第3章

Visual Basic 程序设计初步

第 2 章中已提到设计一个 VB 应用程序主要包括两项工作，一项是设计用户操作界面，VB 中所有的用户界面是以窗体为基础的。另一项是设计程序代码，使用 VB 语言编写程序要遵循 VB 的语法规定。本章将进一步讨论窗体的属性和事件以及 VB 的基本语法规定等。

3.1 利用窗体进行界面设计

在第 2 章的程序中已经涉及了窗体的使用。我们曾将窗体比喻为一张画纸，在 VB 集成开发环境中，可以使用“工具箱”中的控件在窗体上进行界面设计，也可以使用窗体的属性来“装扮”窗体（如改变窗体的外观，添加丰富的色彩，装入事先准备好的图片，改变它的尺寸等）。

在 Windows 系统中，窗体是最常用的对象，例如，打开资源管理器实际上就是打开一个窗体。Windows 中的窗体都有类似的结构和特点。图 3-1 所示是一个窗体的结构。窗体右上方有三个按钮，自左而右分别是：“最小化”按钮 、“最大化”按钮 和“关闭”窗体按钮 。若单击窗体“最小化”按钮 ，窗体消失，窗体缩小为屏幕底部任务栏上的一个按钮 Form1 ，表示它不是当前打开的窗体；单击该按钮可以恢复窗体，使之成为当前窗体。单击窗体的“最大化”按钮 ，可使窗体充满屏幕，此时窗体的“最大化”按钮变成两个重叠的小方块 ；单击该按钮，能够恢复原来的窗体。单击“关闭”窗体按钮 可以关闭窗体。若再单击工程窗口的“查看对象”按钮，可再次打开窗体，此时窗体被激活。

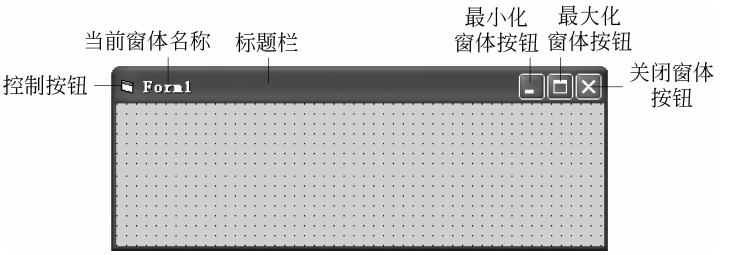

图 3-1 窗体的基本元素

下面举例说明怎样利用窗体进行 VB 应用程序设计。

【例 3-1】 设计一个 VB 程序，窗体上画有三个命令按钮，标题分别是“窗体变小”、“窗体变大”和“关闭窗体”。运行程序时，在窗体上会装入一幅图片。当单击“窗体变大”命令按钮时，窗体变大；单击“窗体变小”按钮时，窗体变小；单击“关闭窗体”按钮，结束程序的运行。窗体界面设计如图 3-2 所示。

图 3-2　例 3-1 的窗体外观

先选择需要的控件，根据题意应选择 3 个命令按钮控件。对窗体和控件的有关属性进行设置，如表 3-1 所示。

表 3-1　例 3-1 对象属性设置

对　象	属　性	设　置
窗体	（名称）	Form1
	Caption	改变窗体尺寸
命令按钮 1	（名称）	cmdLarge
	Caption	窗体变大
命令按钮 2	（名称）	cmdSmall
	Caption	窗体变小
命令按钮 3	（名称）	cmdExit
	Caption	关闭窗体

然后根据题目要求编写有关事件过程的代码：

```
Private Sub Form_Load()                    (窗体装入事件的相应过程)
    Picture=LoadPicture("D:\flower.jpg")
End Sub
```

运行程序的时候，系统自动将窗体装入内存，这就出现了窗体的 Load 事件，触发了 Form_Load 事件过程。Form_Load 事件过程通常用来对窗体的属性和变量进行初始化。

要在窗体上显示图片，需要使用 LoadPicture 函数。LoadPicture 函数的作用是将括号中指定的图形文件调入内存。括号中双引号里的内容是图形文件名，调用 LoadPicture 函数的一般格式为：

LoadPicture("文件名")

“文件名”要求要包含完整的文件路径。调用图片的目的是把它装入某一对象，因此要把它赋值给一个对象，其一般形式为：

[对象.]Picture=LoadPicture("文件名")

其中的“对象”指窗体、图片框、图像框等（注意不是所有的对象中都可以装入图片的，例如

不能把图片装入文本框)。

LoadPicture("D:\flower.jpg")的作用是将 D 盘中的图形文件 flower.jpg 调入内存,并将此值赋给 Picture 属性。赋值号左侧的 Picture 是窗体的一个属性。在第 2 章中已提到,指定属性值有两种方法,一是在属性窗口中设置属性值,二是在程序中设置属性值。在程序中设置属性时,需要指明是哪一个对象的属性。在程序中引用一个属性时,一般要在其前面加上对象名(如 cmdLarge.Caption 指命令按钮 cmdLarge 的 Caption 属性值)。如果在属性前不指定对象名,则默认指当前窗体,所以 Picture 与 Form1.Picture 是等价的。

要想改变窗体的大小尺寸,需要使用窗体的两个相关属性 Height 和 Width。Height 指窗体的高度,Width 指窗体的宽度,单位为 twip(缇),一英寸约等于 1440twip。

如果希望程序运行后,单击一次"窗体变小"命令按钮,窗体的高 Height 和宽 Width 在原来尺寸的基础上减少 500 缇;再单击一次,Height 及 Width 的值再减少 500;不断单击,不断递减,窗体越变越小,那么可以写出使窗体变小的代码如下:

```
Private Sub cmdSmall_Click()
  Form1.Height=Form1.Height-500
  Form1.Width=Form1.Width-500
End Sub
```

程序中的 Form1.Height 表示窗体的高度,其中的对象名 Form1 可以省略。但为了增强程序的可读性,建议不要省略对象名。运行程序后,单击"窗体变小"按钮,其效果见图 3-3。

图 3-3　运行例 3-1

与窗体变小的程序相反,如单击"窗体变大"按钮,窗体的 Height 及 Width 属性值在原有属性值的基础上增加 500,使窗体的尺寸变大。编写窗体变大的代码如下:

```
Private Sub cmdLarge_Click()
  Form1.Height=Form1.Height+500
  Form1.Width=Form1.Width+500
End Sub
```

不断单击"窗体变大"命令按钮,Height 和 Width 属性值不断递加,使窗体越变越大。

在编写程序时,可以选择自己喜欢的图片。需要注意的是要写出图形文件所在位置的完整路径。

下面编写"关闭窗体"按钮事件的代码:

```
Private Sub cmdExit_Click()
  End
End Sub
```

End 语句命令的作用是结束程序的运行。

注意:一般来说,一个程序中应该包括结束程序运行的操作。例如,本例题中的"关

闭窗体”按钮对应的事件过程能够结束程序的运行。

【例 3-2】 设计一个程序，当单击“改变位置”命令按钮时，使窗体的位置改变到屏幕的左上角，单击“还原位置”命令按钮又使它的位置还原，并在标签中显示出所在位置。

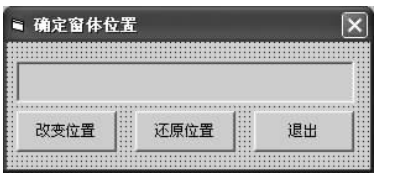

图 3-4　例 3-2 窗体外观

用户界面设计如图 3-4 所示。各个控件的属性设置见表 3-2。

表 3-2　例 3-2 对象属性设置

对　象	属　性	设　置
窗体	（名称）	Form1
	Caption	确定窗体的位置
	BackColor	&H00FFFF80&（浅蓝色）
	BorderStyle	1（Fixed Single）
命令按钮 1	（名称）	cmdMove
	Caption	改变位置
命令按钮 2	（名称）	cmdReset
	Caption	还原位置
命令按钮 3	（名称）	cmdExit
	Caption	退出
标签	（名称）	Lable1
	Caption	置空
	BorderStyle	1（Fixed Single）

说明：窗体的属性 BackColor 的值决定窗体的背景颜色（该属性的设置方法参阅第 2 章）。属性 BorderStyle 决定窗体的“边界风格”（BorderStyle）。它有 4 种可以选择的值。

- 0（None）：窗口无边界。
- 1（Fixed Single）：窗口的边界为单线条，且运行期间窗口的尺寸是固定的（即不能改变其大小）。
- 2（Sizable）：窗口的边界是双线条，且运行期间可以改变窗口的尺寸。
- 3（Fixed Double）：窗口的边界是双线条，且运行期间不可以改变窗口的尺寸。

此处选择 1（Fixed Single），即窗口的边界为单线条，且运行期间窗口的尺寸是固定的。

和例 3-1 一样，程序开始运行时，执行 Form_Load 事件过程，进行初始化窗体的工作，编写该事件过程的代码如下：

```
Private Sub Form_Load()
    Form1.Left=2000
```

```
    Form1.Top=2000
    Label1.Caption="Left 值是:2000,Top 值是:2000"
End Sub
```

程序开始运行,先将 2000 分别赋给窗体的 Left 和 Top 两个属性。Left 和 Top 是用来确定窗体位置的两个属性。Left 属性指明窗体左边界距屏幕左边界的距离(x 轴方向)。Top 属性指明窗体窗口顶部距屏幕顶部的距离(y 轴方向)。确定了 Left 和 Top 属性值,也就确定了窗体在屏幕上的位置。执行 Form_Load 事件过程后,窗体左上角的坐标为(2000,2000),然后,将窗体窗口的位置信息显示在标签中(即 Top 和 Left 属性值)。

题目要求当单击"改变位置"命令按钮时,把窗体移动到指定的位置,可以编写以下事件过程以实现改变窗体位置的功能。

```
Private Sub cmdMove_Click()
    Form1.Left=100
    Form1.Top=100
    Label1.Caption="Left 值是:100,Top 值是:100"
End Sub
```

单击"改变位置"命令按钮时,执行 cmdMove_Click 事件过程,将窗体的位置改变到屏幕的左上角,即坐标为(100,100)的位置。分别将 100 赋给窗体的 Left 和 Top 两个属性,并把这两个值显示在标签中。

如果单击"还原位置"命令按钮,应恢复窗体的初始位置,只需将最初的 Top 和 Left 属性值重新赋给这两个属性即可。其过程代码如下:

```
Private Sub cmdReset_Click()
    Form1.Left=2000
    Form1.Top=2000
    Label1.Caption="Left 值是:2000,Top 值是:2000"
End Sub
```

即 Left 属性值及 Top 属性值均为 2000,窗体回到原来位置。

3.2 Visual Basic 语言的语法基础

进行程序设计必须使用相应的计算机语言。计算机语言有很多种(例如 FORTRAN、PASCAL、C、C++ 等),各有不同的用途。任何一种计算机语言都有其特定的语法规定。在使用这些语言编写程序时,必须要遵守相应的语法规则。这一节介绍最常用的 VB 语法知识。

3.2.1 Visual Basic 的数据类型

计算机能够处理不同类型的信息,如数值、文字、声音、图形、图像等,这些统称为**数据**。数据可以分为不同的种类,称为**数据类型**,如数值类型的数据、字符类型的数据等。

不同类型的数据，在内存中的存储结构不同，占用空间不同，取值范围不同，能够对数据进行的操作也不同。

程序中的数据有两种表示形式：**常量**和**变量**。常量是一个固定的值，如 3、4.5。变量的值在程序运行期间是可以改变的，可以先后向一个变量赋予不同的值。每一个数据（无论常量或变量）都属于一定的数据类型，如 12 是整数类型，34.67 是实数类型。

在 VB 中主要有两大类数据类型，一种是**基本数据类型**，包括数值类型、字符类型等，一种是用户**自定义数据类型**。

在程序设计中，对不同类型的数据可以进行不同的操作。例如，数值型数据之间能够进行算术运算，(2＋3.5)是合法的运算。而(2＋'VB 程序设计')是非法的运算，因为数值数据与字符数据不能直接进行加法运算。因此在程序设计中需要注意数据的类型。表 3-3 中列出 Visual Basic 所允许使用的基本数据类型及取值范围。

表 3-3　Visual Basic 基本数据类型

类　　型	占用字节数	值的有效范围	类型声明符
Integer(整型)	2	－32 768～32 767	%
Long(长整型)	4	－2 147 483 648～2 147 483 647	&
Single(单精度实型)	4	＋1.40E－45～＋3.40E38	!
Double(双精度实型)	8	＋4.97D－324～＋1.79D308	#
Currency(货币类型)	8	－922 337 203 685 477.580 5 ～922 337 203 685 477.580 7	@
String(字符串类型)	1/每字符	0～65 535 个字符	$
Byte(字节)	1	0～255	
Boolan(布尔型)	2	True 或 False	
Date(日期类型)	8	1/1/100～12/31/9999	无
Variant(变体类型)		上述有效范围之一	

"类型声明符"的作用是用简洁的方式表示数据的类型，例如 a%表示 a 是整型变量。如何使用类型声明符将在本章例 3-3 介绍。

VB 中的整型数据和长整型数据都属于**整型数据**，它们的区别在于数据的取值范围不同。整型数据是不带小数点、范围在－32 768 到 32 767 之间的数。在这个范围内的某个数或尾部加一个%符号，表示该数据是一个整型数据。

长整型数是在－2 147 483 647～2 147 483 647 之间不带小数点的数。在这个范围中数或尾部带一个 & 符号的数表示为一个长整数。

VB 中带小数点的实数可以用单精度数或双精度数据表示。**单精度数**是带小数点的实数，有效值数为 7 位。在内存中用 4 个字节(32 位)存放一个单精度数。通常以指数形式(科学记数法)来表示，以 E 或 e 表示指数部分。

双精度数也是带小数点的实数，有效数为 15 位。在内存中用 8 个字节(64 位)存放一个双精度数。双精度数通常以指数形式(科学记数法)来表示，以 D 或 d 表示指数部分。

货币类型(Currency)是专门为计算货币而设置的定点数据类型,它的精度要求高,规定精确到小数点后 4 位。一般的数值型数据在计算机内是通过二进制方式进行运算的,因而可能会有误差,而货币型数据是用十进制方式进行运算的,所以具有比较高的精确度。

字符串类型用以定义一个字符序列,例如,"VisualBasic"就是一个字符串。在内存中一个字符用一个字节来存放。"VisualBasic"需要 11 个字节保存。

日期类型用以表示日期,在内存中一个日期型数据用 8 个字节来存放。其实,有时也可以用字符串表示日期,例如"2011-07-20"。但是用字符串表示的日期不能确保其日期是有效的,也就是说,字符串有可能出现"2014-13-32"。而使用日期数据类型,系统按照日期数据类型的约束和限制,对数据进行检查,就可以有效地避免出现不合理日期数据的情况。

VB 中还有一种 Variant 数据类型,称为**变体类型**或**通用类型**。变体类型可以表示上述任何一种数据类型。假设定义 a 为变体类型变量:

```
Dim a As Variant
```

则在变量 a 中可以存放任何类型的数据,如:

```
a=3.5                    (存放一个实数)
a="BASIC"                (存放一个字符串)
a="03/31/1998"           (存放一个日期型数据)
```

根据赋给 a 的值的类型不同,变量 a 的类型不断变化,这就是称之为变体类型的由来。如果没有定义变量的数据类型,VB 自动将该变量定义为 Variant 类型。不同类型的数据在 Variant 变量中是按其实际的数据类型存放,例如将一个整数赋给变体类型变量 a,在内存区中按整型数据的方式存放。用户不必进行任何数据类型的转换工作,数据类型的转换工作由 VB 系统自动完成。

怎样知道一个变体类型的变量在程序中究竟被作为何种数据类型? VB 中有一个函数 VarType,能够测定一个 Variant(变体型)变量在程序中实际的数据类型。VarType 函数的值是一个数值,其含义如表 3-4 所示。

表 3-4　VarType 函数值

VarType 函数值	数值类型	VarType 函数值	数值类型
0	空	5	双精度
1	Null	6	货币型
2	整型	7	日期型
3	长整型	8	字符串型
4	单精度		

【例 3-3】 编写一段程序,给不同的变量赋予不同的值,利用 VarType 函数测试这些变量的数据类型。为了完成这项工作,在窗体上添加一个名称为 cmdTest 的命令按钮,

在该命令按钮的事件过程中编写数据类型测试的代码如下：

```
Private Sub cmdTest_Click()
  Dim Var1 As Variant            (指定变量 Var1 是通用类型)
  Int1=123
  Long1=186&
  Single1=12.6!
  Double1=34.5
  Str1="abcd"
  Cur=8886@
  Da=#10/21/1997#
  Print VarType(Var1), VarType(Int1), VarType(Long1), VarType(Single1)
  Print VarType(Str1), VarType(Cur), VarType(Double1), VarType(Da)
End Sub
```

Var1 被定义为 Variant(变体型)变量，其他各变量(如 Int1，Long1，Single1 等)均未定义为何种类型，因此都作为 Variant 类型处理。当分别对 7 个 Variant 型变量赋值后，再用 VarType 函数测试这 8 个变量实际的数据类型。运行此程序，输出结果如图 3-5 所示。

图 3-5　VarType 函数测试结果

Var1 是变体型变量，由于程序中未对它赋值，因此 VarType(Var1)的值为 0。Int1 也是变体变体型变量，由于已被赋值为整数 123，因此 VarType(Int1)的值为 2，由表 3-4 可知变量 Int1 的类型为整型。与此类似，变体型变量 Long1 被赋以长整数 186&，因此 VarType(Long1)的值为 3，表示此时该变量为长整型。其他变量的类型可以用类似的方法判定。

Print 是输出语句，在 Print 前面没有指定对象，故默认为在窗体上输出为各函数的值。

仅有以上基本数据类型有时不能满足设计的需要。有些情况下，我们希望将不同类型的数据组合成一个有机的整体，以便于引用。例如把一个职工的职工号、姓名、年龄、电话、地址等简单数据组合成一个复合的数据，这样一个复合的数据可以由若干不同类型的、互相有联系的数据项组成。在程序设计领域一般把这种复合的数据称为**记录**(record)。在 Visual Basic 中，用户可以用 Type 语句来自己定义这种数据类型。它的一般形式为：

```
Type  类型名
  成员名  As  类型
  成员名  As  类型
  成员名  As  类型
  ⋮
End Type
```

例如可以定义一个名为 Employee(职工)的类型，其中包括有职工号、姓名、年龄、电

话和住址等信息：

```
Type  Employee
 EmpNo    As  Integer
 Name     As  String * 10
 Age      As  Integer
 Tel      As  String * 10
 Address  As  String * 20
End Type
```

在定义了 Employee 类型之后，就可以用它定义 Employee 类型的变量了。例如可以定义一个 Emp 类型的变量：

```
Dim  Emp  As  Employee
```

此语句定义了 Employee 类型的变量 Emp，它包括有 5 个成员。在程序中我们可以用“变量.成员”这样的形式来引用各个成员，如下面这样：

```
Emp.Name        表示 Emp 变量中的 Name 成员的值(某一职工的名字)
Emp.Address     表示 Emp 变量中的 Address 成员的值(某一职工的地址)
Emp.EmpNo       表示 Emp 变量中的 EmpNo 成员的值(某一职工的号码)
```

3.2.2 变量名和变量值

在程序执行过程中，其值可以发生变化的量称为**变量**(**variable**)。例 3-2 中有一个赋值语句：

```
Form1.Left=2000
```

赋值号左侧的 Form1 是窗体对象的名字，Left 是窗体的属性，可以看到属性 Left 的值是可以改变的，因此在 VB 程序中它是一个变量。

变量需要有一个名称，作为标识，这就是**变量名**。在 Visual Basic 中，对变量命名有如下规定：

(1) 变量名的第一个字符必须是字母，如 12ab 是非法的变量名。

(2) 变量名的第二个字符及其后的字符可以是字母、数字及下划线。

(3) 变量名的长度不能超过 255 个字符。

(4) 可以用表示变量类型的字符(如 $，%，# 等)作为变量名的最后一个字符。

(5) 不能将 Visual Basic 语言中规定的保留字(如语句命令、函数名等)作为变量名使用，例如，Print 不是合法的变量名。

(6) 在变量名中，大小写字母是等价的，例如在同一个程序中，变量名 Abc、abc、ABC 表示的是同一个变量。

(7) 变量名中间不能有空格。例如，wang hong 不是合法的变量名。

(8) 在同一个程序模块中，不能有相同的变量名。

根据上述规则，变量名 a、a1、flag、well$ 等均是合法的变量名。

(9) 变量名的前面可以有对象名,指明是哪个对象中的变量,在对象名和变量名之间用句点(.)相隔,如 a.b 表示 a 对象的 b 变量。同样,Form1.Left 表示 Form1 对象的 Left 变量(此属性是变量)。如果不出现对象名,默认为当前对象(例如窗体)。

3.2.3 定义变量

前已说明,所有变量都属于一定的数据类型。在 VB 程序中应当对变量进行定义(define),定义的作用是告诉 VB 系统该变量是什么类型,以便系统据此对该变量进行适当的内存分配(不同类型的数据在内存中所占的存储空间和存储方式是不同的)。可以用 Dim 或 Static 对变量进行定义,也可以使用表 3-3 中列出的类型声明符指定变量的类型。例如:

```
Dim a  As Integer          (指定变量 a 为整型)
Dim ch As Char             (指定变量 ch 为字符型)
Y=123%                     (没有指定变量 Y 的类型)
```

上述语句中的第 1 条语句定义了一个整型变量 a。第 2 条语句定义了一个字符变量 ch。第 3 条语句未显式地指定变量 Y 的类型,VB 允许不定义变量直接使用,凡是未指定变量类型的,均默认为变体类型变量(通用类型)。在第 3 条语句中将 123%赋给变量 Y,而"%"是"类型声明符",其作用是用简洁的方式指定数据的类型,即 123%是一个整型数据,因此变量 Y 当前是整型变量。

1. 用 Dim 定义变量

使用 Dim 定义变量的语句格式如下:

Dim <变量名>As <数据类型>

例如:

```
Dim  Name  As  String
Dim  Sum  As  Long
Dim  Num  As  Integer
Dim  X Integer,Y As Single
```

上述 Dim 语句定义了几个变量:变量 Name 为字符串类型变量;变量 Sum 为长整型变量;变量 Num 为整型变量;变量 X 为整型变量,变量 Y 为单精度型变量。

可以省略 Dim 语句中的 As 子句,即按照如下格式定义变量:

Dim <变量名>

由于省略了 As 子句,未指定类型,VB 把变量默认为变体 Variant 类型。例如,如下语句定义了名称为 What 的变体类型变量:

```
Dim  What
```

需要特别说明的是,若使用一条 Dim 语句定义多个变量,例如: