第3章 常用组件

Delphi 7.0 的组件板上含有 27 个选项卡，总共包括 350 多个组件，如图 3-1 所示。本章将对 Delphi 7.0 中常见的组件作详细的介绍。

图 3-1　组件板和选项卡

3.1 窗　　体

用户界面是应用系统中直接面对用户的窗体，包括主窗体、子窗体和弹出对话框窗体等，它是对空白窗体加工后的系统运行结果，在任何工程中，都是由窗体的不断开发和累加完成的集合。

3.1.1 Form 组件

一个应用系统至少有一个窗体，一个大型软件工程甚至包含几个、几十个到成百上千个窗体或程序模块，如图 3-2 所示。

Windows 操作系统中，人机交互的界面主要是通过一些窗口和对话框实现的。在 Delphi 7.0 中的可视化设计工作就是在窗体中进行的。通常，窗体中会有一些组件，通过这些组件可以实现各种各样的功能。在 Delphi 7.0 中，把这些运行期间出现在窗口和对话框中的组件称为可视组件。在窗体中，不仅可以放置组件，还可以放置一些运行期间不可视的组件，这些不可视的组件集中地实现了一些特殊的功能。

窗体是应用程序的操作界面，是放置组件的基础。在设计时，窗体就像一个桌面，可以放置各种组件；在运行时，窗体就像一个窗口，是程序与用户交流的地方。可以说，没有窗体，应用程序的框架就很难建立起来。窗体由标题栏、工作区和边界组成，通常标题栏中还有控制菜单、最大化/复原按钮、最小化按钮和关闭按钮等。用鼠标左键按下标题栏可以拖动窗体，用鼠标指向窗体的边界，如果出现双向的箭头，拖动鼠标可以改变窗体的大小。

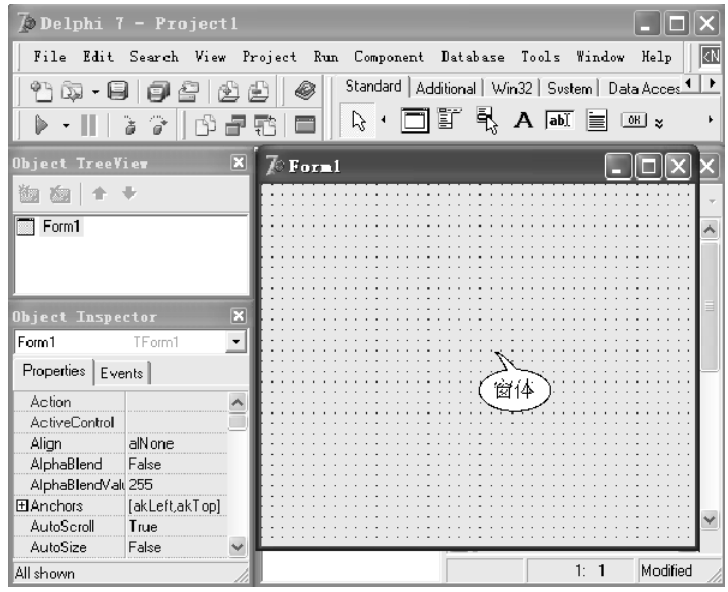

图 3-2　窗体

1. TForm 的主要属性

窗体组件(TForm)是一种特殊的组件,它在运行时表现为一个窗体,窗体是一个容器构件,它可以包含其他种类的构件,并协同完成应用程序的整体功能。窗体和其他组件一样由属性、事件和方法组成。

1) BorderIcons 属性

BorderIcons 属性用来制定窗体标题栏上的图标,可以设置为下列取值。

- biSystemMenu: 可以通过单击标题栏左边的图标或在标题栏上单击右键来显示控制菜单。控制菜单有时也称为系统菜单。
- biMinimize: 在标题栏右边显示最小化按钮。
- biMaximize: 在标题栏右边显示最大化按钮。
- biHelp: 在标题栏右边显示帮助按钮。只有窗体的 Borderstyle 属性设置为 bsDialog 或者窗体属性 BorderIcons 中不包括 biMinimize 和 biMaximizes 时,biHelp 设置才有效。

2) BorderStyle 属性

Borderstyle 属性用来设置窗体的外观和边框,可以制定为下面的取值。

- bsDialog: 窗体为标准的对话框,边框大小不可以改变。
- bsSingle: 窗体具有单线边框,大小不可以改变。
- bsNone: 窗体没有边框,也没有标题栏,边界的大小不可以改变。
- bsSizeable: 边框大小可变的窗体。
- bsToolwindow: 风格与 bsSingle 相同,只是标题栏比较小。另外,对于这种风格的窗体,属性 BorderIcons 中设置的 biMinimize 和 biMaximize 并不起作用。

- bsSizeToolWin：风格与 bsSizeable 相同，只是标题栏比较小。对于这种风格的窗体，属性 BorderIcons 中设置的 biMinimize 和 biMaximize 也不起作用。

3）Name 属性

Name 属性是对象的名称，它用来唯一地标识对象，Name 属性的取值不能为空，如果工程中有多个窗体，名称不能相同。

通常，应该在系统开发的设计阶段就将整个工程所有窗体的名称确定，然后在编程阶段根据设计文档修改窗体的 Name 属性。一般情况下，不要在程序运行期间通过代码修改 Name 属性。

4）Caption 属性

Caption 属性用来指定窗体标题栏中的说明文字，可以为空。默认时，Caption 属性与 Name 属性相同。

5）Font 属性

Font 属性用来设置窗体中文字的字体、颜色和字号等，其中 Font. style 属性为集合型。

6）FormStyle 属性

FormStyle 属性用来指定窗体的类型。

从窗体类型的角度来看，Windows 环境中的应用程序可以分为以下三类。

第一类：多文档界面(MDI)应用程序。这种应用程序一般具有一个父级窗口和多个子窗口，可以同时打开多个文档，分别在多个子窗口中显示。例如常用的文字处理软件 Word 等，可以同时编辑多个文档。

第二类：单文档界面(SDI)应用程序。这种应用程序只能同时打开一个文档。例如 Windows 操作系统附件中自带的“记事本”，只能同时编辑一个文本文件。

第三类：对话框应用程序。这种应用程序的主界面是基于一个对话框类型的窗体。例如 Windows 系统中自带的“扫雷”游戏程序。

FormStyle 属性的取值如下。

- fsNormal：普通类型的窗体，既不为 MDI 应用程序的父级窗口，也不为 MDI 应用程序的子窗口。
- fsMDIChildMDI：应用程序中的子窗体。
- fsMDIFormMDI：应用程序中的父窗体。
- FsStayOnTop：在桌面最前端显示窗体。

7）Icon 属性

Icon 属性用来指定标题栏中显示的图标。

单击对象编辑器 Icon 属性右边的“省略号”按钮 ···，在打开的 PictureEditor 对话框中单击 Load 按钮，就可以装入一个制作好的图标。

8）Position 属性

Position 属性用来描述窗体的大小和显示的位置，其取值如下。

- poDesigned：窗体显示的位置和大小与设计期间的一致。
- poDefault：窗体每次显示时，与上次比较，往右下角移动了一些位置。窗体的高度和宽度由 Windows 决定。
- poDefaultPosOnly：窗体以设计期间的大小显示，窗体显示的位置较上次向右下角

移动了一些。如果窗体以设计时的大小不能在屏幕上完全显示，就移动到屏幕的左上角显示。

- poDefaultSizeOnly：窗体以设计期间的位置显示，窗体的大小由 Windows 系统决定。
- poScreenCenter：窗体以设计期间的大小显示，窗体显示的位置总在屏幕的中间。考虑多个监视器时位置的调整。
- poDesktopCenter：窗体以设计期间的大小显示，窗体显示的位置总在屏幕的中间。不考虑多个监视器时位置的调整。

9）WindowsState 属性

WindowsState 属性用来描述窗体显示的状态，其取值如下。

- wsNormal：窗体以普通状态显示(既不是最大化状态，也不是最小化状态)。
- wsMinimized：窗体以最小化状态显示。
- WsMaximized：窗体以最大化状态显示。

2. TForm 的事件

窗体是一个可视化的组件，它几乎可以响应和处理所有的事件，包括外部事件和内部事件。窗体常用的响应事件如表 3-1 所示。

表 3-1　窗体常用的事件响应

事　件	含　义
OnActive	当窗体对象被激活时产生此事件
OnClose	当窗体对象被关闭时产生此事件
OnCloseQuery	当窗体对象被关闭时或者调用系统菜单的关闭菜单项时产生此事件，其中包含 canclse 参数，用于决定是否关闭窗体
OnCreate	当窗体对象创建时产生此事件
OnDeactivate	当窗体对象变为非激活时产生此事件
OnDestroy	当窗体对象被销毁前产生此事件
OnHide	当窗体对象被隐藏前产生此事件
OnPaint	当窗体对象需要被重画时产生此事件
OnResize	当窗体对象位置移动时产生此事件
OnShow	当窗体对象显示时产生此事件
OnKeyDown	键盘按下任意键(含组合键)时产生此事件
OnKeyPress	键盘按下字符键时产生此事件
OnKeyUp	键盘放开时产生此事件
OnClick	鼠标单击事件
OnDblClick	鼠标双击事件
OnDragDrop	鼠标拖放事件
OnDragOver	鼠标拖过事件
OnMouseDown	鼠标按下事件
OnMouseMove	鼠标移过事件
OnMouseUp	鼠标释放事件

3. 窗体的方法

窗体对象从其父类 TcustomForm 中继承了许多方法，表 3-2 列出了其中一些常用的方法(过程或函数)。

表 3-2 窗体常用的方法

方法名称	含义
Create	用来创建一个窗体并进行初始化，同时引发一个 OnCreate 事件。用该方法创建的窗体需要调用 Show 方法使之可见
Close	用来关闭一个显示中的窗体，同时调用 CloseQuery 方法来判断是否可以关闭，若可以，则引发一个 OnClose 事件并关闭窗体
CloseQuery	用来判断窗体是否可以被关闭，返回一个逻辑值
Release	用于将窗体对象从内存中彻底清除
Show	用于显示窗体，同时引发一个 OnShow 事件
ShowModal	用于显示一个模式窗体，同时引发一个 OnShow 事件
Print	用于打印窗体

4. 窗体的创建

创建一个新的应用程序(Application 工程)，Delphi 将自动建立一个新的窗体，它代表应用程序的主窗口。应用程序也可以拥有多个窗体作为子窗口、对话框和数据录入窗口等，这就需要创建和显示新的窗体。创建窗体的方法分为静态创建和动态创建两种。所谓静态创建窗体是指在工程的编辑、设计时创建新窗体；而动态创建窗体是指在工程的运行时通过代码生成窗体。

1) 静态创建新窗体

【例 3.1】 静态创建窗体。

具体操作步骤如下。

① 界面设计。

通过执行集成开发环境中的 File|New|Application 命令，创建一个应用程序，此时应用程序自动生成一个窗体 Form1，再执行 File|New|Form 命令生成一个窗体 Form2。在 Form1 中添加两个 Button 组件、一个 Label 组件；Form2 中添加一个 Label 组件，即可完成界面设计，各组件的属性如图 3-3 所示。

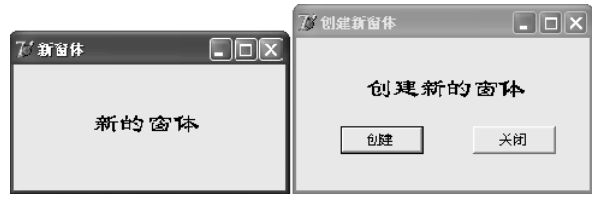

图 3-3 静态窗体的创建

② 程序设计。

代码如下：

```
procedure TForm1.Button1Click(Sender: TObject);          //创建按钮事件
begin                                                    //关键分析
form2.show;                                              //调用 Show 方法显示 Form2 窗体
end;
procedure TForm1.Button2Click(Sender: TObject);
begin
  form1.Close;
end;
```

③ 程序分析。

关键分析：编译上述工程时，系统会弹出如图 3-4 所示的提示信息。

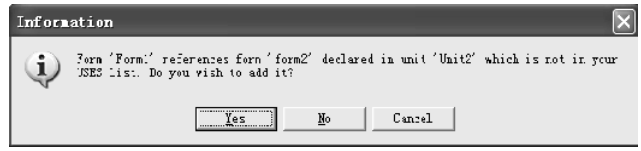

图 3-4　提示信息

因为在调用 form2.show 显示 form2 窗体时需要引用 Unit2 单元，但是 Unit1 中又没有声明要使用 Unit2 单元，所以弹出此提示信息问是否在 Unit1 单元中添加 Unit2 单元。单击 Yes 按钮，Delphi 将自动在 Unit1 单元中添加对 Unit2 单元的引用。

也可以手动添加，单击 Cancel 按钮，在 Unit1 中的实现部分(implementation)添加如下代码：

```
implementation
uses unit2
```

2）动态创建新窗体

用静态方法创建的多窗体的工程，在运行时将把所有的窗体装入内存。如果工程非常复杂，系统资源将变得非常紧张。这种情况使用下面的方法：在需要某个窗体的时候，临时创建它，使用后将其立即释放。这种方法称为窗体的动态创建。

【例 3.2】　动态创建窗体。

具体操作步骤如下。

① 界面设计。

通过执行集成开发环境中的 File|New|Application 命令，创建一个应用程序，此时应用程序自动生成一个窗体 Form1，再执行 File|New|Form 命令生成一个窗体 Form2。在 Form1 中添加两个 Button 组件、一个 Label 组件，Form2 中添加一个 Label 组件，即可完成界面设计，各组件的属性如图 3-3 所示。

② 属性设置。

执行 Project|Options 命令，打开 Project Options for Project1.exe 对话框。在 Forms 选项卡中，所有工程中的窗体都出现在 Auto-create forms 列表框中，表示这些窗体在运行时都是在内存中自动创建的。选中 Form2，将其移至 Available forms 列表框中，如图 3-5 所示。

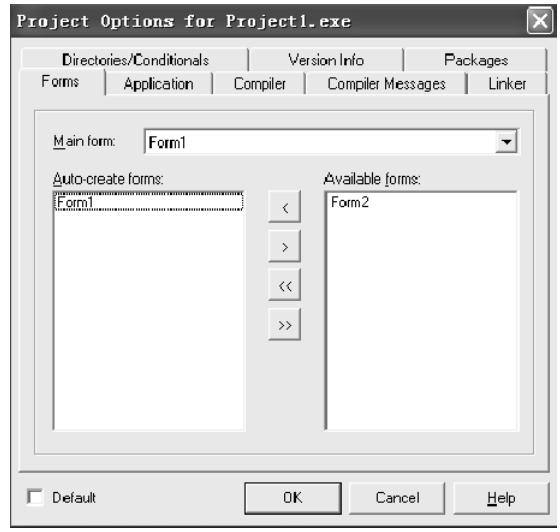

图 3-5　Project Options for Project1.exe 对话框

③ 程序设计。

代码如下：

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  form2 := Tform2.Create(nil);                          //关键分析 1
  form2.Show;
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
  form1.Close;
end;

procedure TForm2.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  form2.Release;                                        //关键分析 2
end;
```

④ 程序分析。

关键分析 1：使用 Create 方法创建并完成 Form2 的初始化。

关键分析 2：释放 Form2 所占有的内存。

3.1.2　弹出对话框窗体

对话框是用户与应用程序交换信息的最佳途径之一。使用对话框函数或过程可以调用 Delphi 的内部对话框，这种方法具有操作简单及快速的特点。Delphi 提供的内部对话框有

如下两种。

第一种：信息输出对话框。包括 ShowMessage 过程、ShowMessageFmt 过程、MessageDlg 函数、MessageDlgPos 函数和 CreateMessageDialog 函数。

第二种：信息输入对话框。包括 InputBox 函数和 InputQuery 函数。

1. ShowMessage 过程

ShowMessage 过程显示一个最简单的对话框，其语法格式为：

```
ShowMessage(＜信息内容＞);
```

说明：

(1) ShowMessage 过程显示的对话框以应用程序的执行文件名作为标题，对话框中只含有一个 OK 按钮，单击该按钮对话框即关闭并返回。

(2) ＜信息内容＞指定在对话框中出现的文本。在＜信息内容＞中使用硬回车符(#13)可以使文本换行。对话框的高度和宽度随着＜信息内容＞的增加而增大。

2. ShowMessageFmt 过程

语法格式为：

```
ShowMessageFmt(＜信息内容＞,＜参数组＞);
```

说明：ShowMessageFmt 过程与 ShowMessage 过程的功能基本相同，只是参数＜信息内容＞为格式化了的字符串，与＜参数组＞配合，形成显示在对话框中的信息。例如下述代码将得到如图 3-6 所示的对话框。

图 3-6　信息对话框

```
ShowMessageFmt('张宇同学 %s 考了 %d 分！',['外语',100]);
```

3. MessageDlg 函数

调用 MessageDlg 函数，可以在屏幕的中心处显示信息对话框，其语法格式为：

```
＜变量＞ = MessageDlg(＜信息内容＞,＜类型＞,＜按钮组＞,HelpCtx);
```

说明：

(1) ＜信息内容＞是显示在对话框中的信息。

(2) ＜类型＞是对话框的类型，其取值如表 3-3 所示。

表 3-3　对话框类型的取值

取　值	说　明
mtWarning	弹出含有感叹号符号的警告对话框
mtError	弹出含有红色叉符号的错误对话框
mtInformation	弹出含有蓝色 i 符号的信息对话框
mtConfirmation	弹出含有绿色？号的确认对话框
mtCustom	弹出不含图标的一般对话框，对话框的标题是程序的名称

(3) ＜按钮组＞指定对话框中出现的按钮,其中出现的按钮与参数的取值如表 3-4 所示。

表 3-4 对话框按钮类型的取值

取　值	说　明
mbYes	单击 Yes 按钮,函数返回 mrYes 或 6
mbNo	单击 No 按钮,函数返回 mrNo 或 7
mbOK	单击 OK 按钮,函数返回 mrOK 或 1
mbCancel	单击 Cancel 按钮,函数返回 mrCancel 或 2
mbHelp	help 按钮
mbAbort	单击 Abort 按钮,函数返回 mrAbort 或 3
mbRetry	单击 Retry 按钮,函数返回 mrRetry 或 4
mbIgnore	单击 Ignore 按钮,函数返回 mrlgnore 或 5
mbAll	单击 all 按钮,函数返回 mrall 或 8
mbNoToAll	单击 Nottoall 按钮,函数返回 9
mbYesToAll	单击 Yestoall 按钮,函数返回 10

＜按钮组＞可以是组的形式,如[mbYes,mbNo]表示对话框中出现 Yes 和 No 两个按钮;也可以是常量的形式,如 mbokcancel 表示对话框中出现 OK 和 Cancel 两个按钮。按钮常量的含义如表 3-5 所示。

表 3-5 按钮常量的取值

取　值	说　明
mbYesNoCancel	3 个按钮:Yes、No、Cancel
mbOKCancel	两个按钮:OK、Cancel
mbAortRetryIgnore	3 个按钮:Abort、Retry、Ignore

(4) HelpCtx 指定当用户单击 Help 按钮或按 F1 键时,显示的帮助主题。

(5) MessageDlg 函数将根据用户所选择的按钮,返回相应的值(Word)类型,参见说明(3)。

4. MessageDlgPos 函数

调用 MessageDlgPos 函数,可以在屏幕的指定位置显示信息对话框,其语法格式如下:

```
＜变量＞ = MessageDlgPos(＜信息内容＞,＜类型＞,＜按钮组＞,HelpCtx,X,Y);
```

说明:MessageDlgPos 函数只比 MessageDlg 函数多一项功能,即它可以指定对话框的显示位置坐标 X,Y。

5. CreatMessageDialog 函数

与上述函数和过程不同,CreatMessageDialog 函数生成一个信息框窗体,可以在程序中多次使用方法调用。其语法格式如下:

```
＜变量＞ = CreatMessageDialog (＜信息内容＞,＜类型＞,＜按钮组＞);
```

说明：函数的参数与 MessageDlg 函数相似，只是返回一个 TForm 类型的对话框，而且并没有把它显示出来。在需要显示该对话框的时候，可以使用窗体 ShowModal 的方法把它弹出。

6. InputBox 函数

InputBox 函数显示一个能接受用户输入的对话框，并返回用户输入的信息。其语法格式如下：

```
<变量> = InputBox (<对话框标题>,<信息内容>,<默认内容>);
```

说明：

(1) <对话框标题>指定对话框的标题。

(2) <信息内容>指定在对话框中出现的文本。在<信息内容>中使用硬回车符可以使文本换行。对话框的高度和宽度随着<信息内容>的增加而增大。

(3) <默认内容>指定对话框的输入框中显示的文本，可以修改。如果用户单击“确定”按钮，输入框中的文本将返回到<变量>中；若用户单击“取消”按钮，则将<默认内容>返回到<变量>中。

7. InputQuery 函数

如果希望对单击 Cancel 按钮(退出事件)另作处理，可以使用 InputQuery 函数。该函数与 InputBox 函数相似，只是返回值是一个布尔值。其语法格式如下：

```
<变量> = InputQuery (<对话框标题>,<信息内容>,<字符串变量>);
```

说明：InputQuery 函数与 InputBox 函数的参数相似，默认内容存放在<字符串变量>中，可以修改。如果用户单击 OK 按钮，输入框中的文本将赋值到<字符串变量>中，并且函数返回 True；若用户单击 Cancel 按钮，<字符串变量>中的值则保持不变，并返回 False。

3.2　输入显示类组件

3.2.1　Edit 组件

编辑框(Edit)是一种通用组件，既可以输入文本，又可以显示文本，是应用程序中最常用的组件之一。编辑框组件位于 Standard 组件板中，如图 3-7 所示。

图 3-7　编辑框

以下介绍 Edit 的主要属性。

(1) AutoSelect 属性：用于设置编辑框得到焦点时，文本是否自动被选中。

(2) AutoSize 属性：决定编辑框是否自动随字体的变化而改变大小。

(3) Enable 属性：用来设置编辑框是否能用。

(4) BorderSytle 属性：用来设置编辑框的边框类型，取 bsSigle 为单线，取 bsNone 为无框。

(5) MaxLength 属性：用于设置所能接受的最大字符数。

(6) PasswordChar 属性：该属性设置非 #0 字符时，将代替用户输入的字符被显示。

(7) ReadOnly 属性：决定编辑框中的文本是否可以编辑。

(8) SelStart 属性：用于设置被选中文本的开始位置，或光标在文本中的位置。

(9) SelText 属性：用于设置被选中的文本。

(10) SelLength 属性：用于设置被选中文本的长度。

(11) Text 属性：用于设置编辑框中的文本内容。

(12) CharCase 属性：控制编辑框中文本的大小写，取值为 ecLowerCase，Text 中的文本将转换为小写；取值为 ecNormal，Text 中的文本不作改变；取值为 ecUpperCase，Text 中的文本将转换为大写。

3.2.2 Label 组件

标签(Label)是 Delphi 中最常用的输出文本信息的工具。标签组件位于 Standard 组件板中，如图 3-8 所示。

图 3-8 标签

1. Label 的主要属性

(1) Caption 属性：用来显示标签的文本。

(2) ShowAccelChar 属性：决定是否将“&”作为热键字符的标记。

(3) AutoSize 属性：决定标签是否自动随文本的变化而改变大小。

(4) Alignment 属性：决定对齐方式，左对齐、居中对齐或右对齐。

(5) Layout 属性：控制文本显示在标签的顶部、中部或底部。

(6) WordWrap 属性：控制是否折行显示。

(7) Transparent 属性：决定背景是否透明。

(8) FocusControl 属性：用来设置按下热键时，获得焦点的组件名。

2. Label 的使用

【例 3.3】 利用标签设计一个投影效果。

具体操作步骤如下。

① 界面设计。