第3章　建立简单的 Visual Basic 应用程序

第 2 章介绍了 Visual Basic 中对象的概念，讨论了两种最主要的预定义对象，即窗体和控件。本章将通过一个简单例子说明 Visual Basic 应用程序开发的一般过程。

3.1　语　　句

程序是对计算机要执行的一组操作序列的描述，而高级语言源程序的基本组成单位是语句，它是执行具体操作的指令。

3.1.1　Visual Basic 中的语句

1. 语句的构成

Visual Basic 中的语句由 Visual Basic 关键字、对象属性、运算符、函数以及能够生成 Visual Basic 编辑器可识别指令的符号组成。每个语句以回车键结束，一个语句行的最大长度不能超过 1023 个字符。在书写语句时，必须遵循一定的规则，这种规则称为语法。

一个语句可以很简单，也可以很复杂。例如：

```
Cls
```

就是一个简单的 Visual Basic 语句，它由一个关键字组成。而

```
Print "计算结果为:"; (a / (b + c) / (d + e / Sqr(f)))
```

是一个稍复杂些的语句。

2. 自动语法检查

为了使程序能被 Visual Basic 正确地识别，在书写代码时必须遵循一定的语法规则。如果设置了“自动语法检测”(用“工具”菜单“选项”命令对话框中的“编辑器”选项卡，如图 3.8 所示)，则在输入语句的过程中，Visual Basic 将自动对输入的内容进行语法检查，如果发现了语法错误，则弹出一个信息框，提示出错的原因。

Visual Basic 按自己的约定对语句进行简单的格式化处理，例如命令词的第一个字母大写，运算符前后加空格等。在输入语句时，命令词、函数等可以不必区分大小写。例如，在输入 Print 时，不管输入 Print，print，还是输入 PRINT，按回车键后都变为 Print。为了提高程序的可读性，在代码中应加上适当的空格，同时应按惯例处理字母的大小写。

3. 复合语句行

在一般情况下，输入程序时要求一行一句，一句一行。但 Visual Basic 允许使用复合语句行，即把几个语句放在一行中，各语句之间用冒号(:)隔开。例如：

```
a = 7: b = 3: c = 4: Print a Mod 3 + b^3 / c \ 5
```

4. 续行

当语句较长时，为了便于阅读程序，可以通过续行符把一个语句分别放在几行中。Visual Basic 中使用的续行符是下划线(_)。如果一个语句行的末尾是下划线，则下一行与该行属于同一个语句行。例如：

```
Print Val(Text1.Text) _
    + Val(Text2.Text) _
    + Val(Text3.Text)
```

它与下面的语句等价：

```
Print Val(Text1.Text) + Val(Text2.Text) + Val(Text3.Text)
```

注意，续行符只能出现在行尾，而且与它前面的字符之间至少要有一个空格。

Visual Basic 中可以使用多种语句。早期 BASIC 版本中的某些语句(如 PRINT，CLS 等)，在 Visual Basic 中称为方法，而有些语句(如流程控制、赋值、注释、结束、暂停等)仍称为语句。本节介绍 Visual Basic 的几个语句，包括赋值、注释、暂停和结束语句，其他语句将在以后的章节中介绍。

3.1.2 赋值、注释、暂停和结束语句

1. 赋值语句

用赋值语句可以把指定的值赋给某个变量或某个带有属性的对象，其一般格式为：

[Let] 目标操作符 = 源操作符

这里的“源操作符”包括：变量(简单变量或下标变量)、表达式(数值表达式、字符串表达式或逻辑表达式)、常量及带有属性的对象；而“目标操作符”指的是变量和带有属性的对象。“=”称为“赋值号”。赋值语句的功能是：把“源操作符”的值赋给“目标操作符”。例如：

```
Total = 99                      '把数值常量 99 赋给变量 Total ( '是注释符)
ReadOut$ = "Good Morning!"      '把字符串常量赋给字符串变量
Try1 = Val(Text1.Text)          '把对象 Text1 的 Text 属性转换为数值赋给数值变量
Text1.Text = Str$(Total)        '把数值变量 Total 转换为字符串赋给带有
                                'Text 属性的对象
Text1.Text = Text2.Text         '把带有 Text 属性的对象 Text2 赋给带有 Text 属性
                                '的对象 Text1
StartTime = Now                 '把系统的当前时间赋给变体类型变量
```

在上面的例子中，把数值常量赋给数值变量或把字符串赋给字符串变量都比较简单，也容易理解，而对象赋值可能抽象一些。所谓对象赋值，实际是对对象的属性赋值，即改变对象的属性值。例如在语句

```
Text1.Text = Str$(Total)
```

中，把 Total 的值转换为字符串，赋给文本框对象，使该对象的 Text 属性变为 Str$(Total)。假定 Total 的值为 99，则执行上述语句后，文本框 Text1 中显示 99。而在

```
Text1.Text = Text2.Text
```

中，则是把文本框 Text2 的 Text 属性赋给文本框 Text1 的 Text 属性，执行该语句后，两个文本框中显示的内容相同。

说明：

(1) 赋值语句兼有计算与赋值双重功能，它首先计算赋值号右边“源操作符”的值，然后把结果赋给赋值号左边的“目标操作符”。例如：

```
BitCount = ByteCount * 8
Energy = Mass * LIGHTSPEED^2
```

(2) 在赋值语句中，“=”是赋值号，与数学上的等号意义不一样。

(3) “目标操作符”和“源操作符”的数据类型必须一致(第 4 章将介绍 Visual Basic 的数据类型)。例如，不能把字符串常量或字符串表达式的值赋给整型变量或实型变量，也不能把数值赋给文本框的 Text 属性。如果数据类型相关但不完全相同，例如把一个整型值存放到一个双精度变量中，则 Visual Basic 将把整型值转换为双精度值。但是，不管表达式是什么类型，都可以赋给一个 Variant(变体)变量。

(4) 如前所述，Visual Basic 中的语句通常按“一行一句，一句一行”的规则书写，但也允许多个语句放在同一行中。在这种情况下，各语句之间必须用冒号隔开。例如：

```
a = 3 : b = 4 : c = 5
```

在一行中有 3 个语句。这样的语句行称为复合语句行。复合语句行中的语句可以是赋值语句，也可以是其他任何有效的 Visual Basic 语句。但是，如果含有注释语句，则它必须是复合语句行的最后一个语句。

(5) 赋值语句以关键字 Let 开头，因此也称 Let 语句。其中的关键字 Let 可以省略。

2. 注释语句

为了提高程序的可读性，通常应在程序的适当位置加上必要的注释。Visual Basic 中的注释是“Rem”或一个撇号“'”，一般格式为：

```
Rem 注释内容
'注释内容
```

例如：

```
' This is a test statement
Rem 这是一个子程序
```

说明：

(1) 注释语句是非执行语句，仅对程序的有关内容起注释作用。它不被解释和编译，但在程序清单中，注释被完整地列出。

(2) 任何字符(包括中文字符)都可以放在注释行中作为注释内容。注释语句通常放

在过程、模块的开头作为标题用，也可以放在执行语句(单行或复合语句行)的后面，在这种情况下，注释语句必须是最后一个语句。例如：

```
Text1.Text = "Good morning"        'This is a test
a = 5 : b = 6 : c = 7              '对变量 a,b,c 赋值
```

(3) 注释语句不能放在续行符的后面。

(4) 当注释语句出现在程序行的后面时，只能使用撇号“'”，不能使用 Rem。例如：

```
intVal = 100    Rem 赋值
```

是错误的，必须改为：

```
intVal = 100    '赋值
```

3. 暂停语句

Visual Basic 中的暂停语句为 Stop，其格式如下：

```
Stop
```

Stop 语句用来暂停程序的执行，它的作用类似于执行“运行”菜单中的“中断”命令。当执行 Stop 语句时，将自动打开立即窗口。

在解释系统中，Stop 语句保持文件打开，并且不退出 Visual Basic。因此，常在调试程序时用 Stop 语句设置断点。如果在可执行文件(*.exe)中含有 Stop 语句，则将关闭所有文件。

Stop 语句的主要作用是把解释程序置为中断(Break)模式，以便对程序进行检查和调试。一旦 Visual Basic 应用程序通过编译并能运行，则不再需要解释程序的辅助，也不需要进入中断模式。因此，程序调试结束后，生成可执行文件之前，应删去代码中的所有 Stop 语句。

4. 结束语句

Visual Basic 中的结束语句为 End 语句，其格式如下：

```
End
```

End 语句通常用来结束一个程序的执行。可以把它放在事件过程中，例如：

```
Sub Command1_Click()
    End
End Sub
```

该过程用来结束程序，即当单击命令按钮时，结束程序的运行。

End 语句除用来结束程序外，在不同的环境下还有其他一些用途，例如：

```
End Sub          '结束一个 Sub 过程
End Function     '结束一个 Function 过程
End If           '结束一个 If 语句块
End Type         '结束记录类型的定义
End Select       '结束情况语句
```

当在程序中执行 End 语句时，将终止当前程序，重置所有变量，并关闭所有数据文件。

在大多数情况下，一个程序中有没有 End 语句，对程序的运行没有什么影响。但是，如果没有 End 语句，或者虽有但没有执行（例如不执行含有 End 语句的事件过程），则程序不能正常结束，必须执行“运行”菜单中的“结束”命令或单击工具栏中的结束按钮。为了保持程序的完整性，应当在程序中含有 End 语句，并且通过 End 语句结束程序。

3.2 编写简单的 Visual Basic 应用程序

用传统的面向过程的语言进行程序设计时，主要的工作就是编写程序代码，遵循编程—调试—改错—运行这样一种模式。在用 Visual Basic 开发应用程序时，完全打破了这种模式，使程序的开发大为简化，而且更容易掌握。

3.2.1 程序设计

一般来说，在用 Visual Basic 开发应用程序时，需要以下 3 步：

(1) 建立可视用户界面。

(2) 设置可视界面特性。

(3) 编写事件驱动代码。

通过一个例子来说明如何在 Visual Basic 环境下设计应用程序。

【例 3.1】 在窗体上画 3 个命令按钮和一个文本框，把窗体的标题设置为“Visual Basic 程序设计示例”，把 3 个命令按钮的标题分别设置为“显示”、“清除”和“结束”，把文本框的内容设置为空白。程序运行后，如果单击第一个命令按钮，则在文本框中显示“欢迎使用 Visual Basic 6.0”，如图 3.1 所示；如果单击第二个命令按钮，则清除文本框中显示的内容；而如果单击第三个命令按钮，则结束程序。

图 3.1 例题运行情况

下面介绍如何设计这个应用程序。

1. 建立用户界面

为了建立应用程序，首先应建立一个新的工程，这可以通过“文件”菜单中的“新建工程”命令来实现。执行该命令后，将打开“新建工程”对话框，双击该对话框中的“标准 EXE”图标（或者单击该图标，然后单击“确定”按钮）即可建立一个新的工程。

用户界面由对象组成，建立用户界面实际上就是在窗体上画出代表各个对象的控件。由题意可知，需要建立的界面包括 5 个对象（窗体本身也是一个对象），即窗体和 4 个控件，其中 3 个是命令按钮，一个是文本框，可以按下面的步骤建立用户界面：

(1) 单击工具箱中的命令按钮图标，在窗体的适当位置画一个命令按钮（命令按钮 1），画完后，按钮内自动标有 Command1。

(2) 重复步骤 1，分别画出命令按钮 2 和命令按钮 3，两个按钮内分别自动标有 Command2 和 Command3。

(3) 单击工具箱中的文本框图标，然后在窗体的适当位置画出文本框控件，文本框内自动标有 Text1。

(4) 上述 4 个控件画完后，根据具体情况，对每个控件的大小和位置进行适当调整。

设计完用户界面后，窗体的结构如图 3.2 所示。

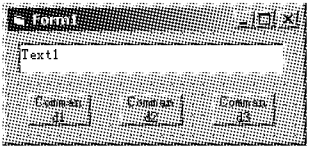

图 3.2 界面设计(1)

2. 设置属性

前面画出的 4 个控件构成了用户界面，这 4 个控件就是 4 个对象。实际上，除 4 个控件外，还有一个对象，这就是窗体，其当前名称(Name 属性)和标题(Caption 属性)为 Form1。在建立用户界面后，每个对象都有一个默认标题(Caption 或 Text 属性)，分别为 Command1、Command2、Command3、Text1 和 Form1。

根据题意，命令按钮 1 的标题应为“显示”。可按如下步骤修改：

(1) 单击标有 Command1 的命令按钮，将其激活，其周围出现 8 个小方块，表明该控件是活动的。

(2) 激活属性窗口，从属性列表中找到 Caption 属性，单击该属性条，其右侧显示该控件的默认设置值 Command1。

(3) 从键盘上输入汉字“显示”(引号不要输入)，取代 Command1，命令按钮 1 中的内容(标题)即变为“显示”。

(4) 输入中文后，由于字体太小，可能看不清楚，可以用属性窗口中的 Font 属性将其放大。在属性列表中找到 Font 属性，单击该属性条，在右端显示 3 个小点，单击这 3 个小点，将打开 Font 对话框。在该对话框中把“字体”设置为“宋体”，把“字形”设置为“粗体”，把“大小”设置为“四号”。

用与修改命令按钮 1 类似的操作把 Command2 修改为“清除”，把 Command3 修改为“结束”，并设置与 Command1 相同的字体和大小。

文本框用来显示信息，应把它变为空白。修改步骤如下：

(1) 单击文本框，将其激活。

(2) 单击属性窗口，从属性列表中找到 Text 属性，双击(或单击)该属性条，设置框内显示“Text1”。

(3) 按 Del 键或退格键，文本框内的 Text1 即被清除。

窗体标题的默认值为 Form1，程序运行时，即为输出窗口的标题。为了把窗体标题改为“Visual Basic 程序设计示例”，可按如下步骤操作：

(1) 单击窗体内没有控件的地方，使窗体成为当前的活动对象。

(2) 用前面介绍的方法找到并单击属性窗口中的 Caption 属性。

(3) 从键盘上输入“Visual Basic 程序设计示例”，则窗体顶部的 Form1 被输入的文本代替。

如前所述，对象可以有多种属性。在上面的操作中，实际上只设置了两种属性，即 Caption 和 Font。设置属性后的窗体如图 3.3 所示。

说明：

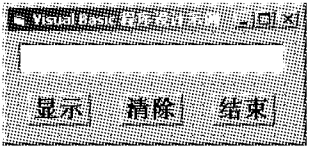

图 3.3　界面设计(2)

(1) 除标题外，每个对象都还有其他一些属性，例如对象名称(Name 属性，在属性列表中为“(名称)”)。每个对象都有 Name 属性，可以通过属性窗口为其赋予一个适当的值。如果不赋值，则使用其默认属性。在上面的例子中，4 个控件的 Name 属性值分别为 Command1，Command2，Command3 和 Text1，窗体的 Name 属性值为 Form1，它们均为默认属性值，可以用其他名字作为 Name 的属性值。注意，标题(Caption)和对象名称(Name)是完全不同的两种属性。Caption 是对象的标识，而 Name 是对象的名字。在编写代码(见本节中 3 条)时，将针对对象的 Name 属性值设计操作。从前面设计界面的过程中可以看出，Caption 属性和 Name 属性使用同样的默认值，Caption 属性的默认值在对象中显示出来，而 Name 属性值从表面上看不出来。

此外还应注意，Name 属性是只读属性，即只能在设计期间设置，在运行期间不能改变。此外，在属性窗口中，Name 属性的属性条为“(名称)”，并放在属性窗口的顶部。但是，在程序代码中，仍使用 Name。

(2) 用户界面是设计应用程序时的重要一环。与传统的程序设计语言不同，在用 Visual Basic 设计应用程序时，用户界面不是程序行，而是放在窗体中的若干个控件，这些控件和窗体均被称为对象。因此，设计用户界面实际上是一个建立对象的过程。为了使界面的设计清晰而有条理，通常在设计前将界面中所需要的对象及其属性画成一个表，然后按照这个表来设计界面。上例中的界面有 4 个控件和一个窗体，可以通过两种格式画出其属性设置表，第一种格式见表 3.1，第二种格式见表 3.2。

表 3.1　对象属性设置(格式 1)

对　象	Name	Caption	Text
窗体	Form1	“Visual Basic 程序设计示例”	
左命令按钮	Command1	“显示”	无
中命令按钮	Command2	“清除”	无
右命令按钮	Command3	“结束”	无
文本框	Text1	无	空白

表 3.2　对象属性设置(格式 2)

对　象	属　性	设置值
窗体	Name	Form1
	Caption	“Visual Basic 程序设计示例”
左命令按钮	Name	Command1
	Caption	“显示”
中命令按钮	Name	Command2
	Caption	“清除”
右命令按钮	Name	Command3
	Caption	“结束”
文本框	Name	Text1
	Text	空白

每个对象都有自己特定的属性，有些属性只能用于部分对象。例如命令按钮没有Text属性，而文本框没有Caption属性。激活一个对象后，该对象所具有的全部或大多数属性值就在属性窗口的属性列表中显示出来。

(3) 控件放在窗体中，窗体及其控件构成了用户界面。程序运行后，如果对界面不满意，可以结束运行，然后进行调整。从前一章中知道，窗体大小及每个控件的位置、大小均可根据需要任意调整，同时可改变标题及输出字体的属性。

3. 编写代码

Visual Basic采用事件驱动机制，其程序代码是针对某个对象事件编写的，每个事件对应一个事件过程。用鼠标单击一个对象是经常用到的事件，可以针对这样的事件编写事件过程。

(1) 程序代码窗口

过程在程序代码窗口中输入和编辑。为了输入过程中的代码，必须先进入程序代码窗口。可以用四种方法打开程序代码窗口，即：双击窗体或窗体中的控件，执行“视图”菜单中的“代码窗口”命令，按F7键和单击工程资源管理器窗口中的“查看代码”按钮。例如，双击窗体，进入程序代码窗口，屏幕上出现如图3.4所示的代码窗口。

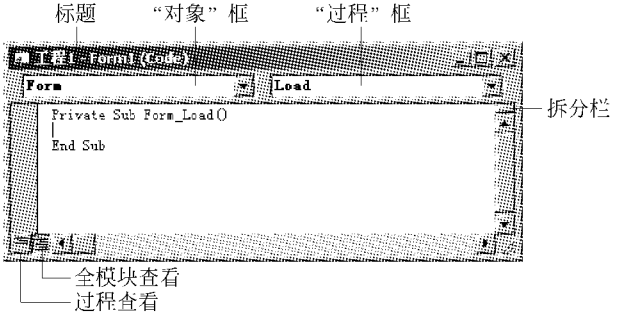

图3.4 程序代码窗口

窗口顶部的“工程1-Form1 (Code)”是代码窗口的标题，在它下面的一行分为两栏，左边一栏为“对象”框，在该方框中的Form是当前对象名。右边一栏为“过程”框，在该栏中显示的是事件的名字，当前的事件名为Load(装入)。在窗口的左下角有两个按钮，如果选择(单击)“过程查看”按钮，则窗口内只显示当前过程代码；而如果选择“全模块查看”按钮，则显示当前模块中的所有过程的代码。此外，在垂直滚动条的上面，有一个“拆分栏”，把鼠标光标移到该栏上，鼠标光标变为上下双向箭头，此时按住左键拖动鼠标，可以把代码窗口分为两个窗口。

事件过程的开头和结尾由系统自动给出，即：

```
Private Sub Form_Load()


End Sub
```

可在这两行之间输入程序代码。

“Private”意为“私有”，用来表明事件过程的类型。过程名(这里是Form_Load())由

两部分组成，前面一部分是对象名(Form)，后面一部分是该对象的事件名(Load)，中间用下划线相连，在过程名的后面有一对括号。事件过程名的两个部分可以根据需要任意组合。例如，单击“对象”栏右端的箭头，将列出各对象的名字，如图 3.5 所示，此时单击 Command3，则原来过程名中的 Form 即变为 Command3。接着单击“过程”栏右端的箭头，以下拉方式列出各种事件(见图 3.6)，然后单击 MouseDown，则过程名中的 Load 被 MouseDown 代替。于是过程名变为：

```
Private Sub Command3_MouseDown(Button As Integer, Shift As Integer, _
        X As Single, Y As Single)


End Sub
```

其中第一行代码最后的下划线是续行符。

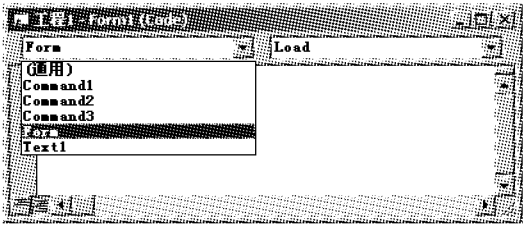

图 3.5　对象名称

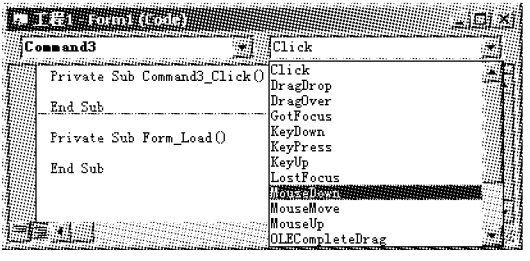

图 3.6　事件名称

不管用哪种方法进入程序代码窗口，都可以通过对象名与事件名的不同组合来改变事件过程名。

可以看出，这里的过程名与其他语言中的过程名或函数名是有区别的，它的名字不能任意指定，而只能由系统提供的对象名和事件名组成，因而称为事件过程。

(2) 编写过程代码

过程代码是针对某个对象事件编写的。为了指明某个对象的操作，必须在方法或属性前加上对象名，中间用句点(.)隔开。例如：

```
Text1.Text = "欢迎使用 Visual Basic 6.0 "
```

这里的 Text1 是控件(对象)名，Text 是文本框的属性。执行上面的语句后，将在 Text1 文本框中显示“欢迎使用 Visual Basic 6.0”。

如果不指出对象名，则方法或属性是针对当前窗体的。

事件过程是对某个对象事件所执行的操作。例如，事件过程 Command1_Click 执行的是单击(Click)控件 Command1 时所执行的操作。根据题意，该过程的代码如下：

```
Private Sub Command1_Click()
    Text1.Fontsize = 12
    Text1.Text = "欢迎使用 Visual Basic 6.0 "
End sub
```

上述事件过程有两个语句，第一个语句用来设置字体大小(默认为 9)，第二个语句用来显示一个字符串。由于前面有对象名 Text1，所以这两个语句都是针对 Text1 文本框执行的。程序执行后，只要单击命令按钮 Command1，就可以在文本框中显示“欢迎使用 Visual Basic 6.0”，其字体大小为 12。根据题意，第二个事件过程的内容如下：

```
Private Sub Command2_Click()
    Text1.Text = ""
End Sub
```

该事件过程的功能是，当单击命令按钮 Command2 时，把文本框中的内容清为空白。

第 3 个事件过程是命令按钮 Command3 的单击(Click)事件过程，用来结束程序：

```
Private Sub Command3_Click()
    End
End Sub
```

该过程的功能是，当单击命令按钮 Command3 时，结束程序的运行。

至此，程序设计工作全部结束。

这个程序比较简单，但展示了 Visual Basic 应用程序设计的全过程。可以看出，这一过程与传统的程序设计过程有着本质的区别。在用 Visual Basic 设计应用程序时，通常不必编写含有大量代码的程序，而是首先建立用户界面，设置各个对象的属性，然后编写由用户启动的事件来激活的若干个微小程序，即事件过程，从而大大简化了程序开发过程。

Visual Basic 能自动进行语法检查。每输入完一行代码并按回车键后，Visual Basic 能自动检查该行的语法错误。如果语句正确(没有语法错误)，则自动以不同的颜色显示代码的不同部分，并在运算符前后加上空格。输入完 3 个事件过程的代码后，代码窗口如图 3.7 所示。

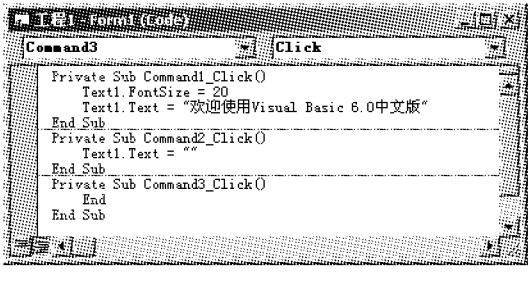

图 3.7　事件过程代码