图 5.5 所示。

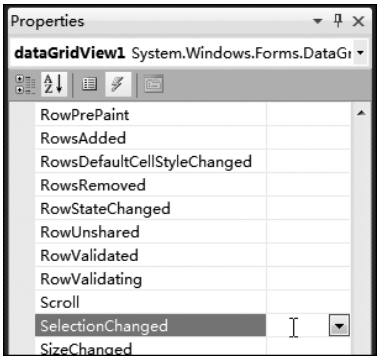

图 5.5　Event 界面示意图

（3）在双击后，VS 2010 将打开编码区，并且将自动生成一个空实现的函数 dataGridView1_SelectionChanged。在 dataGridView1_SelectionChanged 函数体内加入下列代码，如图 5.6 所示。然后，按 Ctrl＋S 组合键保存。

```
//此处变量名 dataGridView1 应与函数名 dataGridView1_SelectionChanged 一致
        DataGridViewRow dvr =dataGridView1.CurrentRow;
        if (dvr !=null)
        {
            DataRowView currentDataRowView =(DataRowView)dvr.DataBoundItem;
            if (currentDataRowView !=null)
            {
                DataRow dr =currentDataRowView.Row;
                //将选中的数据填充到相应的控件中去
                txtid.Text =dr["id"].ToString();
                txtname.Text =dr["name"].ToString();
                cmbsex.SelectedItem =dr["sex"].ToString();
                mtxtbirth.Text =dr["birthday"].ToString();
                txtfield.Text =dr["field"].ToString();
                txtclass.Text =dr["class"].ToString();
                cmbcollege.SelectedItem =dr["college"].ToString();
                txtpersonalid.Text =dr["personalid"].ToString();
                txtfrom.Text =dr["fromO"].ToString();
                txtwhere.Text =dr["whereO"].ToString();
                txtphone.Text =dr["phone"].ToString();
            }
            else
            {
                //若没有选中数据,将控件的数据恢复成默认值
                txtid.Text ="";
                txtname.Text ="";
                cmbsex.SelectedItem ="男";
                mtxtbirth.Text ="";
                txtfield.Text ="";
                txtclass.Text ="";
                cmbcollege.SelectedItem ="理学院";
                txtpersonalid.Text ="";
                txtfrom.Text ="";
                txtwhere.Text ="";
                txtphone.Text ="";
            }
        }
```

(4) 启动程序调试。单击 Start Debugging 按钮(或者单击 VS 2010 菜单栏 Debug→Start Debugging),在 Form1 窗体的菜单栏中,单击“数据输入”→“学生数据”,打开 w_student 窗体。然后,在 DataGridView 中单击一行数据,此时,该行数据将被填充到窗体的对应控件上去,如图 5.7 所示。若想将控件的数据恢复成默认值,则只需选中一个空行。

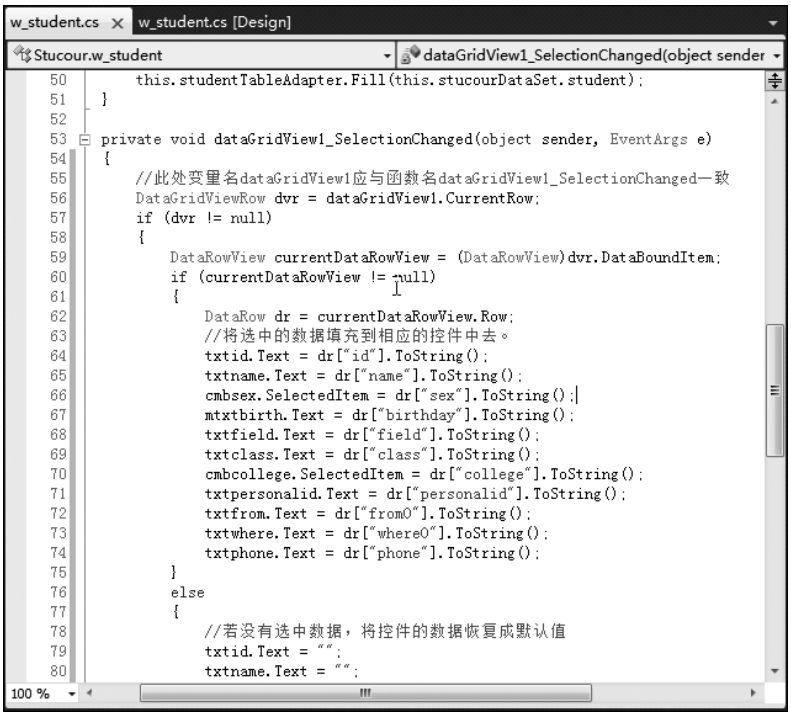

图 5.6　w_student 代码插入示意图

图 5.7　w_student 窗体调试结果示意图

实验三　跟踪调试

(1) 启动程序测试。单击图 5.8 中鼠标处按钮(或者 VS 2010 菜单栏 Debug→Start Debugging)。

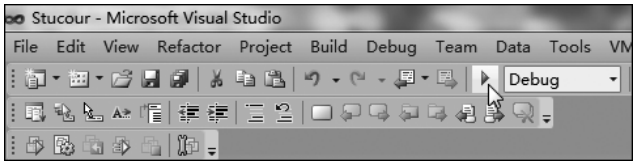

图 5.8　Start Debugging 位置示意图

(2) 在 Form1 窗体的菜单栏中,单击“数据输入”→“学生数据”,打开 w_student 窗体。输入各项数据,然后单击“插入”按钮。窗体中的数据将插入到数据库中,同时,窗体中的 DataGridView 得到更新,将在 DataGridView 中看到刚刚插入的数据(如图 5.9 所示,DataGridView 中最后一条数据,即为刚刚插入的数据)。

图 5.9　数据输入窗体插入操作结果示意图

(3) 如图 5.10 所示,学号对应的文本框为张三的学号(即与前一次一样)。然后再次单击“插入”按钮,然而在下面的 DataGridView 中看不到这些数据,即表明插入数据失败,数据没有被插入到数据库中。什么原因呢? 因为学号是主键,不允许重复。现在利用断点调试跟踪的方法找出原因。

图 5.10　插入数据失败结果示意图

(4) 设置断点。由于是在“插入”按钮的 Click 事件中失败,可以在“插入”按钮的 Click 事件的代码里设置断点。如图 5.11 所示,在代码区的最左边(图中鼠标处,即红点处)单击鼠标,将出现红点(即为断点,在程序调试运行时,当执行到此处时,程序将停止执行,等待调试)。

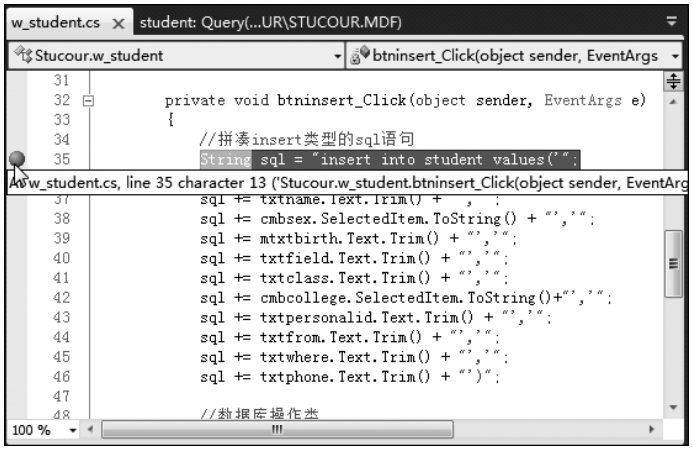

图 5.11　断点位置示意图

(5) 启动程序调试。单击 Start Debugging 按钮(或者 VS 2010 菜单栏 Debug→Start Debugging),在 Form1 窗体的菜单栏中,单击“数据输入”→“学生数据”,打开 w_student 窗体。输入学生数据,如图 5.12 所示。然后,单击“插入”按钮。

图 5.12　w_student 窗体界面示意图

（6）在单击“插入”按钮后，VS 2010 将打开调试窗口。如图 5.13 所示，断点里的黄色箭头表示程序即将执行该行语句。如果没有看到下方的 Autos 视图或者 Call Stack 视图，可以在 VS 2010 的菜单栏里，单击 Debug→Windows→Autos 或者 CallStack 打开这些视图。

图 5.13　调试窗口示意图

(7) 下面介绍 3 个与调试相关的按钮，如图 5.14 所示。在启动调试时，这 3 个按钮一般在 VS 2010 的工具栏的最右边。同时，也可在 VS 2010 的菜单栏中 Debug 菜单下找到 Step Into、Step Over、Step Out 等按钮。另外，还可以使用快捷键，分别为 F11、F10、Shift＋F11。

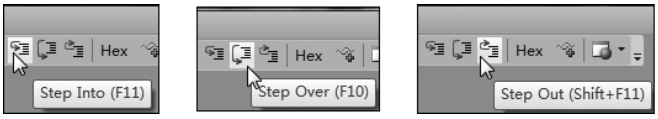

图 5.14　调试按钮位置示意图

Step Into 就是单步执行(即单击一下，执行一句语句)，遇到子函数就进入并且继续单步执行。

Step Over 是在单步执行时，在函数内遇到子函数时不会进入子函数内单步执行，而是将子函数整个执行完再停止，也就是把整个子函数作为一步。

Step Out 就是单步执行到子函数内时，用 Step Out 就可以执行完子函数余下部分，并返回到上一层函数。

(8) 在调试界面里，单击 Step Over 按钮，或者按 F10 键。如图 5.15 所示，黄色箭头移动到了下一行(图中 36 行)，即程序执行完了图 5.15 中第 35 行代码后，即将执行第 36 行代码。同时，在左下方的 Autos 视图中，可以看到 Name 为 sql 的那一行的 Value 为红色，红色表示该变量(字符串变量 sql)在刚刚执行的代码中，其值被改变了。

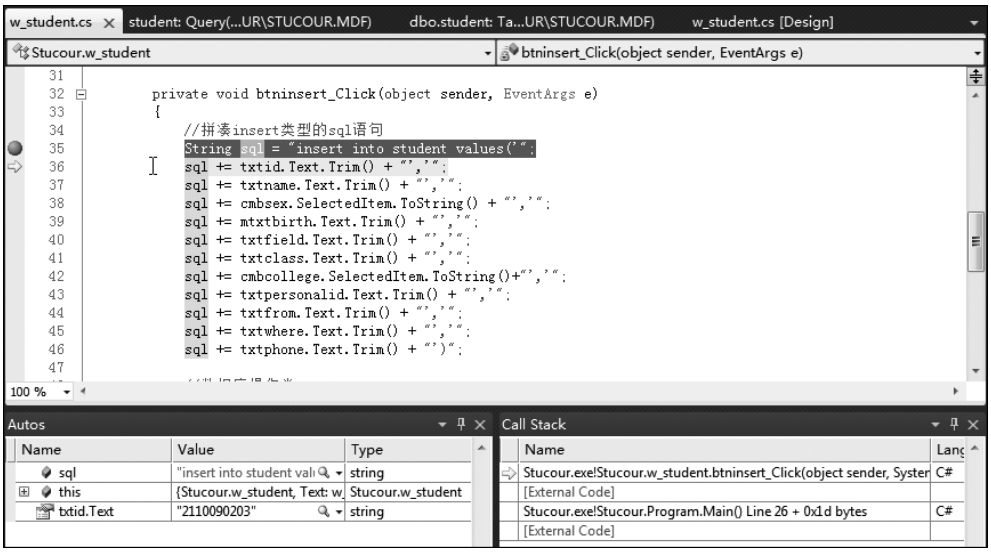

图 5.15　w_student 调试效果示意图

(9) 类似上一步，反复单击 Step Over 按钮，或者按 F10 键(注意观察每一次 Step Into 后，Autos 中的变化)，直到黄色箭头到代码 Database db＝new Database()；这一行(图 5.16 中第 49 行)。此时，拼凑 sql 语句的程序已经执行完成，可以在 Autos 视图中的 sql 行看到其对应的值。将鼠标移动到其值上，VS 2010 将弹出其完整的字符串。另外，