

## 设计应用程序窗体

窗体是 Visual Basic 应用程序的基本构造模块，是应用程序运行时与用户交互操作的实际窗口。窗体对象有自己的属性、方法和事件，可以由属性定义其外观，由方法定义其行为，由事件定义其与用户的交互。通过设置窗体属性并编写响应事件的 Visual Basic 代码，就能定义出满足应用需要的窗体对象。窗体又是其他对象的载体或容器，几乎所有的控件都可以添加在窗体上。除窗体设计原理，还需要考虑应用程序的开始与结束，有一些技巧可以用来决定应用程序启动时的外观。熟悉应用程序卸载时进行的一些处理也是很重要的。

在本项目中，将通过一组任务详细地了解窗体的基本属性、常用事件和方法，学习和掌握创建标准窗体、各种对话框以及 MDI 应用程序的方法。

### 项目目标

- 掌握窗体的常用属性、方法和事件
- 掌握用函数创建输入框和消息框的方法
- 掌握用控件创建各种标准对话框的方法
- 掌握创建 MDI 应用程序的方法

### 任务 3.1 在窗体上显示色彩变化的阴影字

在项目 2 中，曾经多次使用 Print 方法在窗体上显示文本。默认情况下，窗体总是呈现灰色的背景，而文本总是用系统默认的字（宋体）来显示的，至于显示文本的位置则是通过在 Print 方法中使用 Tab(n)或 Spc(n)参数来控制的。在本任务中将创建一个标准 EXE 应用程序，在程序运行时窗体背景将被设置为白色，当单击窗体时将在窗体的指定位置显示不同颜色的阴影字，程序运行结果如图 3.1 和图 3.2 所示。



图 3.1 在窗体上显示红色阴影字



图 3.2 在窗体上显示绿色阴影字



## 任务目标

- 理解窗体的 BackColor、ForeColor、FontName 和 FontSize 属性
- 理解窗体的 CurrentX 和 CurrentY 属性
- 掌握窗体的 Print 和 Cls 方法
- 掌握窗体的 Load 事件和 Click 事件



## 任务分析

在 Visual Basic 中, 通过执行 Print 方法可以在窗体上显示文本信息。窗体的背景颜色可以用窗体的 BackColor 属性来设置, 文本颜色可以用窗体的 ForeColor 属性来设置, 显示文本所用的字体和字号分别用窗体的 FontName 和 FontSize 属性来设置, 对文本显示位置的控制则可以通过设置窗体的 CurrentX 和 CurrentY 属性来实现。阴影字效果是通过在相邻位置上以两种不同颜色显示文本来生成的。



## 任务实施

- (1) 在 Visual Basic 6.0 中创建一个标准 EXE 工程。
- (2) 将窗体 Form1 的 Caption 属性设置为“在窗体上显示阴影字效果”。
- (3) 在窗体 Form1 的代码窗口中编写一个名为 ShowText 的通用过程, 代码如下。

```
Private Sub ShowText(X As Long, Y As Long, Text As String, Color As Long)
```

```
    Me.CurrentX = X
    Me.CurrentY = Y
    Me.ForeColor = Color
    Print Text
```

```
End Sub
```

- (4) 在窗体 Form1 的代码窗口中编写该窗体的 Click 和 Load 事件过程, 代码如下。

```
Private Sub Form_Click()
    Dim msg As String, i As Integer
    Dim Colors

    msg = "Visual Basic 程序设计"
    Colors = Array(vbRed, vbGreen, vbBlue, vbMagenta, vbCyan)
    Cls
    Me.FontName = "方正隶书行简体"
    Me.FontSize = 32
    ShowText 500, 800, msg, vb3DShadow
    Randomize '初始化随机数生成器
    i = CInt(5 * Rnd)
    ShowText 450, 750, msg, CLng(Colors(i))

    Me.FontName = "微软雅黑"
    Me.FontSize = 16
    ShowText 1800, 2000, "请单击窗体继续...", vbBlack
```

```
End Sub

Private Sub Form_Load()
    Me.BackColor = vbWhite
End Sub
```

(5) 将窗体文件保存为“Form03-01.frm”，工程文件保存为“工程 03-01.vbp”。



## 程序测试

- (1) 按 F5 键运行程序。
- (2) 用鼠标单击窗体，此时显示阴影字效果；再次单击窗体，文字颜色随机发生变化。
- (3) 单击窗口右上角的关闭按钮，以结束程序运行。



## 相关知识

### 1. 窗体的常用属性

窗体的属性可以在设计阶段通过属性窗口进行设置，也可以在运行阶段通过代码改变窗体的部分属性。也有一些窗体属性（如 Name）是不能通过代码来设置的，只能在设计阶段在属性窗口中进行设置，而不能在运行阶段通过代码改变该属性的值，这类窗体属性在程序运行阶段是只读的。在属性窗口中所做的设置大多数情况下可立即在窗体设计器中反映出来，而在代码中所做的设置只能在程序运行时才能显示设置结果。

077

窗体的常用属性如下。

- Caption：用于设置显示窗体的标题。在本任务中，通过属性窗口将窗体的 Caption 属性设置为“在窗体上显示阴影字效果”。
- BackColor：用于设置窗体的背景颜色，语法如下。

窗体名.BackColor = 颜色值

当前窗体名可以用关键字 Me 表示。

在设计阶段，可以利用系统提供的调色板进行设置。也可以在程序运行时对该属性值进行设置。表 3.1 和表 3.2 分别列出了常用的颜色常数和系统颜色。

表 3.1 常用的颜色常数

常 数	数 值	描 述
vbBlack	&H0	黑色
vbRed	&HFF	红色
vbGreen	&HFF00	绿色
vbYellow	&HFFFF	黄色
vbBlue	&HFF0000	蓝色
vbMagenta	&HFF00FF	洋红
vbCyan	&HFFFF00	青色
vbWhite	&HFFFFFF	白色

表 3.2 系统颜色

常 数	数 值	描 述
vbScrollBars	&H80000000	滚动条颜色
vbDesktop	&H80000001	桌面颜色
vbActiveTitleBar	&H80000002	活动窗口标题栏颜色
vbInactiveTitleBar	&H80000003	非活动窗口标题栏颜色
vbMenuBar	&H80000004	菜单背景颜色
vbWindowBackground	&H80000005	窗口背景颜色
vbWindowFrame	&H80000006	窗口框架颜色
vbMenuText	&H80000007	菜单上文字的颜色
vbWindowText	&H80000008	窗口内文字的颜色
vbTitleBarText	&H80000009	标题, 尺寸框和滚动箭头内文字的颜色
vbActiveBorder	&H8000000A	活动窗口边框的颜色
vbInactiveBorder	&H8000000B	非活动窗口边框的颜色
vbApplicationWorkspace	&H8000000C	多文档界面(MDI)应用程序的背景颜色
vbHighlight	&H8000000D	控件内选中项的背景颜色
vbHighlightText	&H8000000E	控件内选中项的文字颜色
vbButtonFace	&H8000000F	绘在命令按钮正面的颜色
vbButtonShadow	&H80000010	绘在命令按钮边缘的颜色
vbGrayText	&H80000011	变灰的(无效的)文字
vbButtonText	&H80000012	掀压按钮上文字的颜色
vbInactiveCaptionText	&H80000013	非活动标题内文字的颜色
vb3Dhighlight	&H80000014	三维显示元素的高亮颜色
vb3DDKShadow	&H80000015	三维显示元素的最暗阴影颜色
vb3Dlight	&H80000016	低于 vb3Dhighlight 的三维次高亮颜色
vb3Dface	&H8000000F	文字表面的颜色
vb3DShadow	&H80000010	文字阴影的颜色
vbInfoText	&H80000017	提示窗内文字的颜色
vbInfoBackground	&H80000018	提示窗内背景的颜色

078

- ForeColor：用于设置窗体的前景颜色，该属性改变窗体中图形和文本的颜色。
- FontName：用于设置窗体中显示文本所用的字体。在属性窗口中无法对该属性进行设置，但可以通过 Font 属性设置文本的字体、大小、样式等。
- FontSize：用来设置窗体中显示文本所用的字体的大小。在属性窗口中，可以通过 Font 属性对该属性进行设置。
- CurrentX、CurrentY：用来设置下一次打印或绘图方法的水平或垂直坐标。这些属性在设计时是不可用的。在窗体中进行绘制图形或输出结果时，经常要使用 CurrentX 和 CurrentY 属性来设置或返回当前坐标的水平坐标和垂直坐标。

设置颜色属性时，也可以利用 RGB 函数生成颜色值，语法格式如下。

RGB(red, green, blue)

其中参数 red、green 和 blue 分别表示颜色的红色、绿色和蓝色成分，它们取值范围都是从 0 到 255。

## 2. 窗体的常用方法

窗体的常用方法如下。

- Print：用于在窗体或图片框上输出文本，语法格式如下。

对象.Print 输出的内容列表

- Cls：用于清除在运行时窗体或图片框中显示的文本和图形，语法格式如下。

对象.Cls

- Move：用于移动窗体或控件，语法如下。

对象.Move 左, 上, 宽, 高

其中“对象”指定要移动的窗体或控件。“左”是必需的单精度值，指示对象左边的水平坐标（x-轴）。“上”是可选的单精度值，指示对象顶边的垂直坐标（y-轴）。“宽”和“高”均为可选的单精度值，分别指定对象的新宽度和新高度。

## 3. 窗体的常用事件

下面介绍窗体的几个常用事件。

### (1) Click 事件

Click 事件是在一个对象上按下然后释放一个鼠标按钮时发生。该事件也会发生在一个控件的值改变时。对一个窗体对象来说，该事件是在单击一个空白区或一个无效控件时发生，语法格式如下。

```
Private Sub Form_Click()
```

### (2) DblClick 事件

当在一个对象上按下和释放鼠标按钮并再次按下和释放鼠标按钮时，DblClick 事件发生。对于窗体而言，当双击被禁用的控件或窗体的空白区域时，该事件发生，语法格式如下。

```
Private Sub Form_DblClick()
```

说明：如果在 Click 事件中有代码，则 DblClick 事件将永远不会被触发，因为 Click 事件是两个事件中首先被触发的事件。其结果是鼠标单击被 Click 事件截断，从而使 DblClick 事件不会发生。

### (3) Load 事件

在一个窗体被装载时发生 Load 事件。当使用 Load 语句启动应用程序，或引用未装载的窗体属性或控件时，发生此事件，语法格式如下。

```
Private Sub Form_Load()
```

通常利用 Load 事件过程来设置窗体启动时的初始属性。在本任务中，通过 Load 事件过程设置了窗体的背景颜色。

## 任务 3.2 保持图片在窗体上居中

在本任务中创建一个标准 EXE 程序，其功能是在窗体上显示一幅图片，要求在改变窗体大小时该图片始终保持显示在窗体中央，如图 3.3 所示。当关闭窗体时，弹出一个对话框，提示确认退出程序信息，如果单击“确定”按钮，则退出程序；如果单击“取消”按钮，则返回程序，如图 3.4 所示。



## 任务目标

- 掌握窗体的 Resize、Unload 和 QueryUnload 事件
- 理解窗体的 ScaleHeight 和 ScaleWidth 属性
- 掌握用 PaintPicture 方法在窗体上绘制图像的方法



图 3.3 保持图片上窗体上居中



图 3.4 确认退出程序



## 任务分析

除了使用 Print 方法在窗体上显示文本,也可以使用 PaintPicture 方法在窗体上显示图片。要在窗体上显示一幅图片,需要声明一个 Picture 对象,并使用 LoadPicture 函数将文件中的图片加载到 Picture 对象中。要保证窗体大小发生变化时图片始终保持在窗体的中央,则需要在编写窗体的 Resize 事件过程中,使用 PaintPicture 方法在指定位置上绘制图片文件的内容。若要在退出程序时弹出对话框,则应当编写窗体的 Unload 或 QueryUnload 事件过程。



## 任务实施

- (1) 在 Visual Basic 集成开发环境中创建一个标准 EXE 工程。
- (2) 将窗体 Form1 的 Caption 属性设置为“居中显示图片”。
- (3) 在窗体 Form1 的代码窗口编写 Resize 事件过程,代码如下。

```
Private Sub Form_Resize()  
    Dim pic As Picture  
    Dim x As Single, y As Single  
    Set pic = LoadPicture(App.Path & "\\风景.jpg")  
  
    Cls  
    x = (Me.ScaleWidth - pic.Width) / 2  
    y = (Me.ScaleHeight - pic.Height) / 2  
    Me.PaintPicture pic, x, y, pic.Width, pic.Height  
  
End Sub
```

- (4) 在窗体 Form1 的代码窗口编写 Unload 事件过程,代码如下。

```
Private Sub Form_Unload(Cancel As Integer)  
    Dim choice As Integer  
    choice = MsgBox("确实要退出程序吗?", vbQuestion + vbOKCancel, "温馨提示")
```

```
If choice = vbCancel Then  
    Cancel = True  
End If  
End Sub
```

(5) 将窗体文件和工程文件分别保存为“Form03-02.frm”和“工程 03-02.vbp”。



## 程序测试

- (1) 按 F5 键运行程序，窗口中央显示一幅图片。
- (2) 通过拖动程序窗口的边框或角来调整窗口的大小，此时可以看到图片一起保持在窗口的中央。
- (3) 单击窗口右上角的关闭按钮，此时将弹出一个对话框，如果单击“确定”按钮，将退出程序；如果单击“取消”按钮，则返回程序。



## 相关知识

### 1. 窗体的 ScaleHeight 和 ScaleWidth 属性

窗体的 ScaleHeight 和 ScaleWidth 属性分别用来返回窗体内部的宽度和高度。这两个属性在设计时是不可用的，并且在运行时是只读的。

ScaleWidth 和 ScaleHeight 属性给出窗体的内部尺寸，不包括边框厚度以及菜单或标题等高度。而窗体的尺寸则由 Width 和 Height 属性决定。

窗体是控件的容器，控件在窗体上的位置坐标可由其 Top 和 Left 属性决定。控件的宽度和高度则由 Width 和 Height 属性决定。

### 2. 窗体的 Resize、Unload 和 QueryUnload 事件

窗体的 Resize 事件发生在窗体首次显示或改变窗体大小时，Unload 和 QueryUnload 事件则发生卸载窗体时。

#### (1) Resize 事件

当窗体第一次显示或窗体的外观尺寸被改变时发生 Resize 事件，语法格式如下。

```
Private Sub Form_Resize()
```

当窗体调整大小时，可用 Resize 事件过程来移动控件或调整其大小。也可用此事件过程来重新计算一些变量或属性（如 ScaleHeight 和 ScaleWidth 等），它们取决于该窗体的尺寸。

#### (2) Unload 事件

当窗体从内存中卸载时发生 Unload 事件。当窗体被加载时，它的所有控件的内容均被重新初始化。当使用在控制菜单中的关闭命令、关闭按钮或 Unload 语句关闭该窗体时，此事件被触发，语法格式如下。

```
Private Sub Form_Unload(Cancel As Integer)
```

其中参数 Cancel 为整数，其默认值为 0，用来确定窗体是否卸载。如果 Cancel 为 0，则窗体被卸载。将 Cancel 设置为任何一个非零的值可防止窗体被删除。

通过将参数 Cancel 设置为任何非零的值可以防止窗体被删除，但不能阻止其他事件发生，诸如从 Windows 操作环境中退出等。在本任务中，利用 Unload 事件的这一特性来实现关闭程序时提示确认关闭操作。



(3) QueryUnload 事件

在一个窗体或应用程序关闭之前发生 QueryUnload 事件，语法格式如下。

```
Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
```

其中参数 Cancel 是一个整数。如果将此参数设定为除 0 以外的任何值，则可以在所有已装载的窗体中停止 QueryUnload 事件，并阻止该窗体和应用程序的关闭。UnloadMode 是一个整数值或一个常数，它指示引起 QueryUnload 事件的原因。UnloadMode 参数的返回值在表 3.3 中列出。

表 3.3 UnloadMode 参数的返回值

常数	数值	描述
vbFormControlMenu	0	用户从窗体上的“控件”菜单中选择“关闭”指令
vbFormCode	1	Unload 语句被代码调用
vbAppWindows	2	当前 Microsoft Windows 操作环境会话结束
vbAppTaskManager	3	Microsoft Windows 任务管理器正在关闭应用程序
vbFormMDIForm	4	MDI 子窗体正在关闭，因为 MDI 窗体正在关闭
vbFormOwner	5	因为窗体的所有者正在关闭，所以窗体也在关闭

QueryUnload 事件的典型用法是在关闭一个应用程序之前用来确保包含在该应用程序中的窗体中没有未完成的任务。例如，如果还没有保存某一窗体中的新数据，则应用程序会提示保存该数据。

当一个应用程序关闭时，可以使用 QueryUnload 或 Unload 事件过程将 Cancel 属性设置为 True 来阻止关闭过程。当关闭窗口体对象时，QueryUnload 事件先于 Unload 事件发生。

3. Picture 对象

Picture 对象可以用于将各种图像赋值给具有 Picture 属性的对象。声明 Picture 对象的语法格式如下。

```
Dim pic As Picture
```

声明 Picture 对象后，可以使用 LoadPicture 函数将指定图片文件的内容加载到该对象中。例如，在本任务中就是这样处理的，代码如下。

```
Set pic = LoadPicture(App.Path & "\风景.jpg")
```

其中关键字 App 表示一个全局对象，通过它可以指定如下信息：应用程序的标题、版本信息、可执行文件和帮助文件的路径及名称，以及是否运行前一个应用程序的示例。App.Path 属性表示可执行文件的路径。

4. PaintPicture 方法

PaintPicture 方法可以用来在窗体 (Form)、图片框 (PictureBox) 或打印机 (Printer) 上绘制图片文件的内容。基本语法格式如下。

```
object.PaintPicture picture, x1, y1, width1, height1
```

其中参数 object 是一个对象表达式，其值为窗体、图片框或打印机对象。如果省略 object，则默认为带有焦点的窗体对象。picture 指定要绘制的图片来源。x1 和 y1 均为单精度值，指定在绘制图片的坐标。width1 和 height1 均为单精度值，指示图片的目标宽度和目标高度。



### 5. 用 MsgBox 函数显示消息框

调用 MsgBox 函数可以弹出一个对话框来显示消息, 此时将等待用户单击按钮, 并返回一个整数告诉用户单击的是哪个按钮。本任务调用 MsgBox 函数传递了 3 个参数, 代码如下。

```
choice = MsgBox("确实要退出程序吗?", vbQuestion + vbOKCancel, "温馨提示")
```

其中第 1 个参数指定要显示在对话框中的消息; 第 2 个参数指定在对话框中显示的图标样式、按钮的数目和形式, 上述语句指定显示“确定”、“取消”按钮和问号图标; 第 3 个参数给出在对话框标题栏中显示的字符串。如果用户单击“确定”按钮, 则 MsgBox 函数返回整数 vbOK (1); 如果用户单击“取消”按钮, 则返回整数 vbCancel (2)。

## 任务 3.3 用鼠标在窗体上绘制图形

在本任务中将创建一个简单的绘画程序, 可以通过拖动鼠标左键在窗体上连续画线, 释放鼠标左键时则停止画线, 如图 3.5 所示; 如果单击鼠标右键, 则可以从上次画图位置出发画一条线段, 如图 3.6 所示。



图 3.5 画任意线条



图 3.6 画连续直线

083



### 任务目标

- 掌握窗体的 Line 方法
- 掌握窗体的各种鼠标事件



### 任务分析

要用鼠标在窗体上绘制图形, 需要声明一些窗体级变量, 用于保存在窗体上画线时的鼠标坐标。有了这些窗体级变量, 无论在哪一个鼠标事件过程中, 都可以获得鼠标的当前位置坐标。用窗体的鼠标事件 MouseMove 和MouseDown 完成连续画线, 用鼠标事件 MouseUp 结束画线, 用窗体的鼠标事件 MouseDown 和 MouseMove 实现从上次画图位置到当前画直线。



### 任务实施

- (1) 在 Visual Basic 集成开发环境中新建一个标准 EXE 工程。

(2) 把窗体 Form1 调整到所需大小, 将其 Caption 属性设置为“简单绘图程序”。

(3) 在窗体 Form1 的代码窗口中编写以下代码。

'声明窗体级变量

```
Private mouseBX As Integer, mouseEX As Integer      '用于保存鼠标指针位置
Private mouseBY As Integer, mouseEY As Integer
Private NYcrw As Boolean                            '用于判断是否开始画线
```

(4) 在窗体 Form1 的代码窗口中编写该窗体的 DblClick 事件过程, 代码如下。

```
Private Sub Form_DblClick()
    Me.Cls                                           '双击窗体时清除窗体内容
End Sub
```

(5) 在窗体 Form1 的代码窗口中编写该窗体的 Load 事件过程, 代码如下。

```
Private Sub Form_Load()
    NYcrw = False                                    '将画线状态设为"否"
End Sub
```

(6) 在窗体 Form1 的代码窗口中编写该窗体的 MouseDown 事件过程, 代码如下。

'按下鼠标按钮时执行以下事件过程

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

'按下鼠标左键开始连续画线

```
If Button = 1 Then
    mouseBX = X
    mouseBY = Y
    NYcrw = True
End If
```

'按下鼠标右键画直线

```
If Button = 2 Then
    Form1.Line -(X, Y)
End If
End Sub
```

(7) 在窗体 Form1 的代码窗口中编写该窗体的 MouseMove 事件过程, 代码如下。

'在窗体上移动鼠标时执行以下事件过程

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

'当画线状态为 True 时, 开始连续画线

```
If NYcrw = True Then
    Form1.Line (mouseBX, mouseBY)-(X, Y)
    mouseBX = X
    mouseBY = Y
End If
End Sub
```

'释放鼠标按钮时执行以下事件过程

```
Private Sub Form_MouseUp(Button As Integer, Shift As Integer, X As Single,
Y As Single)
    NYcrw = False
End Sub
```

(8) 将窗体文件保存为“Form03-04.frm”, 工程文件保存为“工程 03-04.vbp”。



## 程序测试

- (1) 按 F5 键运行程序。
- (2) 通过拖动鼠标左键连续画线，释放鼠标左键时停止画线。
- (3) 通过单击鼠标右键，从上次画图位置出发绘制一条线段。
- (4) 双击鼠标左键，清除当前窗体内容。



## 相关知识

### 1. 窗体的MouseDown、MouseUp 和MouseMove 事件

窗体的MouseDown、MouseUp 两个事件是当按下(MouseDown)或者释放(MouseUp)鼠标按钮时发生的事件，语法格式如下。

```
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
Private Sub Form_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
```

MouseMove 事件是当鼠标指针在屏幕上移动时发生的事件。当鼠标指针处在窗体和控件的边框内时，窗体和控件均能识别 MoveMove 事件，语法格式如下。

```
Private Sub Form MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
```

在上述鼠标事件过程中，参数 Button 返回一个整数，用来标识该事件的产生是按下(MouseDown)或者释放(MouseUp)哪个按钮引起的。Button 参数的值可以是 1、2 和 4，分别对应于左按钮、右按钮以及中间按钮。

参数 Shift 返回一个整数，在 button 参数指定的按钮被按下或者被释放的情况下，该整数相应于 Shift、Ctrl 和 Alt 键的状态。Shift 的值分别等于 1、2 和 4，分别代表 Shift、Ctrl 和 Alt 键被按下，Shift 参数指示这些键的状态。这些键也可以组合按下，例如，Ctrl 和 Alt 键都被按下，则 Shift 的值就是 6。

参数 X 和 Y 返回一个指定鼠标指针当前位置的数。

MouseDown 和 MouseUp 事件过程不同于 Click 和 DblClick 事件，鼠标事件被用来识别和响应各种鼠标状态，并把这些状态看作独立的事件。在按下鼠标按钮并释放时，Click 事件只能把此过程识别为一个单一的操作——单击操作，而 MouseDown 和 MouseUp 事件能够区分出鼠标的左、右和中间按钮。也可以为使用 Shift、Ctrl 和 Alt 等键编写用于鼠标加键盘组合操作的代码。这是 Click 和 DblClick 事件做不到的。

MouseDown 是 3 种鼠标事件中最常用的事件。例如，在运行时可用它调整控件在窗体上的位置，也可用它实现某些图形效果，按下鼠标按钮时就可触发此事件。在本任务中，就是使用 MouseDown 事件和 Line 方法来实现绘图的。

关于 MoveMove 事件。当指针移过屏幕时要调用多少次 MoveMove 事件？换句话说，指针由屏幕顶端移动到底端时将经过多少个位置？实际上并不是对鼠标经过的每个像素，Visual Basic 都会生成 MoveMove 事件。操作环境每秒生成有限多个鼠标消息。

为了看到实际上有多少次识别 MoveMove 事件，可用下述代码在每次识别 MoveMove 事件之处，应用程序都绘制一个小圆圈。

```
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single,
```

```

Y As Single)
    Line -(X, Y)
    Circle (X, Y), 100
End Sub

```

运行结果如图 3.7 所示。

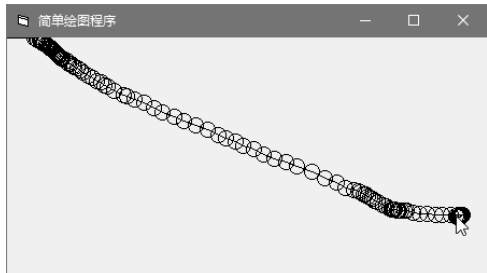


图 3.7 MouseMove 事件发生位置演示

通过测试可以知道,指针移动越快,则在任何两点之间所能识别的 MouseMove 事件越少。如果有很多圆圈挤在一起,则表明鼠标在这个位置移动缓慢。应用程序能够迅速识别大量的 MouseMove 事件。因此,一个 MouseMove 事件过程不应去做那些需要大量计算的工作。

## 2. 窗体的 Line 方法

Line 方法用于在窗体上画直线和矩形,语法格式如下。

窗体名.Line(x1, y1) - (x2, y2), [颜色], [B] [F]

其中参数(x1,y1)是可选的,其数值类型为 Single,指定直线或矩形的起点坐标。如果省略,线起始于由 CurrentX 和 CurrentY 指示的位置。

参数(x2, y2)是必需的,其数值类型为 Single,指定直线或矩形的终点坐标。

参数 color 可选的,其数值类型为 Long,画线时用的 RGB 颜色。如果省略该参数,则使用 ForeColor 属性值。可用 RGB 函数或 QBColor 函数指定颜色。

B 是可选的。如果包括,则利用对角坐标画出矩形。

F 是可选的。如果使用了 B 选项,则 F 选项规定矩形以矩形边框的颜色填充。不能不用 B 而用 F。如果不用 F 只用 B,则矩形用当前的 FillColor 和 FillStyle 填充。FillStyle 的缺省值为 transparent。

画连接的线时,前一条线的终点就是后一条线的起点。线的宽度取决于 DrawWidth 属性值。在背景上画线和矩形的方法取决于 DrawMode 和 DrawStyle 属性值。执行 Line 方法时,CurrentX 和 CurrentY 属性被参数设置为终点。

086

## 任务 3.4 设计键盘按键代码测试程序

在本任务中创建了一个键盘测试程序。在程序运行期间,当用户在键盘上按下任一个键时,将在窗体上显示出该键的代码和相应的字符并显示 Shift、Ctrl 和 Alt 键的状态,当按 Esc 键时则退出程序,运行情况如图 3.8 所示。



### 任务目标

- 理解窗体的 KeyPreview 属性
- 掌握窗体的各种键盘事件

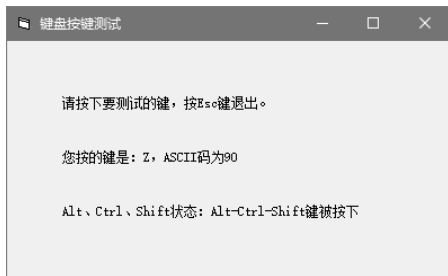


图 3.8 键盘测试程序运行结果

## 任务分析

在程序设计过程中,获取相关按键的信息将有助于使用键盘控制程序的运行。要获取某个按键的代码和相关信息,需要用到窗体对象的3个键盘事件,即KeyDown、KeyUp和KeyPress。利用每个事件响应不同的按键,以获得所有键盘按键的事件响应,并通过标签控件来显示相关按键的信息。

## 任务实施

(1) 在 Visual Basic 集成开发环境中新建一个标准 EXE 工程。

(2) 把窗体 Form1 调整到所需大小,将该窗体的 Caption 属性设置为“键盘按键测试”,将其 KeyPreview 属性设置为 True。

(3) 在窗体 Form1 上添加三个标签控件 Label1、Label2 和 Label3,把它们调整到合适的位置和大小,然后对它们的 Caption 属性值进行设置。

(4) 在窗体 Form1 的代码窗口中编写窗体的 KeyPress 事件过程,代码如下。

```
Private Sub Form_KeyPress(KeyAscii As Integer)
    '判断是否按下了 Esc 键,若是,则结束程序
    If KeyAscii = vbKeyEscape Then
        Unload Me
    Else
        '否则,在标签 Label2 中显示所按键的字符和代码
        Label2.Caption = "您按的键是:" & Chr(KeyAscii) & ", ASCII 码为" & KeyAscii
    End If
End Sub
```

087

(5) 在窗体 Form1 的代码窗口中编写窗体的 KeyDown 事件过程,代码如下。

```
Private Sub Form_KeyDown(KeyCode As Integer, Shift As Integer)
    '通过 Shift 的值判断是否 Shift、Ctrl、Alt 键是否被按下
    Select Case Shift
        Case 1
            Label3.Caption = "Alt、Ctrl、Shift 状态:" & "Shift 键被按下"
        Case 2
            Label3.Caption = "Alt、Ctrl、Shift 状态:" & "Ctrl 键被按下"
        Case 3
            Label3.Caption = "Alt、Ctrl、Shift 状态:" & "Ctrl-Shift 键被按下"
        Case 4
            Label3.Caption = "Alt、Ctrl、Shift 状态:" & "Alt 键被按下"
        Case 5
            Label3.Caption = "Alt、Ctrl、Shift 状态:" & "Alt-Shift 键被按下"
        Case 6
            Label3.Caption = "Alt、Ctrl、Shift 状态:" & "Alt-Ctrl 键被按下"
        Case 7
            Label3.Caption = "Alt、Ctrl、Shift 状态:" & "Alt-Ctrl-Shift 键被按下"
        Case Else
            Label3.Caption = "Alt、Ctrl、Shift 状态:未按"
    End Select
End Sub
```

(6) 在窗体 Form1 的代码窗口中编写窗体的 KeyUp 事件过程,代码如下。

```
Private Sub Form_KeyUp(KeyCode As Integer, Shift As Integer)
    Form_KeyDown KeyCode, Shift
End Sub
```

(7) 将窗体文件保存为“Form03-04.frm”，工程文件保存为“工程 03-04.vbp”。



## 程序测试

(1) 按 F5 键运行程序，在键盘上按任意键，此时窗体中部标签显示所按键的字符和对应的代码。

(2) 当按下键盘上的 Shift、Ctrl 和 Alt 键时，下方标签显示按下的具体键名，松开后显示“Alt、Ctrl、Shift 状态：未按”。

(3) 单击程序窗口右上角的关闭按钮或按下 Esc 键，结束程序运行。



## 相关知识

### 1. 窗体的 KeyPreview 属性

窗体的 KeyPreview 属性是用来返回或设置一个值，以决定是否在控件的键盘事件之前激活窗体的键盘事件，这些键盘事件包括 KeyDown、KeyUp 和 KeyPress。KeyPreview 属性可在设计阶段通过属性窗口进行设置，在本任务中就是采用这种方法。也可以在运行阶段通过代码改变设置，语法如下：

```
窗体名.KeyPreview = boolean
```

其中 boolean 为布尔表达式，指定如何接收事件。如果设置为 True，则窗体先接收键盘事件，然后是活动控件接收事件；如果设置为 False（默认值），则活动控件接收键盘事件，而窗体不接收。

用 KeyPreview 属性可以生成窗体的键盘处理程序，例如，应用程序利用功能键时，需要在窗体级处理击键，而不是为每个可以接收击键事件的控件编写程序。

如果窗体中没有可见和有效的控件，它将自动接收所有键盘事件。

若要在窗体级处理键盘事件、而不允许控件接收键盘事件时，也可以在窗体的 KeyPress 事件中设置参数 KeyAscii 为 0，在窗体的 KeyDown 事件中设置参数 KeyCode 为 0。

需要注意的是，一些控件能够拦截键盘事件，导致窗体不能接收它们。例如，当命令按钮控件有焦点时的 Enter 键以及焦点在列表框控件上时的方向键。

### 2. 窗体的 KeyPress 事件

KeyPress 事件是当用户按下和松开一个 ANSI 键时发生（ANSI 是可见的字符，相应的 ASCII 码为 1~127），语法格式如下。

```
Private Sub Form_KeyPress(KeyAscii As Integer)
```

其中 KeyAscii 是返回一个标准数字 ANSI 键代码的整数。

具有焦点的对象接收该事件。一个窗体仅在 KeyPreview 属性被设置为 True 时才能接收该事件。一个 KeyPress 事件可以引用任何可打印的键盘字符，一个来自标准字母表的字符或少数几个特殊字符之一的字符与 Ctrl 键的组合，以及 Enter 或 Backspace 键。KeyPress 事件过程在截取击键时，它可立即测试击键的有效性或在字符输入时对其进行格式处理。在本任务中，通过 Chr(KeyAscii)函数将 KeyPress 事件获得的标准数字 ANSI 键代码转变成



可显示的字符并在标签中显示。

### 3. 窗体的 KeyUp 和 KeyDown 事件

在 KeyPress 处理不了的功能可以由 KeyDown 和 KeyUp 事件来处理, 这些事件是当一个对象具有焦点时按下 (KeyDown) 或松开 (KeyUp) 一个键时发生, 语法格式如下。

```
Private Sub object_KeyDown(KeyCode As Integer, shift As Integer)
Private Sub object_KeyUp(KeyCode As Integer, shift As Integer)
```

其中 KeyCode 是一个键代码, 诸如 vbKeyF1 (F1 键) 或 vbKeyHome (Home 键); 参数 Shift 是在该事件发生时响应 Shift、Ctrl 和 Alt 键的状态的一个整数。Shift、Ctrl 和 Alt 键在这些位分别对应于值 1、2 和 4。例如, 如果 Ctrl 和 Alt 这两个键都被按下, 则 Shift 的值为 6。

对于这两个事件来说, 带焦点的对象都接收所有击键。一个窗体只有在不具有可视的和有效的控件时才可以获得焦点。KeyDown 和 KeyUp 事件可应用于大多数键, 通常应用于扩展的字符键如功能键、定位键、键盘修饰键和按键的组合、区别数字小键盘和常规数字键; 在需要对按下和松开一个键都响应时, 可使用 KeyDown 和 KeyUp 事件过程。

在下列情况下, 不能引用 KeyDown 和 KeyUp 事件: 窗体有一个命令按钮控件且 Default 属性设置为 True 时的 Enter 键。窗体有一个命令按钮控件且 Cancel 属性设置为 True 时的 Esc 键、Tab 键。

089

## 任务 3.5 创建输入框和消息框

在本任务中创建一个应用程序, 当运行程序时首先弹出一个输入框提示输入用户名, 如图 3.9 所示; 输入用户名并单击“确定”按钮后, 在窗体上显示用户显示欢迎信息, 如图 3.10 所示; 当关闭窗口时将弹出一个消息框, 单击“确定”按钮退出程序, 单击“取消”按钮则返回窗口, 如图 3.11 所示。



图 3.9 通过输入框输入用户名



图 3.10 在窗体上显示欢迎信息

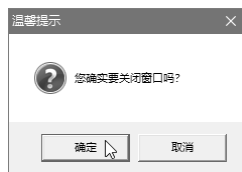


图 3.11 弹出消息框



### 任务目标

- 掌握用 InputBox 函数创建输入框的方法
- 掌握用 MsgBox 函数创建消息框的方法
- 掌握窗体的 Activate 事件的应用

电子工业出版社有限公司  
版权所有 盗版必究



## 任务分析

要想实现这个任务的目标,需要在程序窗体被激活以前通过 Load 事件过程调用 InputBox 函数来获取用户输入的信息;得到用户信息后,可以使用窗体的 Activate 事件在窗体激活时对该用户显示欢迎信息。



## 任务实施

- (1) 在 Visual Basic 集成开发环境中新建一个标准 EXE 工程。
- (2) 把窗体 Form1 调整到所需大小,将其 Caption 属性设置为“创建输入框和消息框”。
- (3) 在窗体 Form1 的代码窗口中声明一个窗体级变量,代码如下。

```
Private Username As String
```

- (4) 在窗体 Form1 的代码窗口中编写该窗体的 Activate 事件过程,代码如下。  
'激活窗口时执行以下事件过程

```
Private Sub Form_Activate()  
    Print: Print  
    Print Tab(4); "欢迎" & Username & "使用本程序!"  
End Sub
```

- (5) 在窗体 Form1 的代码窗口中编写该窗体的 Load 事件过程,代码如下。  
'加载窗体时执行以下事件过程

```
Private Sub Form_Load()  
    Me.FontSize = 20  
    '将 InputBox 函数的返回值赋给变量 Username  
    Username = InputBox("请输入您的尊姓大名:", "欢迎光临")  
End Sub
```

- (6) 在窗体 Form1 的代码窗口中编写该窗体的 Load 事件过程,代码如下。  
'卸载窗体时执行以下事件过程

```
Private Sub Form_Unload(Cancel As Integer)  
    Dim Choice As Integer  
    Choice = MsgBox("您确实要关闭窗口吗?", vbQuestion + vbOKCancel, "温馨提示")  
    If Choice = vbCancel Then  
        Cancel = True        '不卸载窗体  
    End If  
End Sub
```

- (7) 将窗体文件保存为“Form03-05.frm”,工程文件保存为“工程 03-05.vbp”。



## 程序测试

- (1) 按 F5 键运行程序。
- (2) 当出现消息框提示“请输入您的尊姓大名”时,输入用户名并单击“确定”按钮。
- (3) 此时出现程序主画面,并在窗体上显示“欢迎 XXX 使用本程序”。
- (4) 当单击窗口关闭按钮时,弹出一个消息框,单击“确定”按钮结束程序,单击“取消”按钮则继续程序的运行。



## 相关知识

### 1. 窗体的 Activate 事件

当一个窗体成为活动窗口时发生窗体的 Activate 事件，语法格式如下。

```
Private 窗体名_Activate()
```

一个对象通过用户单击，或使用代码中的 Show、SetFocus 方法等操作而变成活动的。Activate 事件仅当一个对象可见时才发生。

窗体的 Load 事件和 Activate 事件的区别。当程序载入（Load）一个窗体的 Visible 属性为 False 的窗体时不产生 Activate 事件，只有当使用 Show 方法将该窗体 Visible 属性设置为 True 时产生 Activate 事件。Load 事件在 Activate 事件之前发生，Load 事件在看不到窗体时就已经发生了，一般是对窗体进行初始化，而 Activate 事件是看到窗体时（当前窗体）才发生。

### 2. 用户对话框

在设计基于 Windows 的应用程序时，对话框用来提示用户应用程序继续运行所需的数据或者向用户显示信息。对话框是一种特殊类型的窗体对象，创建对话框有以下 3 种方法。

- 使用 MsgBox 或 InputBox 函数的代码可以创建预定义对话框。
- 使用标准窗体或自定义已存在的对话框创建自定义对话框。
- 使用 CommonDialog 控件可创建标准对话框，例如“打印”对话框和“打开文件”对话框等。

由于大多数对话框需要用户的交互作用，所以通常显示为模态对话框。在继续使用应用程序的其他部分以前，必须关闭（隐藏或卸载）模态对话框。例如，若在切换到其他窗体或其他对话框前必须单击“确定”或者“取消”，那么这个对话框就是模态对话框。

非模态对话框不需要关闭就可以使焦点在该对话框和其他窗体之间移动。对话框显示时，可以在当前应用程序的其他地方继续工作。非模态对话框很少；通常因为应用程序继续前需要响应才显示对话框。Visual Basic 中“编辑”菜单的“查找”对话框就是一个非模态对话框。使用非模态对话框显示常用的命令或信息。在本任务中介绍创建对话框的第一种方式。

### 3. MsgBox 函数

MsgBox 函数在对话框中显示消息，等待用户单击按钮，并返回一个整数表明用户单击了哪一个按钮，语法格式如下。

```
MsgBox(prompt[, buttons] [, title] [, helpfile, context])
```

其中参数 prompt 是必需的，它是一个字符串表达式，指定显示在对话框中的消息。该参数的最大长度大约为 1024 个字符，由所用字符的宽度决定。如果该参数的内容超过一行，则可以在每一行之间用回车符（Chr(13)）换行符（Chr(10)）或者是回车符与换行符的组合（Chr(13) & Chr(10)）将各行分隔开来。

参数 buttons 是可选的，该参数为数值表达式是值的总和，指定显示按钮的数目及形式、使用的图标样式、默认按钮是什么以及消息框的强制回应等。如果省略 buttons，则默认值为 0。buttons 参数的设置值在表 3.3 中列出。

表 3.3 buttons 参数的设置值

符号常量	数 值	描 述
vbOKOnly	0	只显示“确定”按钮
vbOKCancel	1	显示“确定”和“取消”按钮
vbAbortRetryIgnore	2	显示“终止”、“重试”和“忽略”按钮
vbYesNoCancel	3	显示“是”、“否”和“取消”按钮
vbYesNo	4	显示“是”和“否”按钮
vbRetryCancel	5	显示“重试”和“取消”按钮
vbCritical	16	显示 Critical Message 图标
vbQuestion	32	显示 Warning Query 图标
vbExclamation	48	显示 Warning Message 图标
vbInformation	64	显示 Information Message 图标
vbDefaultButton1	0	第一个按钮是默认值
vbDefaultButton2	256	第二个按钮是默认值
vbDefaultButton3	512	第三个按钮是默认值

参数 title 是可选的，用于指定在对话框标题栏中显示的字符串表达式。如果省略 title，则将应用程序名放在标题栏中。

参数 helpfile 是可选的，该参数为字符串表达式，识别用来向对话框提供上下文相关帮助的帮助文件。如果提供了 helpfile，则必须提供 context。

参数 context 参数是可选的。数值表达式，由帮助文件的作者指定给适当的帮助主题的帮助上下文编号。如果提供了 context，则必须提供 helpfile。

当提供 helpfile 与 context 时，用户可以按 F1 来查看与 context 相应的帮助主题，此时会在对话框中添加一个“帮助”按钮。

MsgBox 函数的返回值在表 3.4 中列出。

表 3.4 MsgBox 函数的返回值

常 数	数 值	返回的按钮
vbOK	1	OK (“确定”)
vbCancel	2	Cancel (“取消”)
vbAbort	3	Abort (“放弃”)
vbRetry	4	Retry (“重试”)
vbIgnore	5	Ignore (“忽略”)
vbYes	6	Yes (“是”)
vbNo	7	No (“否”)

如果对话框显示“取消”按钮，则按下 Esc 键与单击“取消”按钮的效果相同。如果对话框中有“帮助”按钮，则对话框中提供有上下文相关的帮助。但是，直到其他按钮中有一个被单击之前，都不会返回任何值。

如果还要指定第一个命名参数以外的参数，则必须在表达式中使用 MsgBox。为了省略某些位置参数，必须加入相应的逗号分界符。

#### 4. InputBox 函数

InputBox 函数在一个对话框中来显示提示，等待用户输入文本或按下按钮，并返回包含文本框内容的字符串，语法格式如下。

```
InputBox(prompt[,title][,default][,xpos][,ypos][,helpfile,context])
```

其中参数 prompt 是必需的，指定作为对话框消息出现的字符串表达式。prompt 的最大长度大约是 1024 个字符，由所用字符的宽度决定。如果 prompt 包含多个行，则可在各行之间用回车符 (Chr(13))、换行符 (Chr(10)) 或回车换行符的组合 (Chr(13) & Chr(10)) 来分隔。

参数 title 是可选的，指定显示在对话框标题栏中的字符串表达式。如果省略 title，则把应用程序名放入标题栏中。

参数 default 是可选的，指定显示文本框中的字符串表达式，在没有其他输入时作为缺省值。如果省略 default，则文本框为空。

参数 xpos 是可选的，该参数为数值表达式，与 ypos 一起成对出现，指定对话框的左边与屏幕左边的水平距离。如果省略 xpos，则对话框会在水平方向居中。

参数 ypos 是可选的，该参数为数值表达式，与 xpos 一起成对出现，指定对话框的上边与屏幕上边的距离。如果省略 ypos，则对话框被放置在屏幕垂直方向距下边大约三分之一处。

参数 helpfile 是可选的，该参数为字符串表达式，识别帮助文件，用该文件为对话框提供上下文相关的帮助。如果已提供 helpfile，则必须提供 context。

参数 context 是可选的。该参数为数值表达式，由帮助文件的作者指定给某个帮助主题的帮助上下文编号。如果已提供 context，则必须要提供 helpfile。

如果用户单击“确定”按钮或按下回车键，则 InputBox 函数返回文本框中的内容。如果用户单击“取消”按钮，则此函数返回一个长度为零的字符串 ("")。

如果还要指定第一个命名参数以外的参数，则必须在表达式中使用 InputBox 函数。如果要省略某些位置参数，则必须加入相应的逗号分界符。

### 任务 3.6 创建 Windows 通用对话框

在本任务中创建一个应用程序，用于演示各种 Windows 通用对话框的创建和使用。当程序运行时，通过单击“改变图标”按钮可以显示“打开”对话框，选择图标文件后将改变窗体标题栏上的图标并在窗体上显示图标文件的路径，如图 3.12 所示；通过单击“设置颜色”按钮可以打开“颜色”对话框，然后使用所选取的颜色改变窗体上文本的颜色，如图 3.13 所示；通过单击“设置字体”按钮可以打开“字体”对话框，可以用来设置窗体上文本的字体、文本样式、字号以及文本的颜色，如图 3.14 所示。

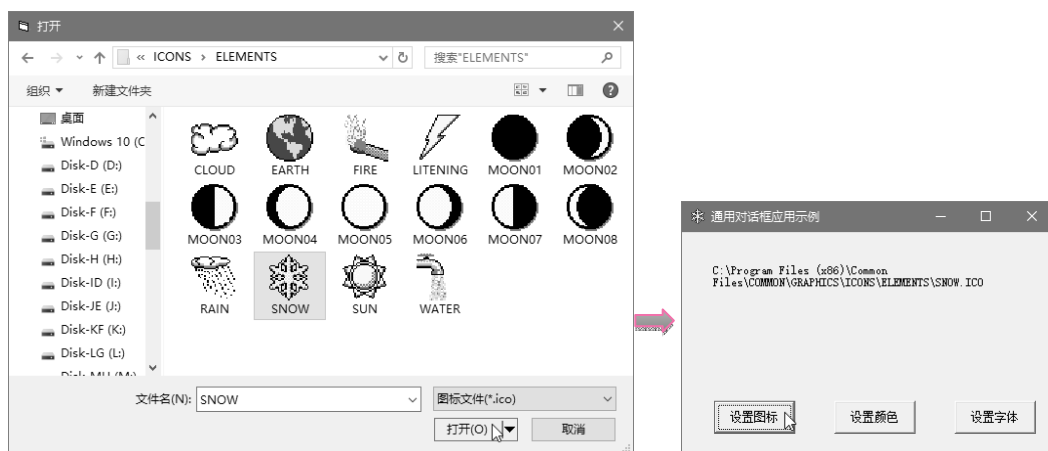


图 3.12 改变窗体图标

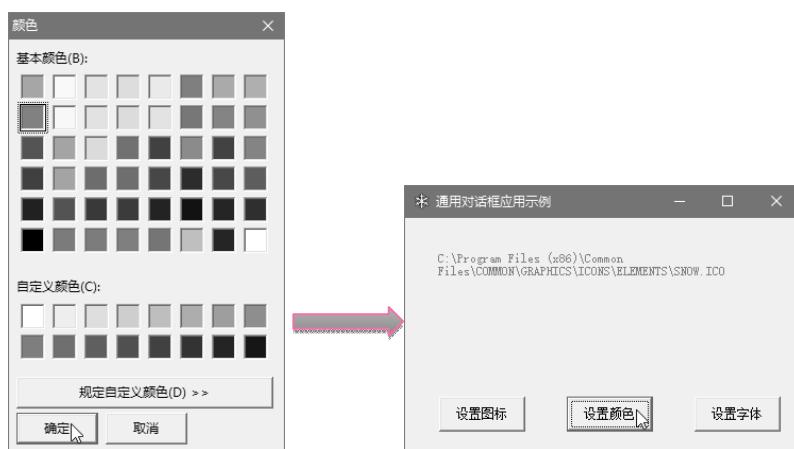


图 3.13 设置标签文本颜色



图 3.14 设置标签文本字体





## 任务目标

- 掌握添加 CommandDialog 控件的方法
- 理解 CommandDialog 控件的常用属性
- 掌握 CommandDialog 控件的常用方法



## 任务分析

CommandDialog 控件就是通用对话框,该控件提供打开文件、保存文件、设置打印选项、选择颜色和字体等操作的一组标准对话框。在本任务中,首先要将 CommandDialog 控件添加到工具箱中。除 CommandDialog 控件外,还要在窗体上添加 3 个命令按钮控件,通过单击这些按钮来显示相关对话框。要显示某个对话框,调用 CommandDialog 控件的相应方法即可。例如,调用 ShowOpen 方法可显示“打开”对话框;调用 ShowFont 方法可显示“字体”对话框;调用 ShowColor 方法可显示“颜色”对话框。



## 任务实施

- (1) 在 Visual Basic 集成开发环境中新建一个标准 EXE 工程。
- (2) 将窗体 Form1 的 Caption 属性设置为“通用对话框应用示例”。
- (3) 在窗体上添加一个标签控件 Label1,调整其位置和大小,并将其 Caption 属性清空。
- (4) 在工具箱中添加 CommandDialog 控件。从“工程”菜单中选择“部件”命令,在弹出的“部件”对话框的“控件”选项卡中选中“Microsoft Common Dialog Control 6.0 (SP6)”复选框,将该控件添加到工具箱中,如图 3.15 所示。
- (5) 在工具箱中双击 CommonDialog 控件图标,在窗体 Form1 上添加一个 CommonDialog 控件,其名称默认为 CommonDialog1,控件大小自动调整。

095

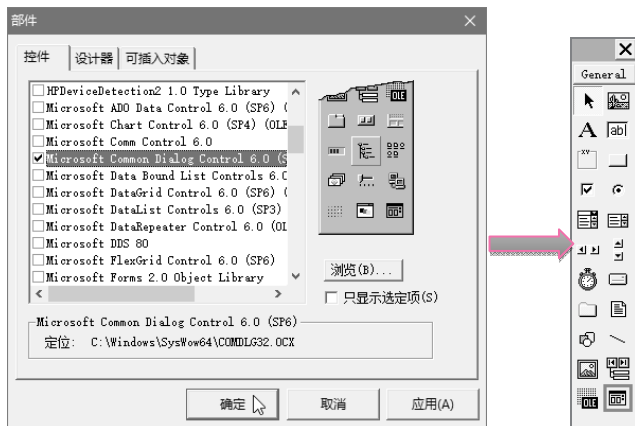


图 3.15 向工具箱添加 CommandDialog 控件

- (6) 在工具箱窗口中单击 CommandButton 控件图标,在窗体 Form1 上拖动鼠标以添加 3 个命令按钮控件,名称分别为 Command1、Command2、Command3,调整控件位置及大小;在属性窗口中将它们的 Caption 属性分别设为“改变图标”“设置颜色”“设置字体”。

至此,应用程序用户界面设计已完成,效果如图 3.16 所示。

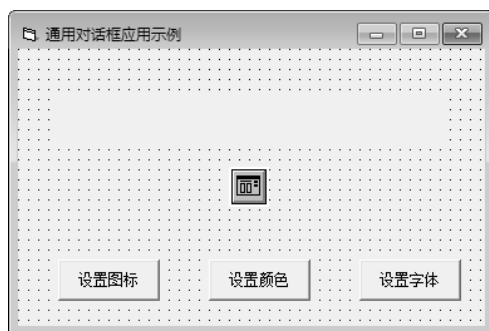


图 3.16 应用程序用户界面设计效果

(7)在窗体 Form1 的代码窗口中编写命令按钮 Command1 的 Click 事件过程,代码如下。

'单击“改变图标”按钮时执行以下事件过程

```
Private Sub Command1_Click()
    On Error GoTo nofile
    CommonDialog1.InitDir = "c:\"
    CommonDialog1.Filter = "图标文件(*.ico)|*.ico"
    CommonDialog1.CancelError = True

    CommonDialog1.ShowOpen
    Label1.Caption = Form1.CommonDialog1.FileName
    Form1.Icon = LoadPicture(CommonDialog1.FileName)
    Exit Sub
nofile:
    If Err.Number = 32755 Then
        Label1.Caption = "放弃操作"
    Else
        Label1.Caption = "其他错误"
    End If
End Sub
```

096

(8)在窗体 Form1 的代码窗口中编写命令按钮 Command2 的 Click 事件过程,代码如下。

'单击“设置颜色”按钮时执行以下事件过程

```
Private Sub Command2_Click()
    On Error GoTo nocolor
    CommonDialog1.CancelError = True
    CommonDialog1.ShowColor
    Label1.ForeColor = CommonDialog1.Color
nocolor:
End Sub
```

(9)在窗体 Form1 的代码窗口中编写命令按钮 Command3 的 Click 事件过程,代码如下。

'单击“设置字体”按钮时执行以下事件过程

```
Private Sub Command3_Click()
    On Error GoTo nofont
    CommonDialog1.CancelError = True
    '设置 CommonDialog 控件中与字体相关的属性
    CommonDialog1.Flags = cdlCFEffects Or cdlCFBoth
    CommonDialog1.FontName = Label1.FontName
```

```

CommonDialog1.FontSize = Label1.FontSize
CommonDialog1.FontBold = Label1.FontBold
CommonDialog1.FontStrikethru = Label1.FontStrikethru
CommonDialog1.FontUnderline = Label1.FontUnderline
CommonDialog1.FontItalic = Label1.FontItalic
'显示"字体"对话框
CommonDialog1.ShowFont
Label1.FontName = CommonDialog1.FontName
Label1.FontSize = CommonDialog1.FontSize
Label1.FontBold = CommonDialog1.FontBold
Label1.FontItalic = CommonDialog1.FontItalic
Label1.FontUnderline = CommonDialog1.FontUnderline
Label1.FontStrikethru = CommonDialog1.FontStrikethru
Label1.ForeColor = CommonDialog1.Color
nofont:
End Sub

```

(10) 将窗体文件保存为“Form03-06.frm”，工程文件保存为“工程 03-06.vbp”。



### 程序测试

- (1) 按 F5 键运行程序。
- (2) 通过单击“改变图标”命令按钮，弹出“打开”对话框，选择所需的图标文件，从而改变窗体标题栏上的图标，并在窗体上显示图标文件的路径。
- (3) 单击“设置颜色”命令按钮，弹出“颜色”对话框，可以改变窗体上文本的颜色。
- (4) 单击“设置字体”命令按钮，弹出“字体”对话框，可以用来设置窗体上文本的字体、样式、大小以及文本的颜色。

097



### 相关知识

#### 1. CommonDialog 控件

CommonDialog 控件提供一组标准的操作对话框，进行打开和保存文件，设置打印选项，以及选择颜色和字体等操作。在 Visual Basic 6.0 中，通用对话框控件不在标准控件工具箱中，需要添加“Microsoft Common Dialog Control 6.0 (SP6)”部件。

使用通用对话框时，可在窗体的任何位置添加一个 CommonDialog 控件，并对其进行属性设置。工程设计阶段在窗体上显示的 CommonDialog 控件图标，在程序运行时不会显示。程序运行时想要显示通用对话框，应该调用 CommonDialog 控件的以下方法。

- ShowOpen：显示“打开”对话框。
- ShowSave：显示“另存为”对话框。
- ShowColor：显示“颜色”对话框。
- ShowFont：显示“字体”对话框。
- ShowPrinter：显示“打印”对话框。
- ShowHelp 方法：显示“帮助”对话框。

##### (1) 显示“打开”对话框

通过调用通用对话框的 ShowOpen 方法，可以显示“打开”对话框，让用户从计算机

上选择要打开的文件。

需要指出的是,“打开”对话框并不能真正打开文件,而仅仅是让用户选择要打开的文件,至于后续处理必须通过编程来解决。

使用“打开”对话框时,需要设置通用对话框控件的以下属性。

- FileName: 指定在“文件名”文本框中初始显示的文件名,返回选定文件的标识符。
- FileTitle: 关闭对话框后,返回所选择不包括路径的文件名。
- Filter: 文件类型过滤器,用于设置对话框中的“文件类型”下拉式列表框中的项目及过滤显示的文件,Filter 属性值格式如下。

描述 1 | 过滤类型 1 | 描述 2 | 过滤类型 2 [ . . . ]

其中描述是指“文件类型”下拉式列表框项目的内容,过滤类型是指对话框中显示的文件类型。例如,在本任务中,为了在计算机上查找图标文件,可以将 Filter 属性设置为“图标文件(\*.ico)|\*.ico”。

- InitDir: 指定对话框打开时的默认路径。

#### (2) 显示“颜色”对话框

程序运行时,通过调用通用对话框的 ShowColor 方法或设置 Action 属性为 3,可以显示“颜色”对话框,“颜色”对话框可以让用户从调色板中选择颜色,并通过 Color 属性返回用户选定的颜色值。

#### (3) 显示“字体”对话框

通过调用通用对话框的 ShowFont 方法,可以显示“字体”对话框。“字体”对话框可以让用户设置应用程序所需要的字体。

使用“字体”对话框时,经常用到通用对话框控件的以下属性。

- Color: 返回在对话框中选定的颜色。
- FontBold: 返回在对话框中是否选定了粗体 (True、False)。
- FontItalic: 返回在对话框中是否选定了斜体 (True、False)。
- FontStrikethru: 返回在对话框中是否选定了下划线 (True、False)。
- FontName: 返回在对话框中选定的字体名称。
- FontSize: 返回在对话框中选定的字体大小。

## 2. 错误处理

在本任务中,进行错误处理时用到了 On Error 语句和 Err 对象。

#### (1) On Error 语句

On Error 语句用于设置错误程序,该语句有多种格式,本任务中使用了以下格式。

On Error GoTo nofile

本格式用于启动一个错误处理程序并指定该子程序在一个过程中的位置。也可以用来禁止一个错误处理程序。

#### (2) Err 对象

Err 对象是一个包括关于运行错误信息的对象,其主要属性是 Number 属性,该属性列出错误的序号,0 表示没有产生错误。

如果将通用对话框的 CancelError 属性设置为 True 时,则单击“取消”按钮将产生 32755 号错误。在本任务中,就是使用 CancelError 属性来判断是否放弃操作的。

## 任务3.7 创建MDI应用程序

在本任务中创建一个 MDI 应用程序，当启动该应用程序时首先出现一个快速显示窗体，如图 3.17 所示；当该窗体消失后出现一个 MDI 窗体，其中包含两个子窗体，如图 3.18 所示。



图 3.17 快速显示窗口

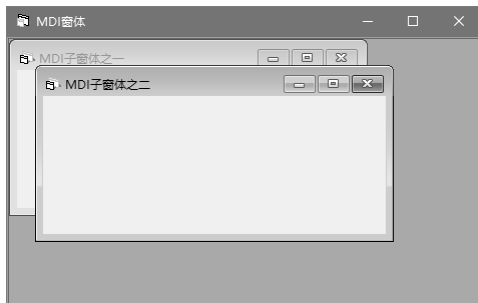


图 3.18 MDI 主窗口



### 任务目标

- 掌握创建 MDI 窗体的方法
- 掌握设置 MDI 子窗体的方法
- 掌握创建快速显示窗体的方法
- 掌握设置应用程序启动对象的方法

099



### 任务分析

Windows 应用程序按照界面可以分为两大类：单文档界面 (SDI) 和多文档界面 (MDI)。在 SDI 应用程序 (如 “记事本”) 中一次只能打开一个文档，如果希望打开另一个文档，就必须关闭前一个文档。在 MDI 应用程序 (如 Microsoft Office Word) 中允许同时打开多个文档，每个文档单独占用一个文档窗口，可以在不同的文档窗口之间切换。创建 MDI 应用程序时需要添加一个 MDI 窗体和若干个 MDI 子窗体，MDI 窗体可以使用菜单命令来添加，要创建一个 MDI 子窗体，只需要将标准窗体的 MDIChild 属性设置为 True 即可。本任务中的快速显示窗体其实就是一个无标题栏的标准窗体。



### 任务实施

- (1) 在 Visual Basic 集成开发环境中新建一个标准 EXE 工程。
- (2) 把窗体 Form1 调整到所需大小，将其 Caption 属性设置为 “MDI 子窗体之一”，将其 MDIChild 属性设置为 True，使该窗体成为一个 MDI 子窗体。
- (3) 添加 MDI 窗体。从 “工程” 菜单中选择 “添加 MDI 窗体” 命令，当出现 “添加 MDI 窗体” 对话框时，在 “新建” 选项卡中选择 “MDI 窗体”，然后单击 “打开” 按钮，如图 3.19 所示。
- (4) 将新添加 MDI 窗体的 Caption 属性设置为 “MDI 窗体”。

(5) 添加另一个 MDI 子窗体。从“工程”菜单中选择“添加窗体”命令，在“添加窗体”对话框中选择“新建”选项卡，选择“窗体”，然后单击“打开”按钮，如图 3.20 所示。



图 3.19 “添加 MDI 窗体”对话框



图 3.20 “添加窗体”对话框

(6) 将窗体 Form2 的 Caption 属性设置为“MDI 子窗体之二”，将其 MDIChild 属性设置为 True，使之也成为 MDI 子窗体。

(7) 将 MDI 窗体 MDIForm1 保存为 MDIForm03-07.frm，将两个 MDI 子窗体文件分别保存为“Form03-07a.frm”和“Form03-07b.frm”，将当前工程保存为“工程 03-07.vbp”。

(8) 从“工程”菜单中选择“添加窗体”命令，当出现“添加窗体”对话框时，从“新建”选项卡中选择“展示屏幕”，然后单击“打开”按钮，如图 3.21 所示。

(9) 新添加窗体的默认名称为 frmSplash，将该窗体文件保存为 frmSplash.frm；此窗体包含有框架、图像控件和标签控件，其显示效果如图 3.22 所示。



图 3.21 添加快速显示窗体



图 3.22 快速显示窗体默认外观

(10) 对窗体 frmSplash 上的控件的以下属性进行设置：将标签 lblCompany 的 Caption



属性修改为“ABC 公司”；将标签 lblCompanyProduct 的 Caption 属性修改为“MDI 应用程序”；将标签 lblPlatform 的 Caption 属性修改为“Windows 平台”；将标签 lblWarning 的 Caption 属性修改为“警告：本软件受版权法和国际条约保护”。

(11) 设置应用程序的启动对象。从“工程”菜单中选择“属性”命令，在“工程 1 - 工程属性”对话框中选择“通用”选项卡，从“启动对象”列表框中选择为 frmSplash，然后单击“确定”按钮，如图 3.23 所示。

此时，可以在工程资源管理器窗口看到应用程序包含的所有窗体，包括一个 MDI 父窗体、两个 MDI 子窗体和一个标准窗体，如图 3.24 所示。



图 3.23 为应用程序设置启动对象



图 3.24 工程资源管理器窗口

(12) 打开窗体 frmSplash 的代码窗口，对已有程序进行修改，代码如下（其中粗体字表示新添加的语句）。

```
' 出现快显窗口时，按任意键关闭快显窗口，显示 MDI 窗口
Private Sub Form_KeyPress(KeyAscii As Integer)
    Unload Me
    MDIForm1.Show
End Sub
' 加载快显窗口时，设置标签显示的内容
Private Sub Form_Load()
    lblVersion.Caption = "版本 " & App.Major & "." & App.Minor & "." & App.Revision
    lblProductName.Caption = "MDI 应用程序"
End Sub
' 在快显窗口单击框架 Frame1 时，关闭快显窗口，显示 MDI 窗口
Private Sub Frame1_Click()
    Unload Me
    MDIForm1.Show
End Sub
```

(13) 在窗体 MDIForm1 的代码窗口中编写该窗体的 Activate 事件过程。

```
'MDI 窗口成为活动窗口时显示两个子窗体
Private Sub MDIForm_Activate()
    ' 依次显示两个子窗体
    Form1.Show
    Form2.Show
End Sub
```

(14) 保存所有文件。



### 程序测试

- (1) 按 F5 键运行程序。
- (2) 当出现快速显示窗口时, 按任意键或用鼠标单击该窗口, 该窗口将被隐藏。
- (3) 出现一个多文档界面窗体, 其中包含两个子窗体。
- (4) 关闭 MDI 窗体, 此时两个子窗体也随之关闭。



### 相关知识

#### 1. 多文档界面 (MDI) 应用程序

MDI 应用程序具有以下特性。

- 所有子窗体均显示在 MDI 窗体的工作空间内。像其他窗体一样, 用户能移动子窗体和改变子窗体的大小, 不过, 它们被限制于这一工作空间内。
- 当最小化一个子窗体时, 它的图标将显示在 MDI 窗体上而不是在任务栏中。当最小化 MDI 窗体时, 此 MDI 窗体及其所有子窗体将由一个图标来代表。当还原 MDI 窗体时, MDI 窗体及其所有子窗体将按最小化之前的状态显示出来。
- 当最大化一个子窗体时, 它的标题会与 MDI 窗体的标题组合在一起并显示于 MDI 窗体的标题栏上。

#### 2. MDI 窗体的特有属性

MDI 窗体拥有一些普通窗口所没有的属性。

- AutoShowChildren: 通过设置该属性, 子窗体可以在窗体加载时自动显示或自动隐藏。
- ActiveForm: 该属性表示 MDI 窗体中的活动子窗体。

#### 3. 创建 MDI 应用程序

创建 MDI 应用程序的步骤如下。

- (1) 从“工程”菜单中选择“添加 MDI 窗体”命令。

注意: 一个应用程序只能有一个 MDI 窗体。如果工程已经有了一个 MDI 窗体, 则该“工程”菜单上的“添加 MDI 窗体”命令就不可使用。

- (2) 创建应用程序的子窗体。要创建一个 MDI 子窗体, 先创建一个新窗体 (或者打开一个存在的窗体), 然后把它的 MDIChild 属性设为 True。

(3) 设计时使用 MDI 子窗体。在设计时, 子窗体不是限制在 MDI 窗体区域之内。可以添加控件、设置属性、编写代码以及设计子窗体功能, 就像在其他 Visual Basic 窗体中做的那样。

#### 4. 快速显示窗体

快速显示窗口一般作为程序的封面使用, 这种窗口一般没有命令按钮, 没有标题栏。当出现快速显示窗口时, 按任意键或鼠标单击窗口, 它就会被卸载并调用应用程序主窗体。

制作快速显示窗体时, 在工程中新建一个窗体, 将窗体的 BorderStyle 属性设置为 3, ControlBox 属性设置为 False, Caption 属性设置为空字符串, 并在该窗体中添加一些文字和图片。此外, 还必须通过设置工程属性, 把快速显示窗体设置为应用程序的启动对象。

### 5. App 对象

App 对象是通过关键字 App 访问的全局对象，通过可以它指定以下信息：应用程序的标题、版本信息、可执行文件和帮助文件的路径及名称，以及是否运行前一个应用程序的示例。App 对象的常用属性如下。

- **CompanyName**：返回或设置一个字符串，该字符串包括运行中的应用程序的公司或创建者名称。该属性在运行时是只读的。
- **EXENAME**：返回当前正运行的可执行文件的根名（不带扩展名）。如果是在开发环境下运行，则返回该工程名。
- **Major**：返回或设置该工程的主要版本号。该属性在运行时是只读的。
- **Minor**：返回或设置该工程的小版本号。该属性在运行时是只读的。
- **Path**：返回或设置当前路径。该属性在设计时是不可用的，在运行时是只读的。
- **PrevInstance**：返回一个值，该值指示是否已经有前一个应用程序实例在运行。
- **Revision**：返回或设置该工程的修订版本号。该属性在运行时是只读的。
- **Title**：返回或设置应用程序的标题，该标题将显示在 Microsoft Windows 的任务列表中。如果在运行时发生改变，那么发生的改变不会与应用程序一起被保存。

## 项目小结

103

在本项目中介绍了 Visual Basic 6.0 窗体的常用属性、方法和事件的使用。

窗体的属性决定了窗体的外观。窗体的多数属性既可以在设计阶段通过属性窗口进行设置，也可以通过代码在程序运行阶段进行设置。常用的窗体属性有：Caption、BorderStyle、Height、Width、Top、Left 等。

通过对窗体事件的编程，可以实现在程序运行时对窗体进行控制，或进行人机交互。常用的窗体事件有：Load 事件、Resize 事件、鼠标事件、键盘事件、Unload 事件。

窗体的方法是窗体固有的能力，通过窗体的方法可以对窗体进行操作。常用的窗体方法有：Show、Hide、Cls、Print、PaintPicture 等。

窗体分为对话框、标准窗体、MDI 窗体以及 MDI 子窗体。对话框可以使用 InputBox 和 MsgBox 函数以及通用对话框控件来创建；标准窗体可以用于创建单文档界面应用程序；MDI 窗体和 MDI 子窗体可以用于创建多文档界面应用程序。

通过实施本项目，要求熟练掌握窗体的常用属性、方法和事件。

## 项目思考

### 一、选择题

1. 当调整窗体大小时将发生（ ）事件。

- A. Load
- B. Unload
- C. Resize
- D. Click

2. 当用户按下和松开一个 ANSI 键时发生（ ）事件。

A. Click

B. KeyUp

C. KeyPress

D. KeyDown

3. 在鼠标事件中, 若 Button 参数的值为 2, 则说明用户按下鼠标的 ( )。

A. 左按钮

B. 右按钮

C. 中间按钮

D. 左按钮和右按钮

4. 在鼠标事件中, 若 Shift 参数的值为 6, 则说明用户按下了 ( )。

A. Alt 键

B. Ctrl 键

C. Shift 键

D. Ctrl 键和 Alt 键

5. 如果用户在对话框中单击了“确定”按钮, 则 MsgBox 函数的返回值为 ( )。

A. 1

B. 2

C. 3

D. 4

6. 在工程资源管理器中, 图标 ( ) 表示 MDI 窗体。

A. B. C. D. 

## 二、判断题

1. ( ) 窗体或控件的名称可通过 Caption 属性来设置。

2. ( ) FontName 和 FontSize 属性用于设置显示文本所用的字体和字号。

3. ( ) BackColor 和 ForeColor 属性用于设置对象的背景颜色和前景颜色。

4. ( ) 若将 KeyPreview 属性设置为 True, 则活动控件接收键盘事件, 窗体不接收。

5. ( ) CurrentX 和 CurrentY 属性用来设置下一次打印或绘图的水平坐标和垂直坐标。

6. ( ) 要在对话框中显示问号图标, 可在 MsgBox 函数的第二个参数中包含符号常量 vbExclamation。

7. ( ) 如果在窗体的 Click 和 DblClick 事件过程中都编写了代码, 则单击窗体时依次执行这两个事件过程中的代码。

8. ( ) 当关闭窗体对象时, Unload 事件先于 QueryUnload 事件发生。

9. ( ) 通过调用 Move 方法可以移动窗体的位置。

10. ( ) 在“工程”菜单中选择“添加窗体”命令, 可以在当前工程中添加 MDI 子窗体。

## 项目实训

1. 创建一个应用程序, 要求使用 Print 方法在窗体中央显示一行文字。

2. 创建一个应用程序, 用于测试键盘按钮, 当在键盘上按下任一个键时, 在窗体上显示出该键的代码和相应的字符并显示 Shift、Ctrl 和 Alt 键的状态, 当按 Esc 键时退出程序。

3. 创建一个简单的绘画程序, 可以通过拖动鼠标左键连续画线, 释放鼠标左键时停止画线; 如果单击鼠标右键, 则画一条从上次画图位置出发的线段。

4. 创建一个应用程序时, 运行时首先弹出一个输入框提示输入用户名, 输入用户名并单击“确定”按钮后, 在窗体上显示对用户的欢迎信息, 关闭窗口时将弹出一个对话框, 单击“确定”按钮退出程序, 单击“取消”按钮则返回窗口。

5. 创建一个应用程序, 在窗体上包含一些按钮和标签, 通过单击这些按钮可以显示“打开”、“字体”和“颜色”对话框, 并用来设置窗体的图标、标签文字的字体、字号和颜色。

6. 创建一个 MDI 应用程序, 要求运行时显示 MDI 窗体和两个 MDI 子窗体。