第 9 章　创建和使用自己的类

类是具有多态性的用户定义类型。封装将代码和数据组织在同一个类模块中，使得对象能保护和验证其中的数据。借助创建对象的类能将数据和过程组织成一个整体。本章介绍类如何增强代码的活力，如何导致新的编程方法的产生。

9.1　创建自己的类

在VB中创建自己的类是通过在工程中添加类模块，然后设计类模块的相关代码实现的。本节讨论类的用途和在工程中定义类模块的过程。

9.1.1　类的作用

用户定义类型是一个使相关数据项成组的强大工具。例如，假设我们定义一个名为Point的用户定义类型：

```
Public Type Point
    x As Integer
    y As Integer
End Type
```

可以声明一个Point类型的变量p，然后编写两个过程，一个为setvalue，用于给p的各个字段赋值；另一个为display，用于显示其值。这些代码如下：

```
Dim p As Point
Public Sub setvalue(ByVal a As Integer, ByVal b As Integer)
  p.x = a
  p.y = b
End Sub
Public Sub display()
  MsgBox "(" + Str(p.x) + "," + Str(p.y) + ")"
End Sub
```

面向对象的编程通过将数据和过程结合成一个单一实体解决了上述问题，当把上面的用户定义类型Point变成一个CPoint类时，其数据变成了私有型，访问它们的过程移到类中并变成了属性和方法。这就是所谓的封装，也就是说，对象是一个包括代码和数据的单元。

当从CPoint类创建一个变量（即创建该类的一个对象）时，访问其数据必须通过构成

其接口的属性及方法。

现在，暂不要考虑如何将过程放进类中，也不要考虑如何理解属性过程和私有变量的语法。重要的是要记住，可以定义一个对象来封装数据和确保数据的可靠性。

9.1.2　定义类模块

在VB中通过“类模块”来创建类。我们以一个例子说明如何使用类模块定义类，还将讨论如何为新的类创建属性和方法，并演示对象是如何创建和撤销的。

【例9.1】　创建一个工程并在其中添加一个类模块Class1和一个窗体Form1，共同实现用户输入的一个字符串的逆转。

首先打开一个新的“标准.EXE”工程，建立一个工程名为“工程1”的工程，在“工程”菜单中选中“添加类模块”，插入一个类模块Class1。开始时，该类模块窗口是空白的，我们可以输入相应的类定义代码，如图9.1所示是创建的Class1类模块的代码。

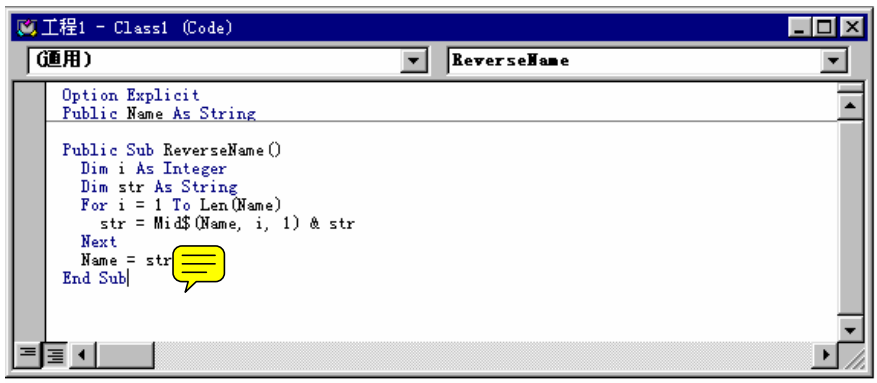

图 9.1　类 Class1 创建窗口

该类定义了一个全局变量Name和一个过程ReverseName，该过程用于逆转Name指定的字符串。

类模块有两个事件：Initialize和Terminate。从类模块的“对象”下拉列表中，选择“类”便在“过程”列表中显示这2个事件的事件名。一般Initialize事件包含了对象创建时所需的所有代码。Terminate事件包含撤销对象后进行清理的所有代码。本例不设计这两个事件过程。

在该工程中添加一个普通的窗体Form1，在其中添加两个标签label1和label2，它们的标题分别为“原来的字符串”和“逆转后字符串”；再添加两个文本框text1和text2；最后添加一个命令按钮command1，其标题为“逆转”，并在该命令按钮上设计如下事件过程：

```
Private Sub Command1_Click()
   Set myobject = New Class1
   myobject.Name = Text1.Text
   myobject.ReverseName
   Text2.Text = myobject.Name
```

```
End Sub
```

本窗体的设计屏幕如图9.2所示。启动本工程，出现Form1窗体的屏幕，在文本框text1中输入字符串“0123456789”，然后单击“逆转”命令按钮，则文本框text2中出现字符串“9876543210”，如图9.3所示。

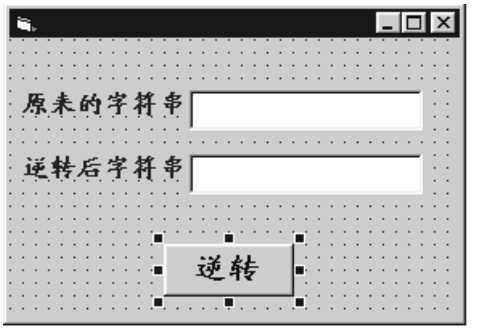

图 9.2　Form1 窗体设计界面

图 9.3　Form1 窗体执行界面

我们从上述Command1_Click()事件过程代码来讨论类的使用方式。第一行语句用于创建类Class1的一个对象myobject。这个对象中的Name变量变成私有变量，必须加上“myobject.”前缀才能使用。第二行语句给该变量赋初值。第三行语句调用myobject对象的ReverseName方法将Name变量之值逆转。第四行语句将变量Name的值在文本框text2中显示出来。

类模块和标准模块的不同点在于存储数据方法的不同。标准模块的数据只有一个备份，这意味着标准模块中一个公共变量的值改变以后，在后面的程序中再读取该变量时，它将得到同一个值。而类模块的数据，是相对于类实例（也就是由类创建的每一个对象）而独立存在的。

同样，标准模块中的数据在程序作用域内存在，也就是说，它存在于程序的存活期中；而类实例中的数据只存在于对象的存活期，它随对象的创建而创建，随对象的撤销而消失。

最后，当变量在标准模块中声明为Public时，则它在工程中任何地方都是可见的；而类模块中的Public变量，只有当对象变量含有对某一类实例的引用时才能访问。因此，在类模块中所有用于外部调用的变量或过程都要定义为Public型，否则在外部无法调用它们。

9.2　在类中添加属性和方法

类的默认接口由类的属性、方法和事件组成。与此相对应的是类的内部私有数据、过程和函数。它们被类的封装隐藏起来。属性、方法和事件对外界是可见的。一般来说，属性表示对象的数据，方法表示对象能执行的动作。或者说，属性提供了对象的描述，而方法则是它的行为。

注意：事件不属于默认接口。事件是调出接口(即该接口可连接其他对象)，而属性和方法属于调入接口(即该接口的成员被其他对象引用)。

9.2.1　向类中添加属性

定义类属性的最简单的方法，是向类的模块添加公共变量。例如，只要在名为Student的类中声明两个公共变量，可非常简单的定义该类的属性：

```
Public name As String
Public age As Integer
```

给类创建私有数据也很简单，只须声明一个Private变量，它就能被该类模块中的编码所访问：

```
Private f As Single
Private c As Single
```

1. 数据隐藏

数据隐藏是指保护对象的部分数据，而其余数据则以属性形式公开。这是面向对象的封装原则的一个方面。

数据隐藏可以改变类而不影响已存在的编码。例如，将Class类的私有变量x的类型从Integer改变为Long，而不会改变调用Class对象已存在的编码。

数据隐藏也允许定义只读属性。例如，可用Property Get属性过程返回一个私有变量，这就产生了属性过程。下一小节详细讨论属性过程。

2. 属性过程

如果只能通过声明公共变量来创建属性，那么就不能在更高层次上实现数据隐藏。例如，一个类Class有一个公共属性x，myobject是该类的对象，我们可以使用myobject.x将该属性设置为任意值。为了达到更高的数据隐藏，VB提供了属性过程，只能通过属性过程来获取属性值和对属性值进行修改。

实际上，当设置或检索属性值时，VB就执行对应属性过程的代码。

【例9.2】 创建一个类Class2说明属性过程的使用。

在本工程中添加一个类模块，并包含以下一段代码：

```
'pn为存储n属性的私有数据
Private pn As Integer
Public Property Get n() As Integer
    n = pn
End Property
Public Property Let n(ByVal Newn As Integer)
    If (Newn <= 0) Then
        MsgBox "n值不能小于等于0", vbOKOnly, "信息提示"
    Else
        pn = Newn
    End If
End Property
```

假设有一个名为myobject的变量，引用Class2对象。当执行x=myobject.n代码时，就会调用Property Get过程，返回存于类模块私有变量pn的值。当执行myobject.n=10代码时，将调用Property Let过程。

```
Set myobject = New Class2
myobject.n = 10          '调用Property Let过程使pn=10
x=myobject.n             '调用Property Get过程将pn赋给n使x=10
```

从中看到，属性过程可允许对象保护并验证自己的数据。这是使用公共变量的属性不能实现的。

9.2.2 属性过程与公共变量的比较

属性过程具有封装功能，它是如此强大的工具，以致于有时会觉得公共变量有点没用。对于编程来说答案是“有时是”。以下是一些基本原则。

（1）以下情况应使用属性过程：

- 属性为只读，或一旦设置就不能改变。
- 属性已设置的值需要验证。
- 超出特定范围的值，例如，负数虽符合属性的数据类型，但属性如果允许这样的假设值出现，就会导致程序出错。
- 属性的设置可导致一些对象状态的明显变化，例如Visible属性。
- 属性设置可改变内部变量或其他属性的值。

（2）以下情况应使用只读属性的公共变量：

- 属性是自验证类型。例如，如果把一个不是True或False的值赋给Boolean变量，

则或者出错，或者数据自动转换。

- 在数据类型所支持范围内的值都是有效的，像许多Single或Double类型的属性。
- 属性是String数据类型，并没有大小或字符串取值的限制。

注意：不要仅仅为了避免函数调用的额外开销而用公共变量来实现一个属性。其实，由于类型库的要求，VB在类模块中以任意方式将公共变量作为属性过程对来使用。

9.2.3　运行属性过程

VB提供了表9.1所列的三种属性过程。

表9.1　属性过程及其说明

属性过程	说明
Property Get	返回属性的值
Property Let	设置属性的值
Property Set	设置对象属性的值(即该属性含有对象引用)

从上表中看到，每一个属性过程都有自己特殊的功能。典型的属性由一对属性过程组成：Property Get检索属性的值，Property Let或Property Set给属性赋值。如果有Set语句，则VB调用Property Set，如果没有则调用Property Let。

属性过程可有多个参数，甚至可选参数。当使用多个参数时，属性过程对的参数必须匹配。表9.2给出了在属性过程声明中参数的要求。

表9.2　属性过程声明中参数的要求

属性过程	声明语法
Property Get	Property Get propertyname(参数$_1$,...，参数$_n$) As type
Property Let	Property Let propertyname(参数$_1$,...，参数$_n$，参数$_{n+1}$)
Property Set	Property Set propertyname(参数$_1$,...，参数$_n$，参数$_{n+1}$)

在Property过程中，（参数$_1$,...,参数$_n$）必须具有与过程相同的名称和相同的数据类型。就像其他过程类型一样，列表中的必需参数应排在第一个可选参数前。

注意：Property Get过程的声明比相关的Property Let或Property Set少一个参数。Property Get过程的数据类型声明必须与Property Let或Property Set过程的最后一个参数（参数$_{n+1}$）的数据类型一样。

例如，一个类似二维数组的属性的Property Let声明：

```
Private AValue As Variant
Public Property Let PropArr(ByVal X As Integer,_
        ByVal Y As Integer,ByVal NewValue As Variant)
```

```
  ...
  AValue=NewValue
End Property
```

Property Get过程的参数声明必须与Property Let过程的参数名称和数据类型相同：

```
Public Property Get PropArr(ByVal X As Integer,ByVal Y As Integer) As Variant
  ...
  PropArr(X,Y)=AValue
End Property
```

在Property Set声明中的最后参数的数据类型，要么是对象类型，要么是Variant类型。

图9.4给出了这些参数匹配规则的理由，并显示了VB如何将赋值语句部分与Property Let参数相匹配。

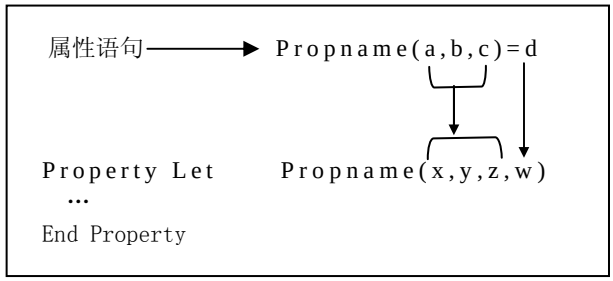

图 9.4　Property Let 过程的调用

要创建只读属性，只须简单地省略Property Let；对于对象属性，只须简单地省略Property Set。

【例9.3】　设计一个类Class3，采用属性过程的方法实现从华氏温度到摄氏温度之间的转换，它们之间的转换公式为：

摄氏温度＝（5.0／9.0）*（华氏温度－32.0）

然后设计一个窗体使用该类进行这两种温度的转换。

首先设计的Class3的类代码如下：

```
Private f As Single
Private c As Single

Public Property Get cs() As Single
  cs = c
End Property
Public Property Let cs(ByVal newc As Single)
  c = newc
End Property
Public Property Get fs() As Single
  fs = f
End Property
Public Property Let fs(ByVal newf As Single)
```

```
    f = newf
End Property
Public Sub trans()
    c = (5 / 9) * (f - 32)
End Sub
```

然后设计一个窗体Form3，其设计界面如图9.5所示。其中包含两个标签label1和label2、两个文本框text1和text2，另有一个命令按钮command1，在该命令按钮上设计如下事件过程：

```
Private Sub Command1_Click()
    Set myobject = New Class3
    myobject.fs() = Val(Text1.Text)
    myobject.trans
    Text2.Text = myobject.cs()
End Sub
```

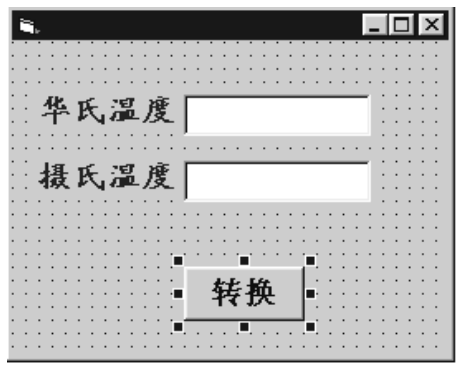

图 9.5　Form3 设计界面

将窗体Form3指定为启动对象，启动本工程，出现该窗体的屏幕，在“华氏温度”文本框中输入“100”，单击“转换”命令按钮，则在“摄氏温度”文本框中出现“37.77778”的结果，如图9.6所示。

图 9.6　Form3 执行界面

9.2.4 向类中添加方法

一般来说，属性是关于某个对象的数据，而方法则是该对象可能被要求去执行的动作。

类的方法就是所声明的Sub或者Function公共过程。例如，为了给Class类创建Comp方法，可将下面的Public Function过程添加到类模块中：

```
Public Function Comp() As Double
...
End Function
```

类的公共接口是由类模块的属性和方法声明来定义的。同数据隐藏一样，声明为Private的过程也不是接口的组成部分。这意味着可以更改类模块内部使用的实用程序的过程，而不影响使用该对象的代码。

更重要的是，更改方法实现的内部公共Sub或者Function过程的代码，不会影响使用该方法的代码。只要不改变过程参数的数据类型，或者不改变Function过程所返回的数据类型，接口就不会改变。

将对象的实现细节隐藏在接口之后，是封装的另一方面。封装可以增强方法的性能，或者完全改变方法的实现途径，而不必对使用该方法的代码进行更改。

9.2.5 把属性和方法设置为默认项

可以为自己的类所创建的对象赋以默认的属性，就像VB所提供的对象默认属性一样。执行步骤如下：

（1）在“工具”菜单上，选择“过程属性”选项，打开“过程属性”对话框。

（2）单击“高级”，扩展“过程属性”对话框。

（3）在“名称”框中，选择类的当前默认的属性或者方法。如果类当前没有任何默认的成员，那么跳到第(5)步。

（4）在“过程标识符”框中，选择“None”来删除属性或者方法的默认状态。

（5）在“名称”框中，选择想使之成为新的默认项的属性或方法。

（6）在“过程标识符”框中，选择“默认”，然后单击“确定”。

一个类只可能有一个默认成员。如果已经将某个属性或者方法标记为默认，那么在将另一属性或者方法设置为默认项之前，必须将原来的过程标识符复位为“None”。如果有两个成员被标记为默认的，那么不会出现任何编译错误，但是没有办法来预测VB将选择哪一个来作为默认项。

9.3 在类中添加事件

属性和方法属于入端接口，因为它们是从对象外面调用的。相对而言，事件被称为出

端接口，因为它们是在对象里边产生，在其他地方处理。本节讨论在类模块中如何声明和处理事件。

9.3.1　事件的声明与触发

我们仍以一个例子讨论事件的声明与触发。

【例9.4】　建立一个类和一个窗体，设计一个事件PercentDone用于显示完成任务的百分比。

我们在工程中创建一个类Class5。该类有一种方法，该方法可能花费很长时间来执行，同时还希望自己的应用程序能够显示出工作完成的进度。为此，可以让Class5对象显示一个用来表示完成百分比的对话框，但是以后在每个使用Class5对象的工程中，就会离不开那个对话框了，有时不希望这样。对象设计的一个较好的原则是让使用该对象的应用程序来处理用户接口。

Class5的作用是执行其他任务，因此合理的做法是：把一个PercentDone事件给它，然后让调用Class5方法的过程来处理那个事件。PercentDone事件也可以提供一种机制来取消任务。

现在，可以开始为该主题来建立代码示例了，操作方法是打开一个“标准.EXE”工程，将两个按钮和一个标签添加到“Form5”中。在“工程”菜单上，选择“添加类模块”，将一个类模块添加到工程中。如表9.3所示的那样给对象命名，其设计界面如图9.7所示。

表9.3　例子程序的对象属性设置

对象	属性	设置值
类模块	Name	Class5
第一个按钮	Caption	启动
第二个按钮	Caption	取消
标签	Name	label1
	Caption	“0%”

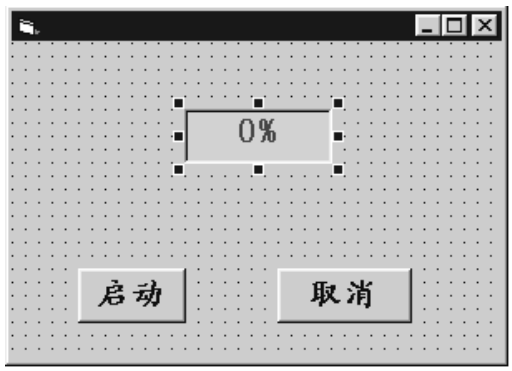

图 9.7　Form5 设计界面

在类模块的声明部分，用Event关键字来声明事件。事件可以有ByVal和ByRef参数，就

像Class5的PercentDone事件所演示的那样：

```
Option Explicit
Public Event PercentDone(ByVal Percent As Single, ByRef Cancel As Boolean)
```

当调用对象得到PercentDone事件时，Percent参数包含了任务完成的百分比。可以将ByRef Cancel参数设置为True，以取消触发该事件的方法。

PercentDone事件是由Class5类的LongTask方法引发的。LongTask方法接受两个参数：总的时间长度和产生PercentDone事件的最小时间间隔。该方法的代码如下：

```
Public Sub LongTask(ByVal Duration As Single, ByVal MinInterval As Single)
  Dim Threshold As Single
  Dim Start As Single
  Dim blnCancel As Boolean
  Start = Timer
  Threshold = MinInterval
  Do While Timer < (Start + Duration)
     '在实际应用程序中，每次循环时将会在这里做某些工作
     If Timer > (Start + Threshold) Then
        RaiseEvent PercentDone(Threshold / Duration, blnCancel)
        '检查一下，看操作是否被取消
        If blnCancel Then Exit Sub
        Threshold = Threshold + MinInterval
     End If
  Loop
End Sub
```

每隔MinInterval秒，使用RaiseEvent函数触发PercentDone事件。当该事件返回时，LongTask将检查blnCancel参数是否设置为True。如果为True，则退出该方法的执行。

9.3.2 处理事件的对象

触发事件的对象叫做事件源。为了处理事件源所触发的事件，可以用WithEvents关键字声明对象类的变量。

为了处理Class5的PercentDone事件，将下面的代码放置到Form5的声明部分：

```
Option Explicit
Private WithEvents myobject As Class5
Private mblnCancel As Boolean
```

WithEvents关键字指定：变量myobject将用于处理对象的事件。可以通过提供类名来指定对象类型，该类是创建这个对象的类。变量myobject是在“Form5”的声明部分所声明的，因为WithEvents变量必须是模块级的变量。变量mblnCancel将用于取消LongTask方法。

使用WithEvents变量时，应该注意下面这些限制条件：

- WithEvents变量不能是派生对象变量。也就是说，不能把它声明为As Object。当声明该变量时必须指定类名。
- 不能把WithEvents变量声明为As New。必须明确地创建事件源对象，并将它赋给WithEvents变量。
- 不能在标准模块中声明WithEvents变量。只能在类模块、窗体模块以及其他定义类的模块中声明。
- 不能创建WithEvents变量数组。

一旦声明了WithEvents变量，变量名就出现在模块“代码”窗口左边的下拉菜单上。当选择了myobject时，Class5类的事件将出现在右边下拉菜单上，如图9.8所示。

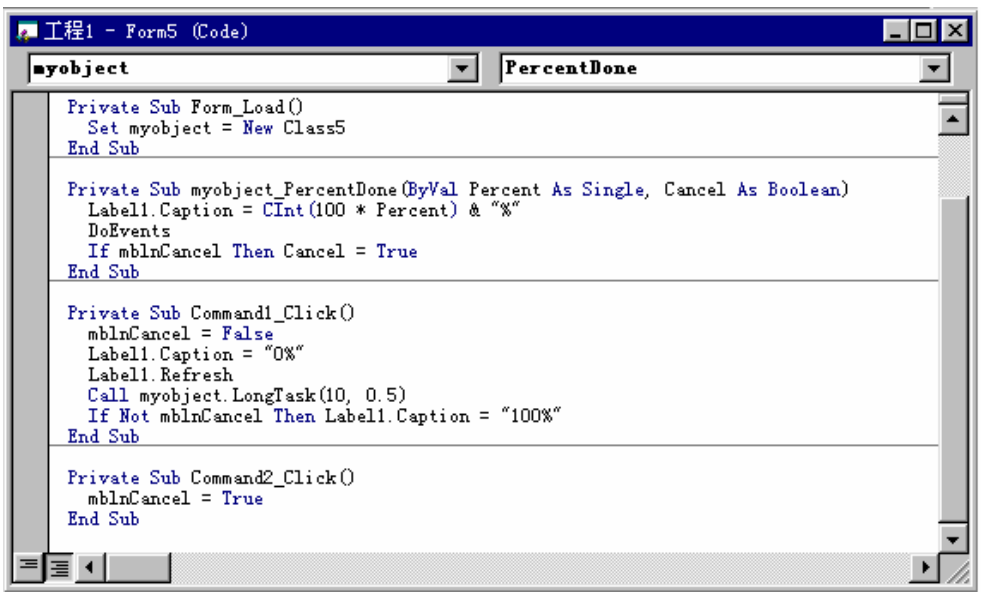

图 9.8　与 WithEvents 变量相关联的事件

选定一个事件，将显示相应的事件过程，以myobject_为前缀。所有跟WithEvents变量相关联的事件过程，都将以该变量名为前缀。将下面的代码添加到myobject_PercentDone事件过程中。

```
Private Sub myobject_PercentDone(ByVal Percent As Single, Cancel As Boolean)
  Label1.Caption = CInt(100 * Percent) & "%"
  DoEvents
  If mblnCancel Then Cancel = True
End Sub
```

不论何时，触发PercentDone事件时，事件过程就在Label1控件中显示完成的百分比。

DoEvents语句允许重新画出该标签，同时也给用户单击“取消”按钮的机会。将下面的代码添加到标题为“取消”按钮的Click事件中。

```
Private Sub Command2_Click()
  mblnCancel = True
End Sub
```

当LongTask正在运行时，如果单击了“取消”按钮，一旦DoEvents语句允许出现对事件的处理，那么将会执行Command2_Click事件。模块级的变量mblnCancel设置为True，myobject_PercentDone事件将对其进行测试，并将ByRef Cancel参数设置为True。

在设计时，声明WithEvents变量时，并没有任何对象与之相关联。WithEvents变量与任何其他对象变量相同。必须创建对象，并将对该对象的引用赋给这个WithEvents变量。

将下面的代码添加到Form_Load事件过程中，以创建myobject：

```
Private Sub Form_Load()
  Set myobject = New Class5
End Sub
```

当执行上面的代码时，VB将创建一个Class5对象，并将它的事件跟myobject相关联的事件过程连接起来。从那时起，一旦Class5产生了PercentDone事件，都将执行myobject_PercentDone事件过程。

为了调用LongTask方法，将下面的代码添加到标题为“启动”按钮的Click事件中。

```
Private Sub Command1_Click()
  mblnCancel = False
  Label1.Caption = "0%"
  Label1.Refresh
  Call myobject.LongTask(10, 0.5)
  If Not mblnCancel Then Label1.Caption = "100%"
End Sub
```

在调用LongTask方法之前，完成百分比的显示标签必须被初始化，而且必须将模块级的Boolean标记(其作用是取消该方法的执行)设置为False。

用10秒任务延迟调用LongTask，每隔0.5秒触发一次PercentDone事件。每次触发该事件时，都将执行myobject_PercentDone事件过程。

LongTask结束以后，都要测试mblnCancel，以确定LongTask是否正常结束，或者是否因为mblnCancel设置为True而停止。只有在前一种情况下，完成的百分比才会被更新。

将Form5设置为启动对象，运行本工程。再单击“启动”命令按钮。每次触发PercentDone事件，带有完成百分比的标签被更新，其执行界面如图9.9所示。单击“取消”命令按钮，就可以停止任务。

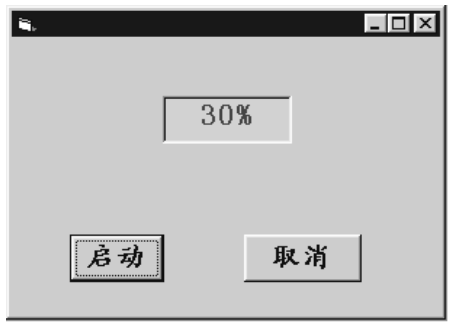

图 9.9　Form5 执行界面

通过将新Class5的引用赋给myobject变量，可以使得myobject变量为不同Class5对象处理事件。事实上，每次单击按钮时，通过添加下面两行代码，可以让Commandl中的代码来做这些：

```
Set myobject=New Class5          '<-新行
Call myobject.LongTask(10,0.5)
Set myobject=Nothing             '<-新行
```

每次按下按钮时，上面的代码都将创建一个新的myobject。一旦LongTask方法结束，对Class5的引用就会通过将myobject设置为Nothing而释放，而且这个Class5也被撤销。

WithEvents变量每次只能包含一个对象引用，因此如果将不同的Class5对象赋给myobject，先前Class5对象的事件将不会再得到处理。如果myobject是惟一包含对旧Class5引用的对象变量时，那么该对象将被撤销。

不论何时，当赋给变量myobject的Class5对象产生事件时，都将调用跟myobject相关联的事件过程。为了终止事件处理，可以将myobject设置为Nothing，就像在以下代码块中所示的：

```
Set myobject=Nothing    '终止myobject事件处理
```

当把WithEvents变量设置为Nothing时，VB将断开对象的事件和与该变量相关联的事件过程的连接。

9.3.3　向窗体添加事件

本小节讨论如何给窗体创建自定义事件。

【例9.5】　建立一个类和一个窗体，在其中创建一个自定义事件Gong。

首先打开一个前面建立的工程文件，然后按照以下基本步骤进行操作：

（1）在“工程”菜单上，选择“添加类模块”，将类模块Class6添加到工程中。将下面的代码放置到Class6的“声明”部分：

```
Public Property Get Form6() As Form6
  Set Form6 = mForm6
```

```
End Property
Public Property Set Form6(ByVal NewForm6 As Form6)
  Set mForm6 = NewForm6
End Property
```

（2）创建一个新的窗体Form6，将下面的代码添加到Form6的“声明”部分：

```
Event Gong()
Private mc1 As Class6
```

因为已经创建了Class6，所以创建Class6类型的变量是允许的。

（3）返回Class6，并将下面的代码添加到“声明”部分。

```
Private WithEvents mForm6 As Form6
```

WithEvents关键字意味着Form6的这个实例是与事件相关联的。只有在创建了Gong事件后才能执行这一步。

（4）在Class6的“代码”窗口上的左边“对象”下拉菜单上，选择“mForm6”来为Gong事件得到事件过程。将下面的代码添加到该事件过程中：

```
Private Sub mForm6_Gong()
  MsgBox "你按一个键激活执行一次Gong事件过程!", vbOKOnly, "信息提示"
End Sub
```

（5）返回Form6。将下面的代码添加到事件过程中：

```
Private Sub Form_Load()
  Set mc1 = New Class6
  Set mc1.Form6 = Me
End Sub
```

第一行创建了一个Class6对象，而第二行则将对Form6的引用赋给其Form6属性(在第一步所创建的)。

(6)将一个文本框text1和一个命令按钮command1放置到Form6窗体上，如图9.10所示，并在该窗体上添加以下事件过程：

```
Private Sub Text1_Change()
    RaiseEvent Gong
End Sub
Private Sub Command1_Click()
    Set Form6 = Nothing
    End
End Sub
```

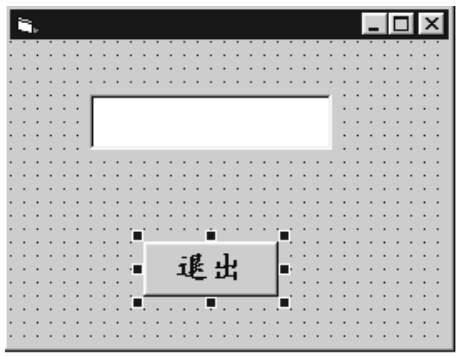

图 9.10　Form6 设计界面

每次改变该文本框的内容时，都将引发窗体执行Gong事件过程。

（7）将Form6设置为启动对象，运行本工程。在其中的文本框内键入字符“1”时，出现如图9.11所示的信息框。

图 9.11　Form6 执行界面

本例显示了怎样将一个事件添加到窗体中，然后从几个控件得到通知。下面我们总结一下将某个事件添加到一个类中，然后使用该事件的具体步骤：

（1）在定义类的类模块声明部分，用Event语句来声明事件，该事件带有希望它带有的任何参数。事件总是Public。

（2）在类模块代码中的合适地方，用RaiseEvent语句来引发事件，并提供所需要的参数。

（3）在将要处理事件的模块声明部分，使用WithEvents关键字，添加该类类型的变量。它必须是一个模块级的变量。

（4）在代码窗口左边的下拉菜单上，选择声明为WithEvents的变量。

（5）在代码窗口右边的下拉菜单上，选择希望处理的事件(可以为类声明多个事件)。

（6）使用所提供的参数，将代码添加到事件过程中。

9.4 习 题 9

9.1 编写一个程序，验证类的Initialize和Terminate事件的操作。

9.2 创建一个类Class2，用于计算两个字符串中子串在主串中的出现次数，并用信息框显示结果。然后设计一个如图9.12所示的窗体，利用该类进行子串次数计算。其执行结果如图9.13所示。

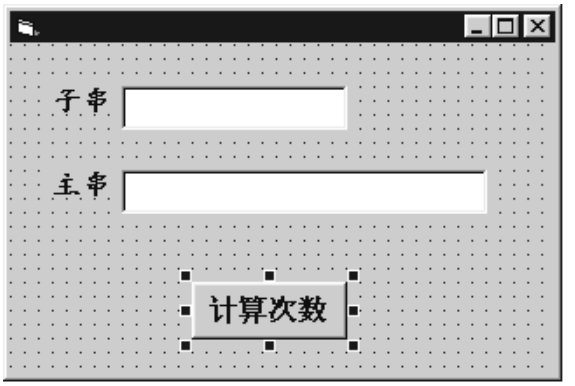

图 9.12 计算子串在主串中的出现次数

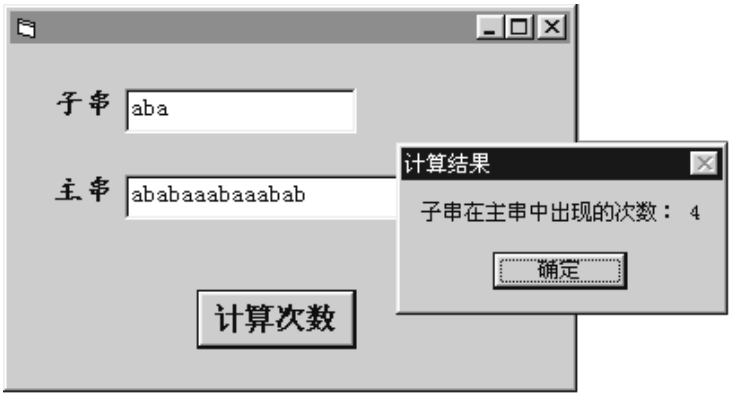

图 9.13 执行结果

9.3 将例9.3采用公共变量的方式实现属性。

9.4 编写一个程序，给窗体增加一个改变大小的属性。

9.5 编写程序，增加一个“窗体居中”的属性，当该属性为True时，窗体位于屏幕中央。