Delphi 5
高级编程丛书之二

Delphi 5 高级编程

——GUI编程

徐新华 编著

人民邮电出版社
www.pptph.com.cn

Delphi 5 高级编程丛书之二

Delphi 5 高级编程
——GUI 编程

徐新华　编著

人 民 邮 电 出 版 社

内 容 提 要

本书全面深入地介绍了如何运用 Delphi 5 进行 GUI 编程，书中的内容非常丰富，涉及以下几个方面：设计应用程序的图形界面；使用 Windows 公共对话框和 Win32 公共控件；通过剪贴板、DDE 和 OLE 来共享信息；多媒体编程，介绍了一个简单的媒体播放器和一个 CD 播放器；Win32 编程，包括进程、内核对象、GDI 和 USER 对象、多任务和多线程、内存管理、错误处理；Win32 和 Delphi 的消息机制；文件、目录和驱动器；各种类型的应用程序，包括 MDI 程序、控制台程序、服务程序、控制面板小程序；屏幕和打印机；图像，并介绍了一个绘图程序；多线程技术。

本书内容全面，叙述简洁，例子丰富。Delphi 的新手可以把它作为入门指导书，资深的 Delphi 程序员可以从中学到许多实用的编程技巧。

Delphi 5 高级编程丛书之二

Delphi 5 高级编程

——GUI 编程

◆ 编　　著　徐新华

责任编辑　李　际

◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街 14 号

北京顺义向阳胶印厂印刷

新华书店总店北京发行所经销

◆ 开本：787 × 1092　1/16

印张：32

字数：797 千字　　2000 年 4 月第 1 版

印数：1 - 6 000 册　　2000 年 4 月北京第 1 次印刷

ISBN 7-115-08461-0/TP·1580

定价：50.00 元

前　言

Delphi 5 是 Inprise 公司面向 Internet 和电子商务推出的一个战略产品。在 Internet 日益普及和众多厂商看好电子商务的今天，Delphi 5 的推出必将受到程序员的热烈欢迎。

为了帮助广大用户全面、准确地掌握 Delphi 5 的编程思想和用法，我们编写了这套《Delphi 5 高级编程丛书》。作为《Delphi 4 高级编程丛书》的升级，本套丛书在保留原丛书精髓的基础上，对丛书的结构和内容作了大刀阔斧的修改，这是考虑到很多程序员已经初步掌握了 Delphi 5 的基本用法，现在需要的是进一步提高和精通。

本套丛书分为 4 册。第一册是《Delphi 5 高级编程——IDE 和面向对象编程》，第二册是《Delphi 5 高级编程——GUI 编程》，第三册是《Delphi 5 高级编程——Database 与 MIDAS 编程》，第四册是《Delphi 5 高级编程——COM、CORBA 与 Internet 编程》。

本书是此套丛书的第二册，书中全面深入地介绍了 Delphi 5 的 GUI 编程技术，包括设计图形界面；公共对话框；Win32 公共控件；用剪贴板、DDE、OLE 共享信息；多媒体编程；Win32 编程和消息；文件、目录和驱动器；Form 和应用程序；屏幕和打印机；图像；多线程等内容。

由于我们的水平有限，再加上时间很紧，尽管我们作了严格的审核和测试，书中还是难免会有一些错误，敬请广大读者不吝赐教，我们谨在此表示感谢。

为了帮助广大程序员更好地掌握这个优秀的开发工具，北京东大阿尔发软件技术有限公司愿意为购买此书的读者提供技术咨询。

北京东大阿尔发软件技术有限公司
地址：北京市上地信息产业基地上地村路 1 号(100085)
电话：(010)62987260　传真：(010)62985141
网址：http://www.allfa.com.cn　邮件：books@allfa.com.cn

徐新华
2000 年 2 月

目 录

第一章 设计图形界面

本章有以下内容：

- 菜单、快捷菜单；
- 标签；
- 编辑框；
- 多行文本编辑器；
- 命令按钮；
- 复选框；
- 单选框；
- 列表框、组合框；
- 滚动条；
- 分组框、单选分组框；
- 窗格；
- 动作列表；
- 框架；
- 位图按钮、快捷按钮；
- 按格式输入编辑框；
- 自绘栅格、字符串栅格；
- 图像；
- 几何图形；
- 分界；
- 滚动箱；
- 带复选框的列表框；
- 尺寸调节杆；
- 静态文本；
- TControlBar；
- 处理 TApplication 对象的事件；
- 定时器；
- 画板；
- 文件列表框；
- 目录列表框；
- 驱动器组合框；
- 文件类型过滤器。

Delphi 5 是开发 Windows 应用程序的最佳工具。使用 Delphi 5 可以轻松地制作出程序的图形界，包括主菜单、快捷菜单、标签、编辑框、命令按钮、复选框、单选框、列表框、组合框、滚杠、分组框、窗格、位图按钮、快捷按钮、栅格、几何图形、分界、滚动箱、尺寸调节杆、静态文本、控制条等。过去许多程序员为了能使程序具有漂亮的界面而伤透脑筋；现在有了 Delphi 5，设计应用程序将成为乐趣。

元件选项板“Standard”页和“Additional”页上的元件可以用来建立 Windows 应用程序的图形界面。此外，元件选项板的“System”页和“Win 3.1”页上的元件实现了一些特殊的系统接口，如定时器、画板、文件列表框、目录列表框、驱动器组合框、文件类型过滤器等。下面就按照它们在元件选项板上的顺序详细介绍这些元件的用法。

1.1 菜　　单

要使应用程序有菜单，必须把 TMainMenu 或 TPopupMenu 元件放到 Form 上。这两个元件的区别是，前者用于建立程序的主菜单，后者用于建立弹出式菜单。

通过 Delphi 5 提供的菜单设计器，可以轻松地设计出复杂的菜单结构。

1.1.1 菜单设计器

要打开菜单设计器有 3 种方式：

一、直接在 Form 上双击 TMainMenu 元件或 TPopupMenu 元件。

二、在 Object Inspector 上单击 Items 特性的省略号按钮。

三、用鼠标右键单击 TMainMenu 元件或 TPopupMenu 元件，在弹出的菜单中选择“Menu Designer”命令。

打开的菜单设计器如图 1.1 所示。

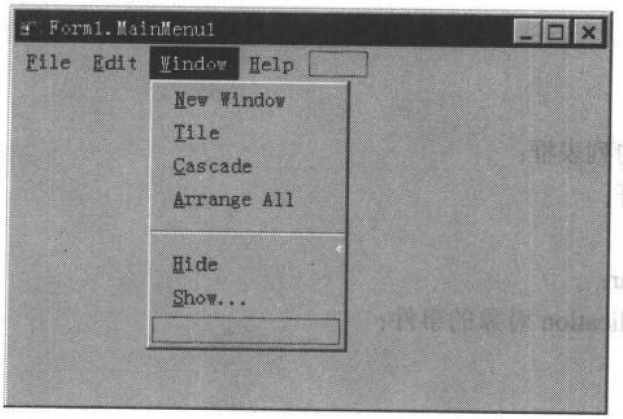

图 1.1　菜单设计器

1.1.2 TMenuItem

Delphi 5 是彻底面向对象的，菜单上的每一个菜单项都是 TMenuItem 对象。下面结合介绍 TMenuItem 对象，讲述菜单设计器的基本操作。

Action 特性

声明：property Action: TBasicAction;

Delphi 5 支持“动作列表”功能。动作列表中预定义了许多常用的操作。Action 特性用于为一个菜单项指定其中一个动作，当用户执行这个动作时，“动作列表”会自动作出反应，而不需要程序通过代码来响应。这个功能的好处是，把用户界面和应用逻辑分开，减少了应用程序的代码行数，有利于程序的维护和调试，并且增加了可靠性。

AutoHotkeys 特性

声明：property AutoHotkeys: TMenuItemAutoFlag;

如果这个特性设为 maAutomatic，每个菜单项将自动具有加速键，并且不会重复。这样，当您在运行期动态地增加菜单项时，菜单项的加速键不会冲突。即使 AutoHotkeys 特性设为 maAutomatic，您仍然可以通过调用 RethinkHotkeys()来调整菜单项的加速键。

AutoLineReduction 特性

声明：property AutoLineReduction: TMenuItemAutoFlag;

如果这个特性设为 maAutomatic，Delphi 将自动去掉多余的或不适当的分隔线。即使 AutoLineReduction 特性设为 maAutomatic，仍然可以通过 RethinkLines()来调整分隔线。

Bitmap 特性

声明：property Bitmap: TBitmap;

这个特性用于指定一个位图，该位图将显示在菜单项的标签旁边。如果 ImageIndex 特性也指定了一个图像，此图像将代替 Bitmap 特性指定的位图显示在菜单项的标签旁边。

Break 特性

声明：property Break: TMenuBreak;

如果一个菜单的命令很多，可以把菜单项分成几栏显示。例如，要把 Help 菜单分成两栏，并且从 Keyboard 命令开始，则可以把 Keyboard 命令的 Break 特性设为 mbBarBreak 或 mbBreak。如果设为 mbBarBreak，在栏与栏之间有一根竖线；如果设为 mbBreak，栏与栏之间只有空格，没有竖线。分栏显示的菜单如图 1.2 所示。

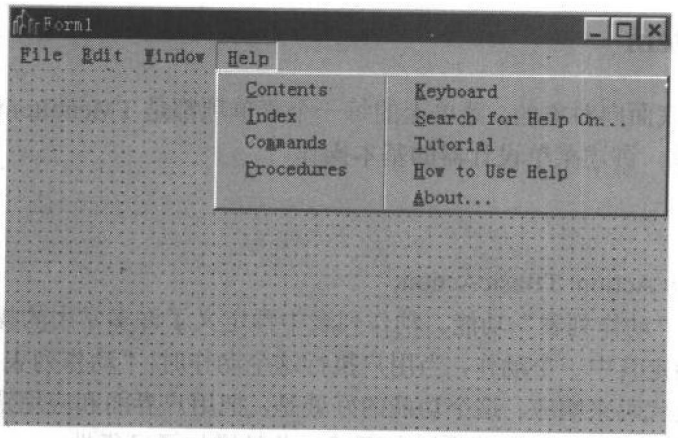

图 1.2 把菜单项分成几栏显示

Caption 特性

声明：property Caption: string;

这个特性用于设置菜单项的标题，如 New、Open、Save 等。如果把 Caption 特性设为“-”，表示其作为菜单项与菜单项之间的分隔线。

可以为菜单项指定加速字符。所谓加速字符，可以参见图 1.2。File 菜单的“F”字符有下划线，Edit 菜单的“E”字符有下划线，Window 菜单的“W”字符有下划线，这些有下划线的字符就是加速字符。程序运行时，用户只要按下 Alt+F 键就相当于单击 File 菜单，按下 Alt+E 键就相当于单击 Edit 菜单，按下 Alt+W 键就相当于单击 Window 菜单。

要为菜单或菜单项指定加速字符，只要在 Caption 特性的某个字母前加一个“&”符号，例如“&File”或“&Edit”，以后“F”字符和“E”字符就会有下划线。

下面的程序示例把应用程序所有的 Form 名称加到 Windows 菜单下，并且在 Form 名称与 Windows 菜单下的原有菜单命令之间加一条分隔线。

```
var
    NewItem: TMenuItem;
    I : integer;
begin
    NewItem := TMenuItem.Create(Self);
    NewItem.Caption := '-';
    Windows.Add(NewItem);
    for I := 0 to Screen.FormCount-1 do
    begin
        NewItem := TMenuItem.Create(Self);
        NewItem.Caption := Screen.Forms[I].Name;
        Windows.Add(NewItem);
    end;
end;
```

Checked 特性

声明：property Checked: Boolean;

如果这个特性设为 True，菜单项的左边将显示一个“✓”，相当于一个复选框。例如，当用户选择“View”菜单上的“ToolBar”命令时，这个命令前就出现一个“✓”，表示要显示工具栏。当用户再次选择这个命令时，这个命令前的“✓”消失，表示不显示工具栏。

类似这样的切换命令，可以参考下面的代码：

```
procedure TForm1. ViewToolBarClick(Sender: TObject);
begin
    ViewToolBar.Checked := not ViewToolBar.Checked;
end;
```

Command 特性

声明：property Command: Word;

这个特性返回菜单项的命令识别号(ID)。当用户选择了某个菜单项，Windows 就会发送 WM_COMMAND 消息到菜单项所在的窗口，消息的 ItemID 字段就是 Command 特性。

下面的代码是一个处理 WM_COMMAND 消息的句柄，它首先检查消息的 ItemID 字段是否与菜单项 MyMenuItem 的命令识别号匹配。若是的话，就显示一个消息框；否则，就调用基类的消息处理句柄，以保证消息总是能得到处理。

```
procedure TForm1.WMCommand(var Msg: TWMCommand);
begin
    if Msg.ItemID = MyMenuItem.Command then
        MessageDlg('This is the Think command', mtInformation, [mbOk], 0);
    inherited;
end;
```

Count 特性

声明：property Count: Integer;

这个只读的特性返回菜单项的子菜单项的数目。程序示例如下：

```
var
    I: Integer;
begin
    for I := 0 to MenuItem1.Count-1 do
        MenuItem1.Items[I].Enabled := False;
end;
```

Default 特性

声明：property Default: Boolean;

如果这个特性设为 True，菜单项将加粗显示，用户只要双击它所在的菜单就相当于单击这个菜单项。假设“File”菜单下有“New”菜单项，如果把“New”菜单项的 Default 特性设为 True，“New”将加粗显示，并且只要双击“File”就等于选择“New”菜单项。

Enabled 特性

声明：property Enabled: Boolean;

如果这个特性设为 False，菜单项将被禁止。这意味着，菜单项将以灰色显示，也不响应键盘和鼠标。下面这段代码把一个菜单项的所有子菜单项都禁止掉：

```
var
    I: Integer;
begin
    for I := 0 to MenuItem1.Count-1 do
        MenuItem1.Items[I].Enabled := False;
end;
```

GroupIndex 特性

声明：property GroupIndex: Byte;

这个特性对于多 Form 的程序特别有用，后面将详细介绍。对于 OLE 客户程序来说，当激活 OLE 对象时，可以把 OLE 服务器程序的菜单合并到客户程序中。

Handle 特性

声明：property Handle: HMENU;

这个特性返回菜单项的句柄，调用某些 Windows 的 API 时需要用到菜单项句柄。只有 Count 特性大于零时，Handle 特性才是有意义的。

HelpContext 特性

声明：property HelpContext: THelpContext;

这个特性用于指定菜单项在帮助系统中的上下文编号。帮助系统的每一屏都对应着唯一的上下文编号(Context ID)。

Hint 特性

声明：property Hint: string;

当鼠标指向某个菜单项时，在该菜单项的旁边将弹出提示，内容由 Hint 特性设定。

ImageIndex 特性

声明：property ImageIndex: Integer;

这个特性用于指定图像列表中的一个序号，该序号的图像将显示在菜单项的标题旁边。图像列表是由 TMainMenu 或 TPopupMenu 的 Images 特性指定的。