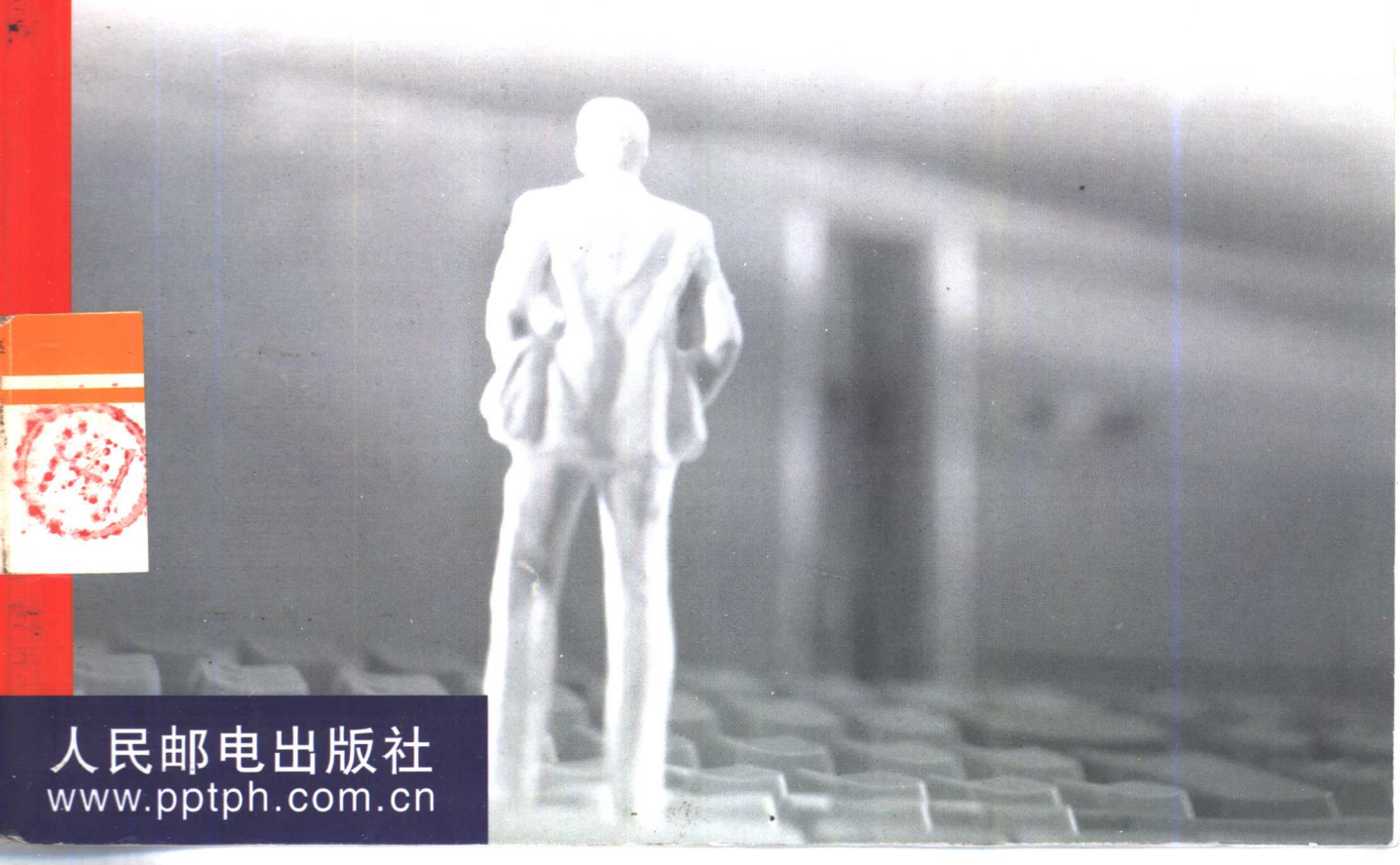
C++ Builder 5

高级编程技术

——GUI编程

徐新华　等编著



人民邮电出版社
www.pptph.com.cn

00100955

C++ Builder 5
高级编程技术

—— GUI 编程

徐新华　等 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用 C++ Builder 5 设计应用程序的图形界面，包括常用的 Windows 控件、公共对话框、Win32 公共控件、剪贴板、DDE、OLE、Form 和应用程序、屏幕与打印机、图像、多线程和 DirectDraw 等内容。

C++ Builder 5 是一个完全面向对象的编程工具。众多长期从事编程的人员从实践中体会到，只要真正领会了面向对象的编程思想，即使是很高深的编程领域，诸如 COM、ActiveX、CORBA、MIDAS 都不难掌握。所以，本书的重点是面向对象编程。

本书内容全面而又不失简洁，例子丰富，既可以作为广大读者学习 C++ Builder 5 的入门指导书，也可以作为程序员编程时的参考手册。

TS121/14

C++ Builder 5 高级编程技术——GUI 编程

◆ 编　　著　徐新华　等
责任编辑　王晓明
◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@pptph.com.cn
网址　http://www.pptph.com.cn
北京汉魂图文设计有限公司制作
北京鸿佳印刷厂印刷
新华书店总店北京发行所经销
◆ 开本：787×1092　1/16
印张：29.25
字数：734 千字　　2000 年 12 月第 1 版
印数：1－5 000 册　　2000 年 12 月北京第 1 次印刷
ISBN 7-115-09011-4/TP·1987

定价：43.00 元

前　言

C++ Builder 5 是用于电子商务、Internet 应用和数据库编程等开发工作的最佳工具之一。C++ Builder 5 最接近 ISO 的 C++标准；同时支持 COM 与 CORBA 两大分布式计算规范；C++ Builder 5 支持 ADO，从而可以访问更多的数据；C++ Builder 5 增加了数据模块设计器，可以轻松地创建和维护数据模块；通过 InterBase Express(IBX)，C++ Builder 5 集成了对 InterBase 数据库的访问，不再需要借助于 BDE；MIDAS 与新增加的 Internet Express 配合使用，使客户(Web 浏览器)能更方便地与 MIDAS 服务器交互；C++ Builder 5 内建了全球 ORB 分发数量最多的 VisiBroker 4.0，集成了 CORBA IDL 编译器，单一步骤就能生成 CORBA 对象；TeamSource 是一个集成的工作流程管理工具，大大简化了团队开发；运用 Integrated Translation Environment(ITE)，可以方便地进行软件的本地化和国际化。另外，C++ Builder 5 可以很方便地操纵 Microsoft Office 97/2000 的文档和程序，这在进行企业级开发时是很重要的。

为了帮助广大用户全面、准确地掌握 C++ Builder 5 的编程思想和用法，作者专门编写了这套《C++ Builder 5 高级编程技术》。这套丛书与作者以前编写的《C++ Builder 4 高级编程丛书》之间有着继承性，但在内容上，因该软件版本的升级又有很大的区别。这套《C++ Builder 5 高级编程技术》可以使读者在掌握了 C++ Builder 4 编程技术以后，进一步学习使用新版本软件编程的方法和技巧。考虑到很多程序员已经初步掌握了 C++ Builder 5 的基本用法，因此本套丛书内容的重点放在了编程技术的进一步精通和提高上。

本套丛书分为 4 册。第 1 册介绍 IDE 与 OOP 编程，第 2 册介绍 GUI 编程，第 3 册介绍 Database 与 MIDAS 编程，第 4 册介绍 COM、CORBA 与 Internet 编程。

本书是此套丛书的第 2 册，全面深入地介绍了如何用 C++ Builder 5 设计应用程序的图形界面，包括常用的 Windows 控件、公共对话框、Win32 公共控件、剪贴板、DDE、OLE、Form 和应用程序、屏幕与打印机、图像、多线程和 DirectDraw 等内容。

本书主要由徐新华编写，另外，参加本书编写工作的还有顾洪均、凌晨、张莉、郭平等人。由于我们的水平有限，再加上时间很紧，因此，尽管我们作了比较严格的审核和测试，但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

作　者
2000 年 8 月

此为试读，需要完整PDF请访问：www.ertongbook.com

目　录

第一章　设计图形界面

C++ Builder 5 是开发 Windows 应用程序的最佳工具。使用 C++ Builder 5 可以轻松地制作出程序的图形界面。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伤透脑筋；现在，有了 C++ Builder 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 C++ Builder 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 打开菜单设计器

要打开菜单设计器有三种方式：

一是直接在 Form 上双击 TMainMenu 元件或 TPopupMenu 元件。

二是在 Object Inspector 上单击 Items 特性的省略号按钮。

三是用鼠标右键单击 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。

1.1.2 TMenuItem

C++ Builder 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 Classes::TBasicAction* Action;

C++ Builder 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，当用户进行这个动作时，“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。

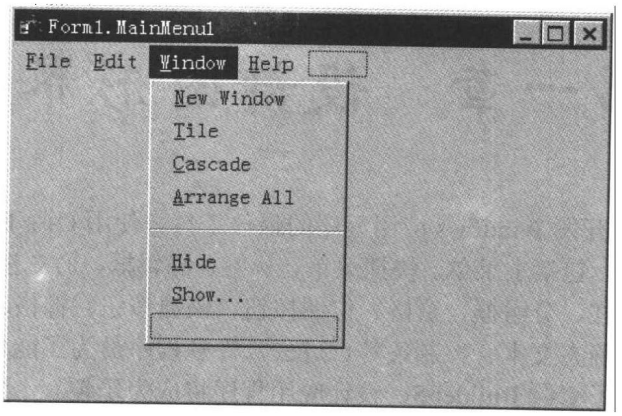


图 1.1　菜单设计器

AutoHotkeys 特性

声明：_property TMenuItemAutoFlag AutoHotkeys;

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，当在运行期动态地增加菜单项时，菜单项的加速键不会冲突。

不过，即使 AutoHotkeys 特性设为 maAutomatic，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 TMenuItemAutoFlag AutoLineReduction;

如果这个特性设为 maAutomatic，C++ Builder 将自动去掉多余的或不适当的分隔线。

不过，即使 AutoLineReduction 特性设为 maAutomatic，仍然可以通过调用 RethinkLines()来调整分隔线。

Bitmap 特性

声明：_property Graphics::TBitmap* Bitmap;

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 TMenuBreak Break;

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。

Caption 特性

声明：_property System::AnsiString Caption;

此为试读，需要完整PDF请访问：www.ertongbook.com

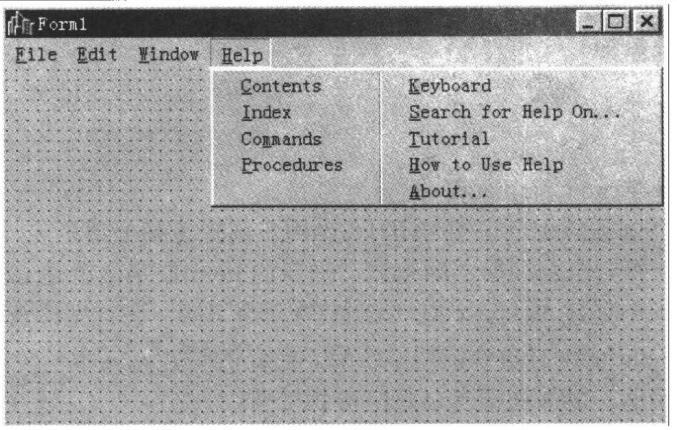


图 1.2　把菜单分成几栏显示

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。

```
TMenuItem *NewItem = new TMenuItem;
NewItem->Caption = "-";
Window1->Add(NewItem);
for (int I = 0; i < Screen->FormCount; I++)
{
    NewItem = new TMenuItem;
    NewItem->Caption = Screen->Forms[I]->Name;
    Window1->Add(NewItem);
}
```

Checked 特性

声明：_property bool Checked;

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，“View”菜单上有一个“ToolBar”命令。当用户选择这个命令时，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：

```
void _fastcall TForm1::ViewToolBarClick(TObject *Sender)
{
```

```
    ViewToolBar->Checked = !ViewToolBar->Checked;
}
```

Command 特性

声明：_property Word Command;

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。

```
MESSAGE void _fastcall TForm1::WMCommand(Messages::TWMCommand &Message)
{
    if (Message->ItemID = = MyMenuItem->Command)
        Application->MessageBox("This is the my command", "Yes", MB_OK);
    else
        TForm::WMCommand(Message);
}
```

Count 特性

声明：_property int Count;

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：

```
for (int i = 0; i < MenuItem1->Count; i++)
{
    MenuItem1->Items[i]->Enabled = false;
}
```

Default 特性

声明：_property bool Default;

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 bool Enabled;

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：

```
for (int i = 0; i < MenuItem1->Count; i++)
{
    MenuItem1->Items[i]->Enabled = false;
}
```

GroupIndex 特性

声明：_property Byte GroupIndex;

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，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 HMENU Handle;

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 Classes::THelpContext HelpContext;

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 System::AnsiString Hint;

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 int ImageIndex;

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。

Items 特性

声明：_property TMenuItem* Items[int Index];

这个只读的特性返回菜单项的所有子菜单项组成的数组。假设“File”菜单下有“New”、“Open”和“Save”三个菜单项，则 FileMenu->Items[2]就是“Save”菜单项。

MenuIndex 特性

声明：_property int MenuIndex;

这个特性指出了菜单项在其“父”菜单中的序号。可以修改这个特性，从而改变菜单项的位置。要说明的是，那些 Visible 特性设为 false 的菜单项也有序号。

Parent 特性

声明：_property TMenuItem* Parent;

这个特性返回菜单项的“父”菜单。如果菜单项已经处于顶层，就返回 Items 特性。

RadioItem 特性

声明：_property bool RadioItem;

这个特性用于一组互斥的菜单项。如果 RadioItem 特性设为 true，当菜单项的 Checked 特性为 true 时不是显示一个“✓”而是显示一个小圆点。读者可以打开 Windows 的“资源管理器”并下拉“查看”菜单，体会“✓”和小圆点的用法。

ShortCut 特性

声明：_property TShortCut ShortCut;

这个特性用于设置菜单项的快捷键，这与加速字符是两个不同的概念。菜单项的快捷键通常是一些键的组合，如 Alt+Shift+F6、Ctrl+G 等，快捷键显示在菜单项标题的右边。

一般来说，给菜单项设快捷键是在设计期进行的，不过有时候菜单项是在运行期动态地生成的，因此要给菜单项设快捷键也必须相应地在运行期进行。C++ Builder 5 提供了几个函数用于这方面的操作：ShortCut、ShortCutToKey、ShortCutToText、TextToShortCut。

Visible 特性

声明：_property bool Visible;

如果这个特性设为 false，可以把菜单项隐去(不是删除)。

Add 函数

声明：void _fastcall Add(TMenuItem* Item);

这个函数用于增加一个或一组菜单项。程序示例如下：

```
for (int I = 0; i < Screen->FormCount; I++)
{
    NewItem = new TMenuItem;
    NewItem->Caption = Screen->Forms[I]->Name;
    Window1->Add(NewItem);
}
```

Clear 函数

声明：void _fastcall Clear(void);

这个函数用于清除菜单项的所有子菜单项。

Click 函数

声明：virtual void _fastcall Click(void);

调用这个函数相当于用鼠标或键盘选择这个菜单项，将触发 OnClick 事件。

Delete 函数

声明：void _fastcall Delete(int Index);

这个函数用于删除一个子菜单项，Index 参数指定子菜单项的序号。程序示例如下：

```
void _fastcall TForm1::Button1Click(TObject *Sender)
{
    File1->Delete(1);
```

}

上例中，当用户按下按钮 Button1，将删除“File”菜单的第二个子菜单项，因为菜单项的序号是从 0 开始的。

Find 函数

声明：TMenuItem* _fastcall Find(AnsiString ACaption);

这个函数用于搜索并返回一个子菜单项，ACaption 参数用于指定要搜索的子菜单项的标签。如果没有找到，就返回 NULL。

GetImageList 函数

声明：TCustomImageList* _fastcall GetImageList(void);

这个函数返回供应图像给这个菜单项的图像列表。如果没有，就返回 NULL。

GetParentMenu 函数

声明：TMenu* _fastcall GetParentMenu(void);

这个函数返回菜单项所在的 TMainMemu 或 TPopupMemu 元件。

IndexOf 函数

声明：int _fastcall IndexOf(TMenuItem* Item);

这个函数返回一个子菜单在菜单中的序号。如果没有找到指定的子菜单，函数返回-1。

下面这个例子首先检查 Open1 菜单项是否在菜单 File1 中。若是，则删除它。

```
void _fastcall TForm1::Button1Click(TObject *Sender)
{
    If (File1->IndexOf(Open1) <> -1) File1->Delete(IndexOf(Open1));
}
```

Insert 函数

声明：HIDESBASE void _fastcall Insert(int Index, TMenuItem* Item);

这个函数在菜单的指定位置插入一个菜单项。程序示例如下：

```
void _fastcall TForm1::Button1Click(TObject *Sender)
{
    TMenuItem *NewItem = new TMenuItem(FileMenu);
    try
    {
        NewItem->Caption = "Form1";
        FileMenu->Insert(1, NewItem);
    }
    catch (Exception &e)
    {
        delete NewItem;
```

```
            throw;
        }
    }
```

InsertNewLineAfter 函数

声明：int _fastcall InsertNewLineAfter(TMenuItem* AItem);

这个函数在指定的菜单项之后插入一个分隔线，并返回这个分隔线在 Items 特性中的序号。如果 AItem 参数指定的菜单项在 Items 特性中没有找到，这个函数将触发异常。

InsertNewLineBefore 函数

声明：int _fastcall InsertNewLineBefore(TMenuItem* AItem);

这个函数在指定的菜单项之前插入一个分隔线，并返回这个分隔线在 Items 特性中的序号。如果 AItem 参数指定的菜单项在 Items 特性中没有找到，这个函数将触发异常。

IsLine 函数

声明：bool _fastcall IsLine(void);

如果这个菜单项是一条分隔线，这个函数就返回 true。

NewBottomLine 函数

声明：int _fastcall NewBottomLine(void);

这个函数用于在所有菜单项之后插入一条分隔线。

NewTopLine 函数

声明：int _fastcall NewTopLine(void);

这个函数用于在所有菜单项之前插入一条分隔线。

Remove 函数

声明：HIDESBASE void _fastcall Remove(TMenuItem* Item);

这个函数用于删除某个子菜单项。与 Delete()不同的是，必须指定要删除的子菜单项，而不是子菜单项的序号。如果 Item 参数所指定的子菜单项不存在，将触发异常。

RethinkHotkeys 函数

声明：bool _fastcall RethinkHotkeys(void);

这个函数将调整子菜单项的标签，使它们都有相异的加速键。如果有任何一个菜单项的 Caption 特性被调整，这个函数将返回 true。

RethinkLines 函数

声明：bool _fastcall RethinkLines(void);

这个函数将去掉多余的分隔线。例如，开头或末尾出现分隔线肯定是多余的；两条分隔线紧挨在一起，其中一条也是多余的。如果至少去掉了一条分隔线，这个函数将返回 true。

此为试读，需要完整PDF请访问：www.ertongbook.com