

第 8 章 使用用户界面

用户界面是用户使用 VBA 处理数据时经常使用的对象。在用户需要使用数据交互时，用户界面是一个非常有用的工具。用户可以根据需要向用户窗体添加各种功能的控件，实现各种复杂的交互作用。

8.1 使用内置对话框

在 Excel 中，为了用户开发的便利，提供了多种固定功能的对话框。用户在实际开发过程中，根据需要调用对应的内置对话框。在本小节中，将结合具体的例子来说明如何使用内置对话框。

案例 153 使用“打开”对话框

1. 功能说明

在程序中需要交互式地打开某个文件时，如果使用 `InputBox` 函数让用户输入路径和文件名，容易出错并且不直观。在这个时候，用户可以选择使用对话框。

2. 语法说明

在 Excel VBA 中，使用 `Application` 对象的 `GetOpenFilename` 方法将打开标准的“打开”对话框，让用户在计算机中选择盘符、路径、文件类型和文件名等信息。其语法格式如下：

`表达式.GetOpenFilename(FileFilter, FilterIndex, Title, ButtonText, MultiSelect)`

该方法的参数都可省略，各参数的含义如下：

- **FileFilter**：一个指定文件筛选条件的字符串。

在 `FileFilter` 参数中传递的字符串由文件筛选字符串对以及后跟的 DOS 通配符文件筛选规范组成，中间以逗号分隔。每个字符串都在“文件类型”下拉列表框中列出。例如，下列字符串指定两个文件筛选——文本和加载宏：

“文本文件 (*.txt)、*.txt、加载宏文件 (*.xla)、*.xla”。

如果省略 `FileFilter`，则此参数默认为

“所有文件 (*.*)、*.*”

要为单个文件筛选类型使用多个通配符表达式，需用分号将通配符表达式分开。例如：

“Excel 文件(*.xls;*.xlsx;*.xlsm),*.xls;*.xlsx;*.xlsm,”

- **FilterIndex**：指定默认文件筛选条件的索引号，取值范围为 1 到由 `FileFilter` 所指定的筛选条件数目。如果省略该参数，或者该参数的值大于可用筛选条件数，则使用第一

个文件筛选条件。

- **Title:** 指定对话框的标题。如果省略该参数，则标题为“打开”。
- **ButtonText:** 在 PC 中不可用。
- **MultiSelect:** 如果为 **False**（默认值），则只允许选择一个文件名。如果为 **True**，则允许选择多个文件名，返回值是一个包含所有选定文件名的数组，即使在“打开”对话框中只选定了—个文件名，也将返回到—个数组中。

注意：当用户在该对话框中单击“打开”按钮时将返回选择的路径和文件名，但并不真正执行打开操作。

3. 案例说明

在本例中，将在程序代码中使用内置的“打开”对话框，选择用户需要查看的文件。然后将文件名显示在工作簿中。

4. 编写代码

使用“打开”对话框的程序代码如下：

```
Sub OpenDiag()  
    Dim sFilt As String  
    Dim sTitle As String  
    Dim sMsg As String  
    Dim sFname As Variant  
    Dim i As Integer  
    Dim sf As Variant  
  
    sFilt = "文本文件(*.txt),*.txt," & _  
        "所有文件(*.*),*.*"  
    sTitle = "打开文件"  
  
    sFname = Application.GetOpenFilename _  
        (filefilter:=sFilt, Title:=sTitle, MultiSelect:=True)  
    If Not IsArray(sFname) Then  
        MsgBox "请选择文件！"  
    Else  
        i = 1  
        ActiveSheet.Columns(1).Clear  
        For Each sf In sFname  
            sMsg = sMsg & sf & vbCrLf  
            ActiveSheet.Cells(i, 1) = sf  
            i = i + 1  
        Next  
    End If  
End Sub
```

5. 运行结果

运行程序代码，选择系统的文件类型，如图 8.1 所示。

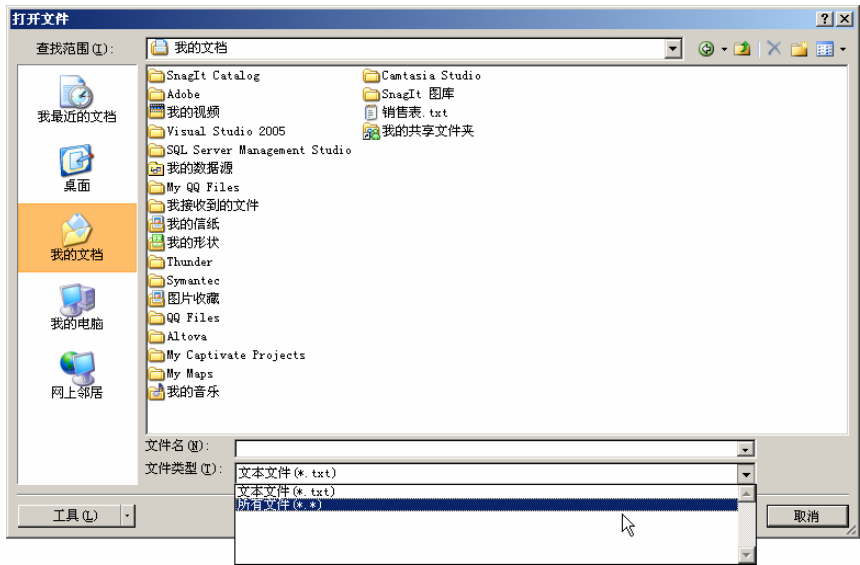


图 8.1 选择系统文件类型

选择多个文件，然后单击对话框中的“打开”按钮，如图 8.2 所示。



图 8.2 选择打开多个文件

关闭对话框，查看 Excel 文件中的保存信息，如图 8.3 所示。

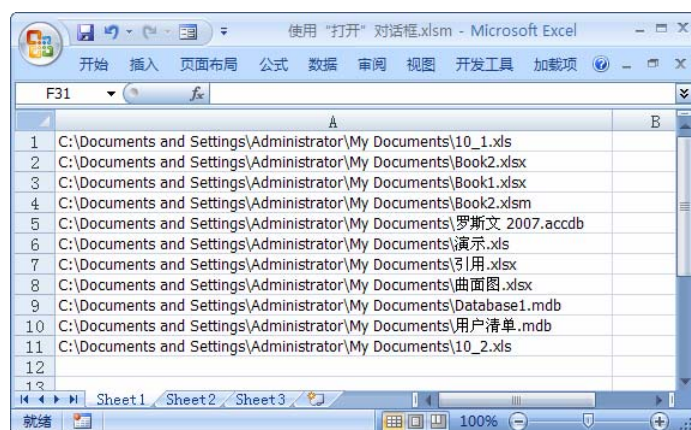


图 8.3 Excel 文件中的保存路径信息

6. 程序分析

在上面的代码中，首先显示一个“打开”对话框，选中的文件名将返回到数组中，使用 `IsArray` 函数检测返回值，如果不是数组则表示没有选择文件。最后使用循环语句将数组中的文件名逐个取出来填充到工作表中。

案例 154 使用“保存”对话框

1. 功能说明

和“打开”对话框类似，用户同样可以使用 VBA 代码调用“保存”对话框，用来保存文件。

2. 语法说明

与 `GetOpenFilename` 方法类似，使用 `Application` 对象的 `GetSaveAsFilename` 方法可打开标准的“另存为”对话框，在该对话框中用户可以选择（或输入）一个文件名。其语法格式如下：

```
表达式.GetSaveAsFilename(InitialFilename, FileFilter, FilterIndex, Title, ButtonText)
```

该方法的参数与 `GetOpenFilename` 方法类似，可参见上例中的介绍。

3. 案例说明

在本例中，将调用系统内置的“保存”对话框，选择保存文件，并显示保存文件的路径。

4. 编写代码

使用“保存”对话框的 VBA 代码如下：

```
Sub UserSaveAS()  
    Dim sFilt As String
```

```
Dim sTitle As String
Dim sMsg As String
Dim sFname As Variant
Dim fileSaveName As String

sFilt = "Excel 文件(*.xls;*.xlsx;*.xlsm),*.xls;*.xlsx;*.xlsm," & _
    "所有文件(*.*),*.*"
sTitle = "保存文件"

fileSaveName = Application.GetSaveAsFilename(filefilter:=sFilt, _
    FilterIndex:=2, Title:=sTitle)

If fileSaveName <> "False" Then
    MsgBox "文件保存路径是: " & fileSaveName
End If
End Sub
```

5. 运行结果

运行程序代码，选择系统的文件类型，如图 8.4 所示。

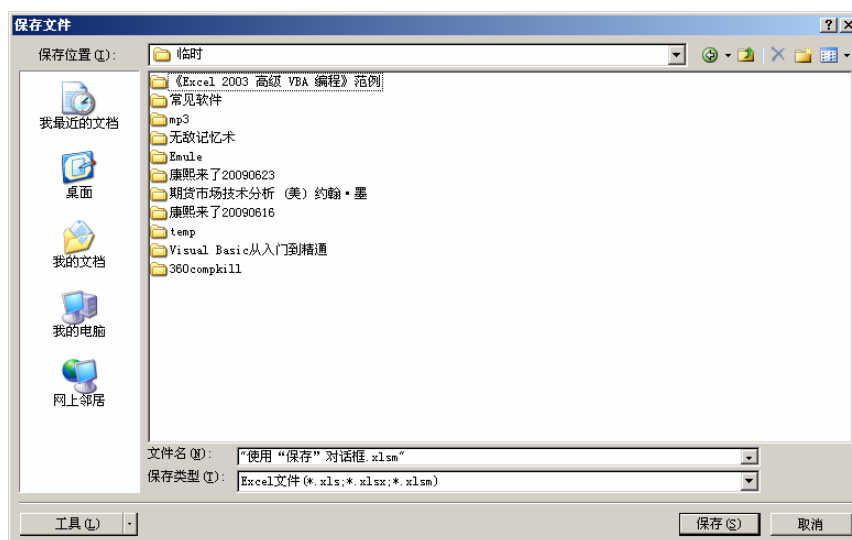


图 8.4 保存文件

6. 程序分析

本例的程序代码和前面小节类似，就不详细展开分析。

案例 155 调用内置对话框

1. 功能说明

前面案例介绍 Excel 的内置对话框，都是通过 Application 对象的方法形式来显示的。在 Excel 中还有更多的内置对话框，这些对话框组成了 Dialogs 集合。用户可以使用该集合调用内置对话框。

2. 语法说明

在 Excel 2007 中，Dialogs 集合包括了多个内置对话框。可使用以下代码查看内置对话框具体的数量：

```
MsgBox Application.Dialogs.Count
```

每个内置对话框由一个预定义的常量表示，要显示某个对话框，只需使用 Dialogs 集合对象的 Show 方法即可。

如使用以下代码，将显示 Excel 的“打开”对话框。

```
Application.Dialogs(xlDialogOpen).Show
```

3. 案例说明

本例的主要功能是在程序代码中调用内置的“打开”和“另存”对话框。

4. 编写代码

（1）调用“打开”对话框的 VBA 代码如下：

```
Sub UserOpenFiles()  
    Application.Dialogs(xlDialogOpen).Show  
End Sub
```

（2）调用“另存”对话框的 VBA 代码如下：

```
Sub UserSaveFiles()  
    Application.Dialogs(xlDialogSaveAs).Show "自定义文件.xlsm"  
End Sub
```

5. 运行结果

运行第一段程序代码，调用“打开”对话框，如图 8.5 所示。

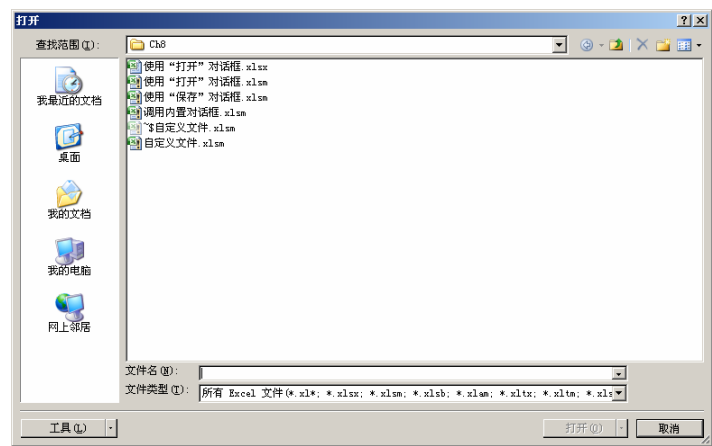


图 8.5 “打开”对话框

运行第二段代码，调用“另存为”对话框，如图 8.6 所示。

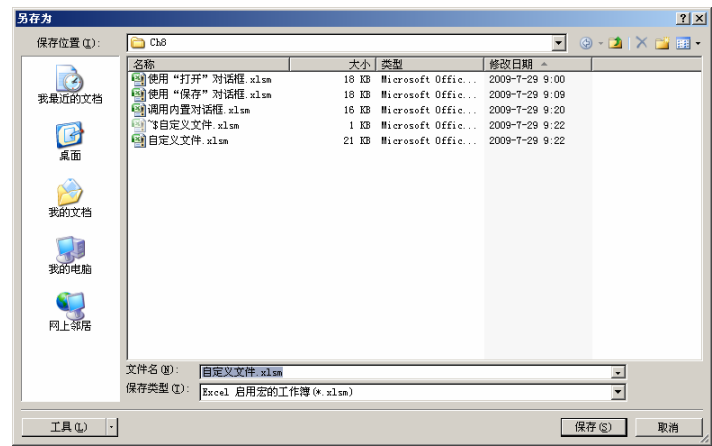


图 8.6 “另存为”对话框

6. 程序分析

用户可以将本例的代码和前面案例中的代码进行比较，查看两种打开对话框方法的不同。

案例 156 调用功能区功能

1. 功能说明

功能区是 Excel 2007 新增的对象，用户可以通过 VBA 代码调用功能区的功能。

2. 语法说明

在 Excel 2007 中，使用 CommandBars 集合的 ExecuteMso 方法可执行由 idMso 参数标识的控件功能。在 Excel 2007 中，内置按钮等控件都具有 idMso 参数，通过 ExecuteMso 方法可调用这些控件的功能，该方法的语法格式如下：

表达式.ExecuteMso(idMso)

参数 idMso 为控件的标识符。

3. 案例说明

本例的主要功能是在程序代码中调用内置的“打开”和“另存为”对话框。

4. 编写代码

(1) 调用“打开”对话框的 VBA 代码如下：

```
Sub 打开文件()  
    Application.CommandBars.ExecuteMso ("FileOpen")  
End Sub
```

(2) 调用“另存为”对话框的 VBA 代码如下：

```
Sub 另存文件()  
    Application.CommandBars.ExecuteMso ("FileSaveAs")  
End Sub
```

5. 运行结果

运行第一段程序代码，调用“打开”对话框，如图 8.7 所示。

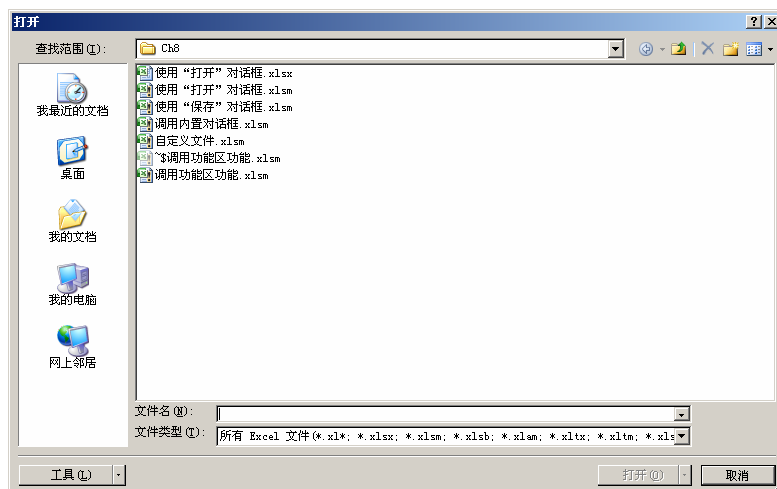


图 8.7 调用“打开”对话框

运行第二段代码，调用“另存为”对话框，如图 8.8 所示。

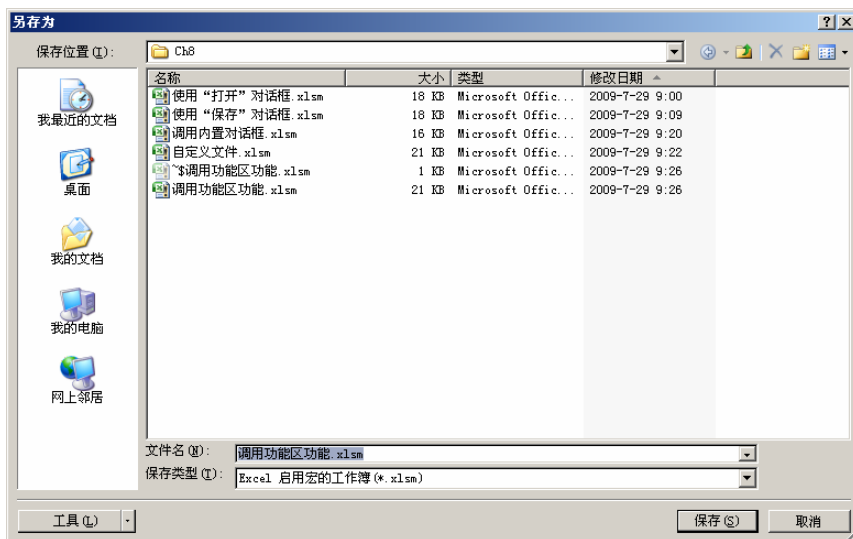


图 8.8 “另存为”对话框

6. 程序分析

用户可以在网址 <http://msdn.microsoft.com/office/tool/ribbon> 中下载文件 2007OfficeControlIDsExcel2007.EXE，该文件包括了 Office 2007 各组件的控件列表。将该文件解压后可以看到一个名为 ExcelRibbonControls.xlsx 的文件，其中包含了 Excel 2007 各控件的 idMso。

8.2 创建自定义窗体

在前面小节中，用户已经了解了如何调用和使用 Excel 的内置对话框，在本小节中，将详细讲解如何创建自定义窗体。在 Excel VBA 中，用户可以根据实际开发的需要，创建自定义窗体。

案例 157 插入窗体

1. 功能说明

插入窗体是用户创建自定义窗体的主要方法，在本小节中，将演示如何插入窗体。

2. 语法说明

本例中介绍插入窗体的方法，其插入方法和前面章节中介绍的插入模块和过程类似。主要是为了演示常见的插入方法。

3. 案例说明

在本例中，将主要演示如何在 VBE 中插入自定义的窗体。

4. 编写代码

在 Excel VBA 中，插入窗体的常见步骤如下：

- (1) 进入 VBE 环境，然后选择“插入”|“用户窗体”选项，如图 8.9 所示。

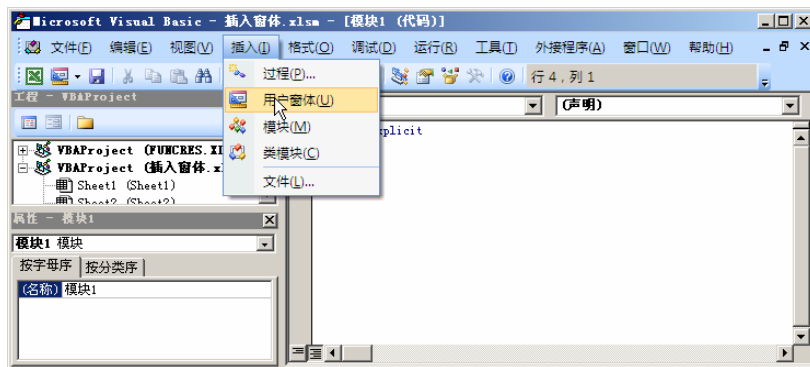


图 8.9 选择插入窗体

- (2) 当用户选择对应的选项后，系统会显示出默认的用户窗体，如图 8.10 所示。

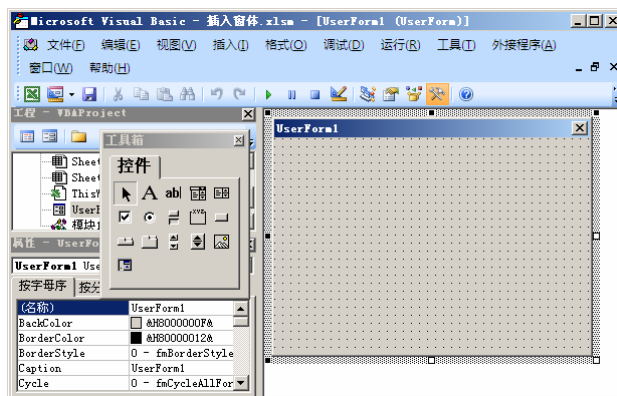


图 8.10 插入默认的用户窗体

- (3) 按 F5 键，调试程序代码，得到的空白窗体如图 8.11 所示。

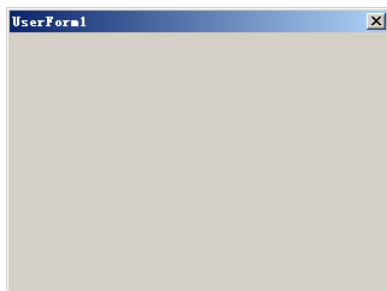


图 8.11 插入的空白窗体

5. 运行结果

本例的主要功能是向 Excel 中插入空的窗体，最后运行的结果如图 8.12 所示。

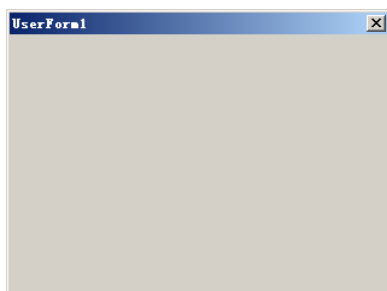


图 8.12 插入的窗体

6. 程序分析

从上面的结果中，用户可以看到关于窗体的默认属性。例如，系统插入的窗体自动命名为“UserForm1”，并同时显示控件工具箱以及属性工具箱。

案例 158 删除窗体

1. 功能说明

窗体是 Excel 的一种对象，当窗体中的数据无效时，用户需要删除窗体。

2. 语法说明

由于用户窗体中一般包含着多种数据信息，因此在用户删除窗体的时候，系统会提示是否导出该窗体，如果用户想彻底删除窗体，则选择不导出。

3. 案例说明

当窗体不再需要时，可将窗体从工程中移除。本例将主要介绍如何删除 Excel 文件中已有的窗体。

4. 编写代码

在 Excel VBA 中，删除窗体的常见步骤如下：

(1) 在工程资源管理器中，选中需要删除的窗体对象 UserForm1。选中的窗体对象上，单击鼠标右键，在弹出的快捷菜单中选择“移除 UserForm1”选项，如图 8.13 所示。

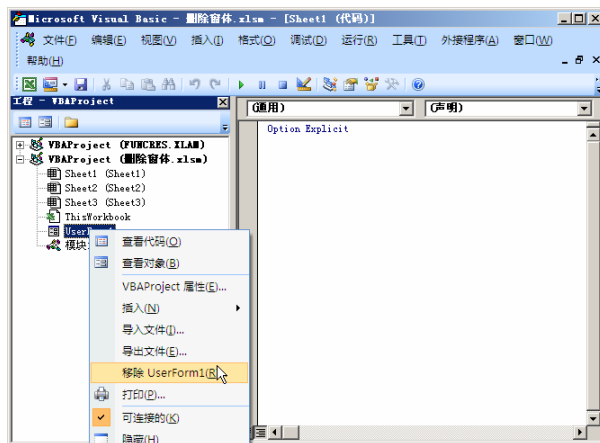


图 8.13 选择移除窗体

(2) 系统弹出如图 8.14 所示的对话框，单击“否”按钮，直接删除窗体。



图 8.14 移除窗口提示

5. 运行结果

本例的主要功能是将 Excel 的窗体删除，最后运行的结果如图 8.15 所示。

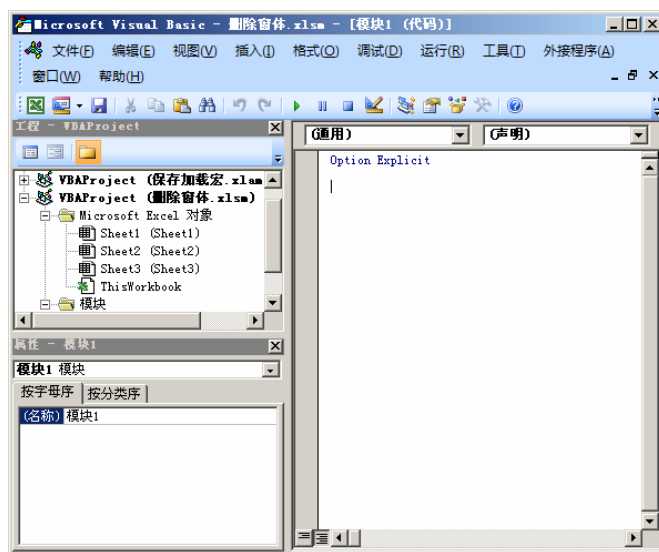


图 8.15 删除窗体

6. 程序分析

移除窗口提示中所使用的 UserForm1 是窗体的名称，选择移除不同的窗体对象时，其取值会有所不同。例如，用户这定了名称为“MyForm”，则显示的是“移除 MyForm”选项。

案例 159 导入窗体

1. 功能说明

和其他对象类似，用户可以在开发过程中，向 VBE 中导入类似功能的窗体。

2. 语法说明

当用户导出窗体进行保存后，在后续的开发中需要使用类似窗体时，就可以直接导出该窗体。

3. 案例说明

在实际开发过程中，用户可能需要创建各种类似的窗体。因此，为了节省创建窗体的时间，用户可以选择导入之前已经保存的窗体文件。本例将主要介绍如何导入窗体。

4. 编写代码

在 Excel VBA 中，导入窗体的常见步骤如下：

- (1) 进入 VBE 环境，然后选择“文件”|“导入文件”选项，如图 8.16 所示。
- (2) 在“导入文件”对话框中，选择导入的窗体文件，然后单击“打开”按钮，就

导入对应的窗体文件，如图 8.17 所示。

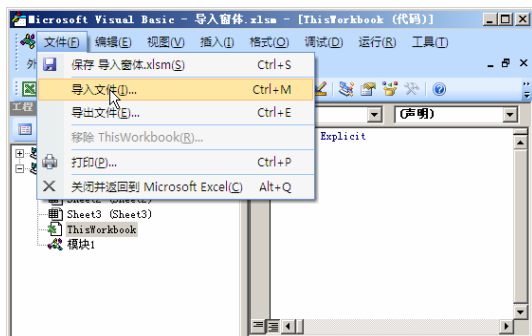


图 8.16 选择导入文件

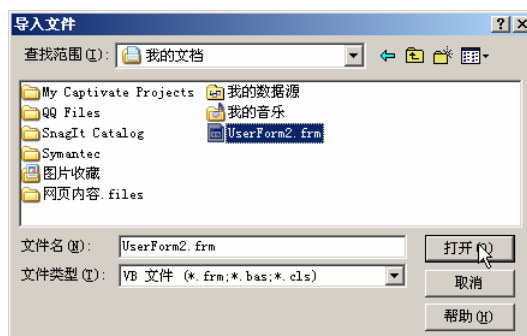


图 8.17 导入窗体文件

5. 运行结果

用户可以查看导入窗体的情况，如图 8.18 所示。

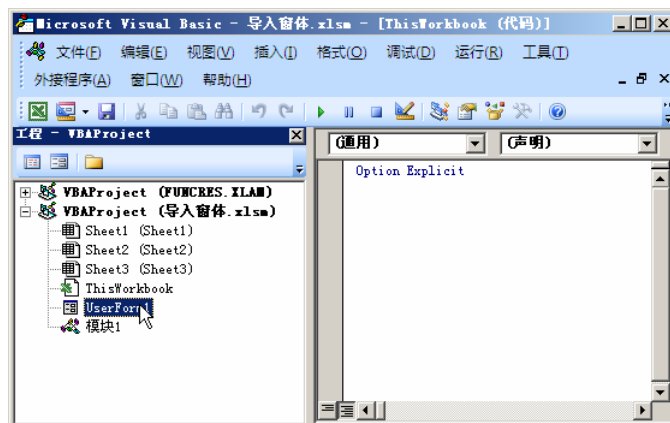


图 8.18 查看导入的窗体

6. 程序分析

从上面的例子中，用户可以看出，尽管导入的窗体文件名是“UserForm2”，但是由于原来的文件中不包含窗体，所以导入后，系统将窗体的名称设置为默认的“UserForm1”。

案例 160 显示窗体

1. 功能说明

用户可以使用 VBA 代码来控制窗体的显示属性。

2. 语法说明

在 Excel VBA 中，用户可以使用 Show 方法显示 UserForm 对象。其语法表达式如下：

```
[object.]Show modal
```

其中参数的含义如下：

- object: 代表对象表达式，其值为“应用于”列表中的对象。如果省略掉 object，则把与活动的 UserForm 模块相关联的 UserForm 当作 object。
- modal: 决定 UserForm 是模态的还是无模式的。

其中 modal 的设置值如下：

- vbModal: 数值是 1，表示 UserForm 是模态的。这是默认数值。
- vbModeless: 数值是 0，表示 UserForm 是无模式的。

3. 案例说明

在程序开发的过程中，有时需要控制窗体的显示。并不是希望窗体任何时候都出现在程序当前，例如，当某个事件触发的时候，才会显示窗体。本例将演示如何在程序中显示窗体。

4. 编写代码

本例的代码下：

```
Sub ShowForms()  
UserForm1.Show  
End Sub
```

5. 运行结果

当用户运行在 Sub 过程中的代码时，得到的程序结果如图 8.19 所示。

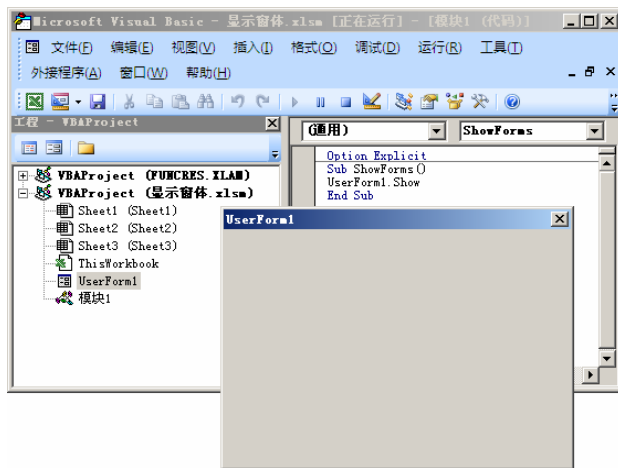


图 8.19 运行代码得到的结果

6. 程序分析

本例的代码十分简单，只是使用 Show 方法直接显示窗体。

案例 161 设置窗体背景图片

1. 功能说明

默认的窗体背景为灰色背景，为了增加窗体的美观性，可以为窗体添加背景图片。

2. 语法说明


通过使用窗体的 **Picture** 属性可以窗体添加背景图片，并可以设置对应的属性。

3. 案例说明

本例中将演示如何设置窗体的背景图片。

4. 编写代码

设置窗体背景图片的主要步骤如下。

(1) 在工程资源管理器中选择窗体，在属性窗口中，选中“**Picture**”属性，单击其后的按钮，如图 8.20 所示。

(2) 弹出“加载图片”窗口，在其中选取相应的图片，如图 8.21 所示。

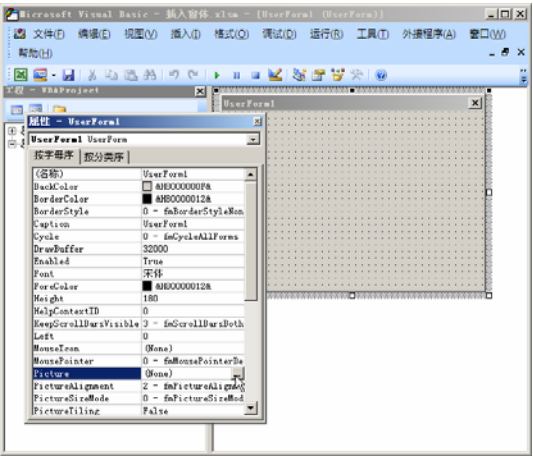


图 8.20 设置图片属性

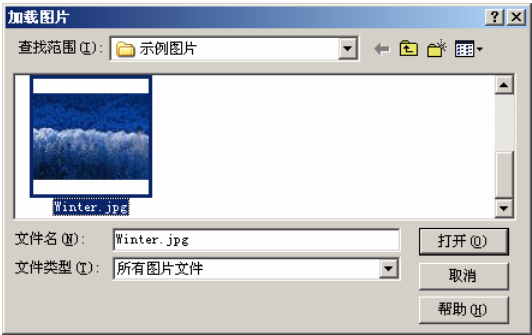


图 8.21 选择设置的图片

5. 运行结果

单击对话框中的“打开”按钮，设置窗体的背景图片，如图 8.22 所示。



图 8.22 设置窗体的背景图片

6. 程序分析

如果加载的图片大小超过了窗体的实际大小，系统会将超出部分截掉。为了使加载的图片全部显示出来，可对设置窗体的 `PictureAlignment` 属性和 `PictureSizeMode` 属性。

案例 162 加载窗体图片

1. 功能说明

在实际开发工程中，当用户希望某些事件触发后，窗体的背景图片修改，此时用户需要为窗体加载图片。

2. 语法说明

除了可以在设计窗体时为窗体添加图片外，还可以在 VBA 程序中为窗体添加图片。此时需要使用 `LoadPicture` 函数，其返回值为一个 `Picture` 对象，使用方法如下所述，其中的参数 `filename` 用于指明图片文件所在的路径和文件名。

`LoadPicture(filename)`

同时，为了设置加载图片的位置，需要使用 `PictureAlignment` 属性。该属性用来指定一个背景图片的位置。其语法表达式如下：

`object.PictureAlignment [= fmPictureAlignment]`

参数说明如下：

- `object`：表示有效对象。
- `fmPictureAlignment`：表示图片与控件对齐的位置。`fmPictureAlignment` 的具体取值如表 8.1 所示。

表 8.1 `fmPictureAlignment`的取值

常量	值	说明
<code>fmPictureAlignmentTopLeft</code>	0	左上角
<code>fmPictureAlignmentTopRight</code>	1	右上角
<code>fmPictureAlignmentCenter</code>	2	中心
<code>fmPictureAlignmentBottomLeft</code>	3	左下角
<code>fmPictureAlignmentBottomRight</code>	4	右下角

同时，为了设置窗体中图片的显示方式，还需要设置 `PictureSizeMode` 属性。该属性的主要功能是指定在控件、窗体或页面上显示背景图片的方式。其语法表达式如下：

`object.PictureSizeMode [= fmPictureSizeMode]`

参数 `fmPictureSizeMode` 表示当图片与包含它的窗体或页面大小不等时，应执行的操作。其具体取值如表 8.2 所示。

表 8.2 `fmPictureSizeMode`的取值

常量	值	说明
----	---	----

fmPictureSizeModeClip	0	裁掉图片中比窗体或页面大的部分（默认）。
fmPictureSizeModeStretch	1	扩展图片使其填满窗体或页面。该设置值使图片在垂直和水平方向都发生变形。
fmPictureSizeModeZoom	3	放大图片，但图片在水平和垂直方向上都不变形

3. 案例说明

本例中，首先显示的是一个空白窗体，如图 8.23 所示。然后在事件触发后，加载不同的图片。

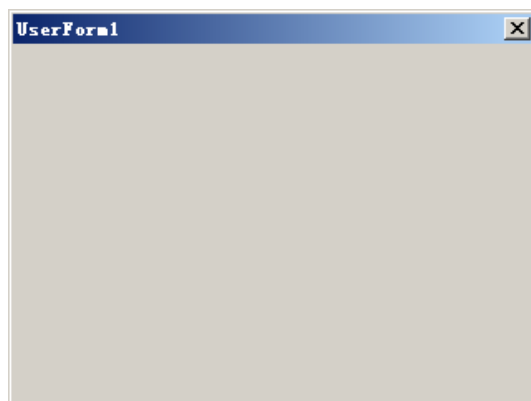


图 8.23 默认的窗体

4. 编写代码

本例的主要代码如下：

```
Private Sub UserForm_Click()  
With Me  
    .Picture = LoadPicture("D:\PerFace.jpg")  
    .PictureAlignment = fmPictureAlignmentCenter  
    .PictureSizeMode = fmPictureSizeModeZoom  
End With  
End Sub
```

5. 运行结果

当用户单击窗口后，系统自动加载图片，结果如图 8.24 所示。

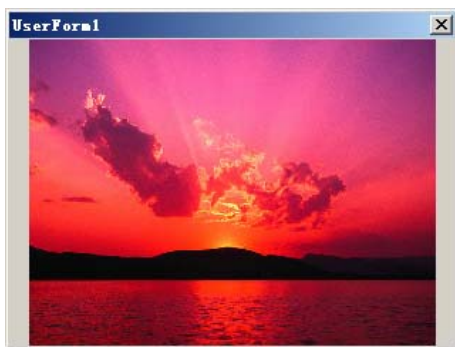


图 8.24 运行结果

6. 程序分析

LoadPicture 方法只能识别位图文件 (.bmp)、图标文件 (.ico)、行程编码文件、(.rle) 图元文件 (.wmf)、增强型图元文件 (.emf)、Gif 文件 (.gif)、JPG 文件 (.jpg)。

案例 163 创建 Splash 窗体

1. 功能说明

所谓 Splash 窗口，就是主界面出现之前先出现的欢迎窗口。本例将需要创建一个带有图片的 Splash 窗口。

2. 语法说明

在 Excel VBA 中，窗体在显示之前，必须装载到内存中。如果显示一个没有装载的窗体，该窗体将自动装载。如果想初始化窗体，而不显示窗体的话，可以使用如下方式装载。

Load frmSplash

窗体装载和卸载的顺序是：装载—显示———隐藏—卸载。卸载会清除窗体模块中的所有变量——类似于停止了一个过程。用户已经输入的任何数值都将丢失，控件将恢复为在属性窗口中输入的缺省值。如果想保存它们的值，需要在卸载窗体前进行保存。

在 Excel VBA 中，用户窗体可以在两种“模式”之间显示，即模式或者无模式。模式窗体，是指窗体显示时，用户只能在窗体中进行操作，不允许用户在 Excel 中进行其他的操作。而无模式窗体，则是在该窗体显示时，还允许用户在 Excel 中进行其他操作，再回到该窗体中来进行操作。

3. 案例说明

在本例中，创建一个带有图片的 Splash 窗体。

4. 编写代码

本例制作 Splash 窗体的步骤如下：

(1) 在 VBE 中插入一个用户窗体，并向窗体中增加一个图像控件，如图 8.25 所示。

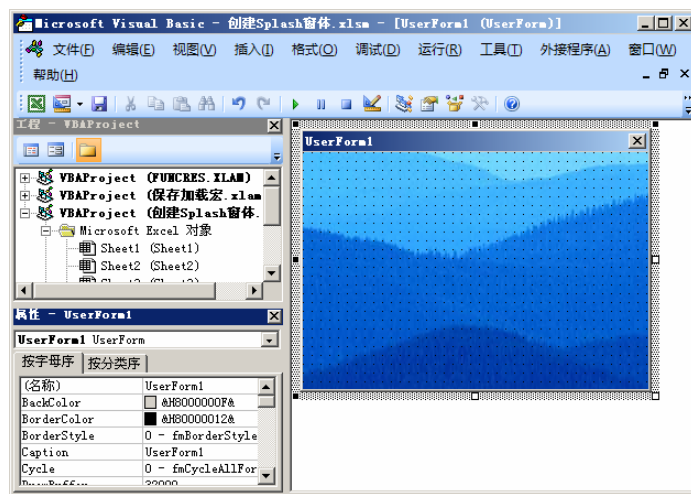


图 8.25 添加窗体对象

(2) 双击窗本打开代码窗口，在窗体的 `Activate` 事件中编写代码，设置调用 `CloseUserForm` 函数的时间间隔（该函数用来关闭 `Splash` 窗体）。具体代码如下：

```
Private Sub UserForm_Activate()  
    Application.OnTime Now + TimeValue("00:00:10"), "CloseUserForm"  
End Sub
```

(3) 在工程中插入一个模块，在模块中编写函数 `CloseUserForm`，用来卸载窗体，具体代码如下：

```
Sub CloseUserForm()  
    Unload UserForm1  
End Sub
```

(4) 为了使工作簿一打开就自动打开 `Splash` 窗体，在“工程”子窗体中双击“`ThisWorkbook`”打开代码窗口，给 `Workbook` 对象的 `Open` 事件编写代码如下：

```
Private Sub Workbook_Open()  
    UserForm1.Show  
End Sub
```

5. 运行结果

保存工作簿后，打开工作簿，查看添加的窗体情况，如图 8.26 所示。

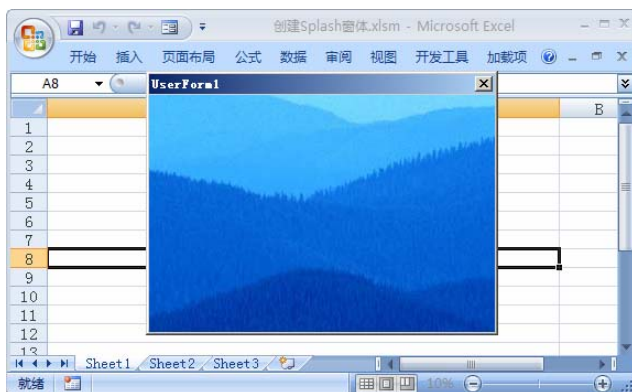


图 8.26 显示 Splash 窗体

6. 程序分析

用户可以修改 Splash 窗体的显示时间，查看程序运行的结果。

8.3 使用控件

在 Excel VBA 中，用户除了可以自行创建窗体之外，还可以根据程序的需求，为窗体添加各种不同功能的控件。在本小节中，将介绍如何在窗体中使用控件。

案例 164 添加工具箱中的控件

1. 功能说明

在 VBE 中，用户可以根据自己的需要向工具箱中添加控件。

2. 语法说明

本例中将主要演示如何通过控件工具箱添加新的控件选项。

3. 案例说明

在 Excel VBA 中，工具箱中列出的是常用控件，还有一些其他控件，没有在工具箱中显示出来。通过向工具箱中附加控件，可以将所需的控件添加到工具箱中。

4. 编写代码

添加工具箱中的控件步骤如下：

(1) 在工具箱的空白处，右击鼠标，弹出快捷菜单，选择“附加控件”菜单项，如图 8.27 所示。

(2) 在“附加控件”列表框中，单击相应的列表项，可选择多项，单击“确定”按

钮，附加控件完成，如图 8.28 所示。

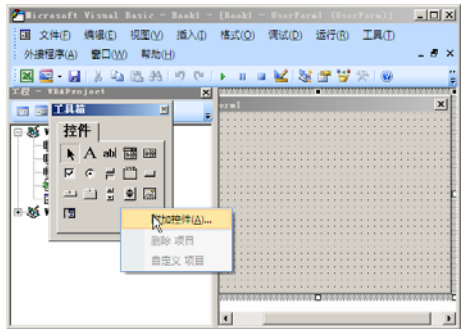


图 8.27 选择添加控件

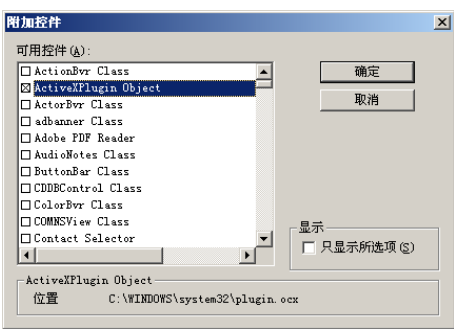


图 8.28 添加控件

5. 运行结果

在工具箱中添加的控件选项如图 8.29 所示。



图 8.29 添加的控件选项

6. 程序分析

在默认的情况下，新添加的控件没有任何提示信息，如果需要对控件进行说明，需要用户自行进行设置相关的信息。

案例 165 自定义控件提示信息

1. 功能说明

对于新添加的控件，其描述信息显示为未知。可以对其进行重命名或更改控件的显示图标。

2. 语法说明

本例中将主要演示如何自定义控件的提示信息。

3. 案例说明

本例中将主要演示自定义添加控件的提示信息。

4. 编写代码

自定义控件信息的步骤如下：

- (1) 在工具箱中，使用鼠标右键单击需要自定义的控件，弹出的快捷菜单，选择“自定义未知”菜单项，如图 8.30 所示。
- (2) 在弹出的“自定义控件”对话框中，输入新的文本“SelfDecide，如图 8.31 所示。

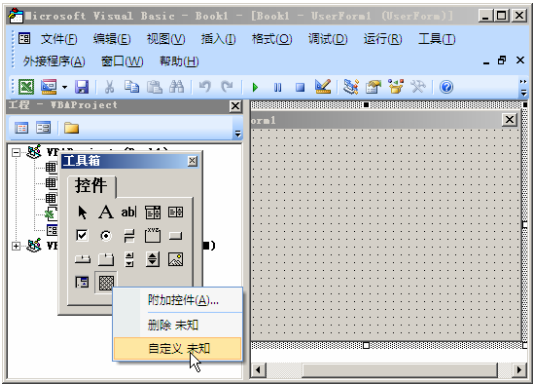


图 8.30 选择自定义控件



图 8.31 输入新的名称

- (2) 单击对话框中的“加载图片”按钮，打开“加载图片”对话框，在其中选择对应的图片，如图 8.32 所示。



图 8.32 自定义控件显示图片

5. 运行结果

一般情况下，控件的图片都是比较小的。如果用户自行加载的图片过大，系统会提示错误，如图 8.33 所示。



图 8.33 系统的提示信息

6. 程序分析

用户可以

案例 166 添加窗体控件

1. 功能说明

本例的主要功能是演示如何向窗体中添加控件。

2. 语法说明

本例所涉及的主要技术是使用鼠标在 VBE 窗体环境下添加窗体控件。

3. 案例说明

本例的主要功能是演示如何向窗体中添加控件。为了简单起见,本例中演示的添加“标签”控件,添加完成的标签控件。

4. 编写代码

添加窗体控件的步骤如下:

- (1) 在控件工具箱中选中“标签”控件,如图 8.34 所示。
- (2) 选中控件后,使用鼠标在窗体中划出相应的范围,如图 8.35 所示。

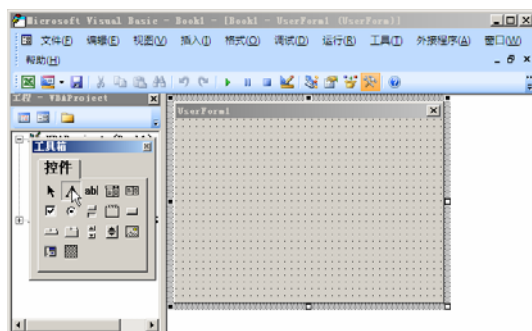


图 9.34 选择控件

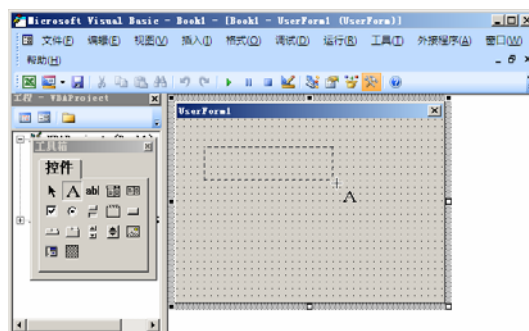


图 9.35 设置控件的位置

- (3) 使用鼠标划出适应的范围后,松开鼠标,得到的结果如图 8.36 所示。

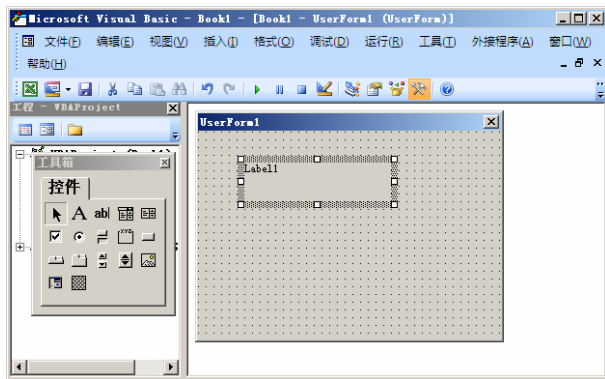


图 8.36 添加完成

5. 运行结果

运行程序代码，得到的结果如图 8.37 所示。

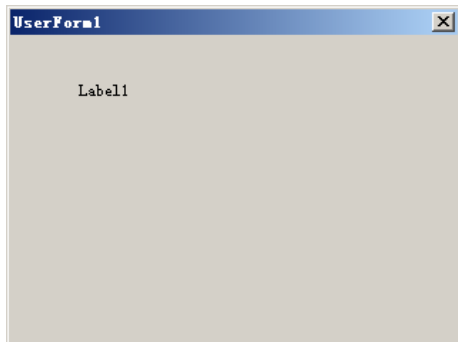


图 8.37 添加的窗体控件

6. 程序分析

当控件的工具箱没有显示时，可单击“视图”下拉菜，弹出下拉菜单，单击“工具箱”，弹出工具箱窗口。

案例 167 显示文本

1. 功能说明

在 Excel VBA 中，用户可以通过标签控件来显示不同的文本信息。灵活使用标签控件，用户可以开发出多种复杂的窗体。

2. 语法说明

在 Excel VBA 的控件中，标签（Label）是最简单的控件，其主要功能是显示告知性的信息，经常与其他控件结合使用，例如与文本框结合。左边使用标签，显示“姓名”，

后边给出一个文本框；当此界面显示时，用户可知在文本框，需要输入姓名。其默认属性是 **Caption** 属性，其常用属性如表 8.3 所示。

表 8.3 标签的常用属性

属性名称	意义
名称	用于标记在程序设计时所使用的标签名称，其作用同于标准变量的变量名。
AutoSize	是一个Boolean型的属性，用于设置标签的大小是否根据内容自动调整。
BackStyle	用于设置标签的背景样式，其值为0表示背景不透明，其值为1表示背景透明。
BackColor	用于设置标签的背景颜色。
BorderColor	用于设置标签的边框颜色。
BorderStyle	设置标签是否具有边框，默认值为0，表示没有边框，若设置为1表示窗体具有边框。
Caption	是一个字符串类型的属性，用于设置标签的标题，即在标签上显示的内容。
Enable	是一个布尔型的属性，其值决定了标签是否可用，如果其值为True，则标签为可用，其值为False，则标签不可用。
Font	返回一个字体对象，用于设置标签上内容的字体、字体大小、字形等。
ForeColor	用于设置或返回标签的前景色。
Height	用于以像素为单位设置标签的高度。
MouseIcon	用于设置鼠标停留在标签上时，鼠标显示的图片。
MousePointer	用于设置当鼠标停留在标签上时，鼠标指针的形状。
Picture	用于设置标签的背景图片
TextAlign	用于设置标签中所显示内容的对齐方式，其值为1、2、3，分别表示左对齐、居中、右对齐。
Visible	是一个Boolean型的属性，用于设置标签的在窗体上是否可见。
Width	用于描述标签的宽度。

3. 案例说明

本例的主要功能是，在窗体中显示不同的文字信息。

4. 编写代码

本例的主要代码如下：

```
Private Sub UserForm_Initialize()
    With Me
        .Caption = "显示文本"
        With .Label1
            .Caption = "Excel 2007"
            .AutoSize = True
        End With
        With .Label2
            .Caption = "中文版 Office 2007"
            .BackColor = RGB(255, 255, 0)
            .ForeColor = RGB(0, 255, 255)
        End With
    End With
End Sub
```

```
With .Font
    .Size = 16
    .Bold = True
    .Italic = True
    .Name = "隶书"
End With
.BorderStyle = fmBorderStyleSingle
.Height = 70
.Width = 150
End With
With Label3
    .Caption = "Excel VBA"
    .Height = 15
    .Width = 200
    .BackColor = RGB(0, 255, 0)
    .TextAlign = fmTextAlignCenter
End With
End With
End Sub+
```

5. 运行结果

运行程序代码，得到的结果如图 8.38 所示。



图 8.38 显示文本

6. 程序分析

在上面的事件中，用户使用到了窗体的事件。在 Excel VBA 中，添加窗体相关的事件，需要首先双击窗体，然后在打开的代码窗口中选择对应的事件名称，如图 8.39 所示。

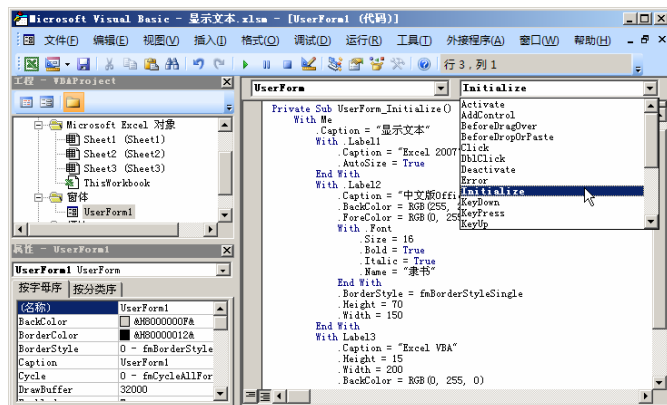


图 8.39 添加窗体代码

案例 168 选择窗体图片

1. 功能说明

当用户单击窗体中的控件时，会触发窗体的 Click 事件。用户可以在该事件中编写多种功能代码。

2. 语法说明

在 Excel VBA 中，Click 事件在下列两种情况下，会触发相应的程序代码：

- 用鼠标单击控件。
- 用户最终在几种可能的值中为控件选择一个值。

其语法表达式如下：

```
Private Sub object_Click( index As Long)
```

其中参数说明如下：

- **object**：必需参数。表示有效的对象。
- **index**：必需。与该事件相关联的多页或者标签的索引。

在导致 Click 事件发生的两种情况中，第一种情况应用于命令按钮、框架、图像、标签、滚动条和数值调节钮控件，而第二种情况用于复选框、组合框、列表框等控件。当选项按钮控件的值变为 True 时，也会导致 Click 事件发生。本例的另外一个重要的技术是 DblClick 事件。当用户指向对象并双击鼠标时，发生 DblClick 事件。其语法表达式如下：

```
Private Sub object_DblClick( ByVal Cancel As MSForms.ReturnBoolean)
```

若要使该事件发生，这两次击键必须发生在由系统的双击速度设置所限定的时间范围之内。

3. 案例说明

本例的主要功能是，在默认情况下，显示提示窗体。当用户在单击窗体时，窗体显示插图 1。当用户双击窗体时，窗体显示插图 2。

4. 编写代码

(1) “单击”窗体的事件代码如下：

```
Private Sub UserForm_Click()  
With Me  
.Caption = "插图 1"  
.Picture = LoadPicture("D:\PerFace.jpg")  
.PictureAlignment = fmPictureAlignmentCenter  
.PictureSizeMode = fmPictureSizeModeZoom  
If .label1.Visible = True Then  
.label1.Visible = False  
End If  
End With  
End Sub
```

(2) “双击”窗体的事件代码如下：

```
Private Sub UserForm_DblClick(ByVal Cancel As MSForms.ReturnBoolean)  
With Me  
.Caption = "插图 2"  
.Picture = LoadPicture("D:\PerFace2.jpg")  
.PictureAlignment = fmPictureAlignmentCenter  
.PictureSizeMode = fmPictureSizeModeZoom  
If .label1.Visible = True Then  
.label1.Visible = False  
End If  
End With  
End Sub
```

5. 运行结果

运行程序代码，提示窗体如图 8.40 所示，插图 1 的窗体如图 8.41 所示。



图 8.40 提示窗口

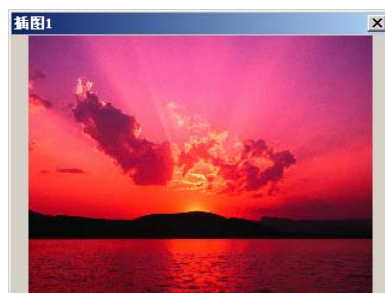


图 8.41 插图 1 窗体

当用户双击窗体，显示的插图 2 窗体如图 8.42 所示。

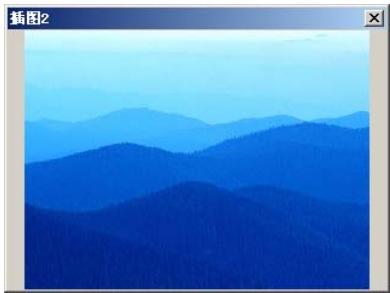


图 8.42 插图 2 窗体

6. 程序分析

在本例中，在提示窗口中，用户设置了窗体的标题。并在其中添加了标签控件，设置了标签中的文字，具体的设置方法，请用户参考对应的技巧，这里就不重复。

案例 169 禁用关闭按钮

1. 功能说明

在用户窗体中，默认情况下都有关闭按钮，当窗口处于运行状态时，单击窗口上的关闭按钮可退出窗口。有时为了程序的安全性，而有必要限制程序的退出方式，常常禁用窗口上的关闭按钮。

2. 语法说明

当用户单击窗体的“关闭”按钮时，触发的是 QueryClose 事件。该事件发生在 UserForm 关闭之前。其语法表达如下：

```
Private Sub UserForm_QueryClose(cancel As Integer, closemode As Integer)
```

其参数说明如下：

- cancel：将此参数设置成 0 以外的任意值，在所有加载的用户窗体中停止 QueryClose 事件，并防止关闭 UserForm 与应用程序。
- closemode：一个值或常数，用来指示引起 QueryClose 事件的原因。closemode 参数返回的数值如表 8.4 所示。

表 8.4 closemode的取值

常数	值	描述
vbFormControlMenu	0	用户在UserForm.上选择“控制”菜单中的“关闭”命令
VbFormCode	1	由代码调用Unload 语句。
vbAppWindows	2	正在结束当前当前 Windows 操作环境的过程。
vbAppTaskManager	3	Windows的“任务管理器”正在关闭这个应用

通常用这个事件确保在关闭应用程序之前，在应用程序包含的用户窗体中没有未完成

的任务。例如，如果用户尚未在任何一个 UserForm 中保存新数据，则应用程序可以提示用户保存。

3. 案例说明

在本例中，首先在窗体中添加标签控件，用户只能单击标签退出窗体。而不能使用窗体中的关闭按钮。

4. 编写代码

(1) 窗体的初始化代码如下：

```
Private Sub UserForm_Initialize()  
Me.Caption = "禁用窗口关闭按钮"  
End Sub
```

(2) 标签控件的单击事件代码如下：

```
Private Sub Label1_Click()  
Unload Me  
End Sub
```

(3) 单击关闭按钮之前的事件代码如下：

```
Private Sub UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)  
If CloseMode = 0 Then  
MsgBox "请单击窗体指定位置，关闭窗口!", vbOKOnly, "禁用关闭按钮"  
Cancel = True  
End If  
End Sub
```

5. 运行结果

其中，默认的窗体如图 8.43 所示。当用户单击窗口的关闭按钮后，系统显示的提示信息如图 8.44 所示。

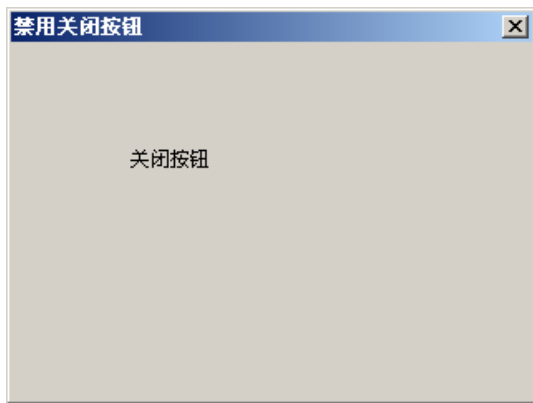


图 8.43 默认窗体

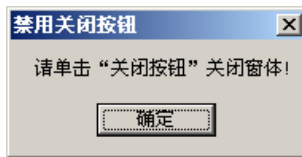


图 8.44 系统的提示信息

当用户单击窗口中的标签时，窗体关闭，如图 8.45 所示。



图 8.45 关闭窗体

6. 程序分析

在 Excel VBA 中，加载窗体需要使用 **Load** 语句，此语句仅是将窗体加载到内存，并不将窗体显示出来，其使用方法如下所示，窗体名是使用 **Dim** 声明窗体变量时所使用的窗体名，加载窗体后如果不卸载窗体，其将一直保存在内存中。

```
Load 窗体名
```