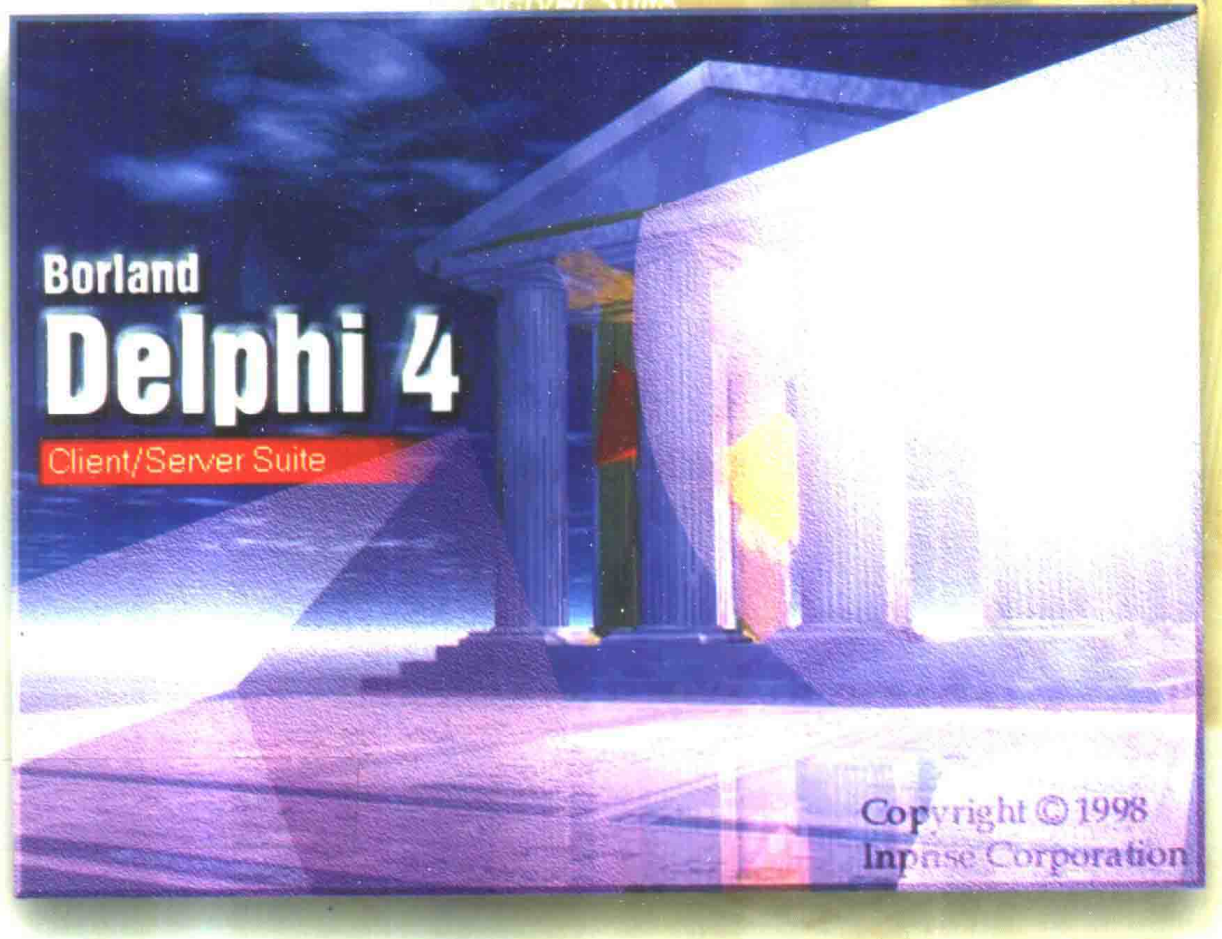
Delphi 4 高级编程丛书之二

GUI 编程技术

徐新华　编著



人民邮电出版社
PEOPLE'S POSTS & TELECOMMUNICATIONS PUBLISHING HOUSE

Delphi 4 高级编程丛书之二

GUI 编程技术

徐新华　编著

人 民 邮 电 出 版 社

Delphi 4 高级编程丛书之二

GUI 编程技术

◆ 编　　著　徐新华
责任编辑　顾　翀
◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街 14 号
北京朝阳展望印刷厂印刷
新华书店总店北京发行所经销

◆ 开本：787 × 1092　1/16
印张：28.25
字数：704 千字　　1998 年 11 月第 1 版
印数：6 001 – 11 000 册　　1999 年 5 月北京第 2 次印刷
ISBN 7-115-07375-9/TP·832

定价：42.00 元

内容提要

本书是《Delphi 4 高级编程丛书》的第二册，书中全面深入地介绍了 Delphi 4 的图形用户界面（GUI）的编程技术。

本书内容全面，层次清晰，语言通俗简洁。全书内容涉及到 VCL 的结构、图形界面、公共对话框、系统功能接口、Win32 控件、Form 和 Application、图像、屏幕与打印机、字体、字符串列表、注册表、剪贴板、多线程、Open Tools API 等。最后，还介绍了怎样编写用户自己的元件。本书通过丰富的实例，讲解的不仅仅是 Delphi 4 的编程方法，更重要的是传授一种面向对象的编程思想。

本书既可以作为广大读者学习 Delphi 4 的指导书，也可以作为程序开发设计人员的编程参考手册。

前 言

Delphi 4 是 Borland 公司更名为 Inprise 后推出的一个具有战略意义的产品，它标志着 Inprise 的工作重心已经从桌面转移到跨平台的分布式应用。Delphi 4 中的 MIDAS 技术奠定了它在业界的领先地位。

为了帮助广大用户全面、准确地掌握 Delphi 4 的编程思想和用法，我们编写了这套《Delphi 4 高级编程丛书》，主要是针对那些已初步掌握了 Delphi 4 的基本用法而又需要进一步提高和精通的读者。本丛书紧紧把握 Delphi 4 的基本特征，即面向对象，重点从“类”这一层次把 Delphi 4 的编程思想讲透。我们的体会是，只要深刻领会了面向对象的编程思想，就很容易理解那些看上去高深莫测的领域如 COM、ActiveX、CORBA、MIDAS。

本套丛书分为四册，第一册是《IDE 和 Object Pascal 语言》，第二册是《GUI 编程技术》，第三册是《Database 和 MIDAS 编程技术》，第四册是《COM、ActiveX、CORBA、Internet/Intranet 编程技术》。

本书是此套丛书的第二册，主要介绍 Delphi 4 的 GUI 编程技术，各章内容如下：

第一章介绍 VCL 的结构，重点是几个公共祖先类包括 TObject、TPersistent、TComponent、TControl、TWinControl、TGraphicControl 和 TCustomControl。

第二章介绍怎样建立应用程序的图形界面，包括菜单、标签、编辑框、按钮、复选框、单选框、列表框、组合框、滚杆、分组框、窗格、动作列表、栅格、图像、几何图形、分界、滚动箱、尺寸调节杆、静态文本、TControlBar 等。

第三章介绍公共对话框，包括“打开”、“另存为”、“字体”、“颜色”、“打印”、“打印设置”、“查找”、“替换”等对话框。

第四章介绍系统控制功能，包括定时器、画板、媒体播放器、OLE 客户、动态数据交换、文件列表框、目录列表框、驱动器组合框和文件类型过滤器。

第五章介绍 Win32 公共控件，包括 TAB 控件、多页控件、图像列表、RTF 编辑器、跟踪条、进程条、加/减控件、热键控件、AVI 播放器、日期和时间控件、月历、树状视图、列表视图、表头控件、状态栏、工具栏、“酷”和 TPageScroller。

第六章介绍怎样操纵 Form 和应用程序，重点是 TForm 和 TApplication，另外还介绍了怎样操纵 MDI、控制台程序和服务程序。

第七章介绍怎样操纵图像，包括 TCanvas、TPen、TBrush、TPicture、TBitmap、TMetafile、TMetafileCanvas，另外还分析了一个示范程序。

第八章介绍怎样操纵屏幕和打印机，重点是 TScreen 和 TPrinter，另外还介绍了 Writeln、DEVMODE 结构和打印机控制码。

第九章介绍怎样操纵字体、字符串列表、注册表和剪贴板。

第十章介绍有关多线程的编程技术，包括创建线程对象、设置线程的优先级、定义线程函数、锁定和阻塞、依赖另一个线程的执行结果、挂起和唤醒，另外还分析了一个典型的多线程应用程序。

第十一章介绍怎样使用 Open Tools API，并实际建立了两种典型的“专家”。

第十二章介绍怎样编写自己的元件，包括选择祖先类、建立元件框架、加入特性、方法和事件、编写特性编辑器和元件编辑器、安装元件。

本书由徐新华执笔，参加编写的人员有郭平、周学成、徐存聪等。

由于水平有限，再加上时间很紧，尽管我们作了严格的审核和测试，书中难免存在错误，敬请广大读者不吝赐教，我们谨在此表示感谢。

考虑到 Delphi 4 中增加了许多崭新的技术，有些技术具有相当的难度，为了帮助广大程序员更好地掌握这个优秀的开发工具，我们愿意为购买此书的读者提供技术咨询，我们将热情、及时地答复读者提出的问题。

电子函件：p_inprise@mail.263.net.cn

作者

1998 年 7 月

目 录

第一章
VCL 的结构

1.1 概 述 …… 1
1.2 TObject …… 2
1.3 TPersistent …… 5
1.4 TComponent …… 6
1.5 TControl …… 10
1.6 TWinControl …… 25
1.7 TGraphicControl …… 37
1.8 TCustomControl …… 38

第二章
设计应用程序的图形界面

2.1 菜 单 …… 39
2.1.1 打开菜单设计器 …… 39
2.1.2 TMenuItem 对象 …… 40
2.1.3 菜单嵌套 …… 46
2.1.4 应用菜单模板和菜单资源 …… 47
2.1.5 TMemu 对象 …… 47
2.1.6 TMainMenu 元件的特性、方法、事件 …… 49
2.1.7 怎样在运行期控件菜单 …… 50
2.2 快捷菜单 …… 50
2.3 标 签 …… 52
2.4 编辑框 …… 55
2.5 多行文本编辑器 …… 60
2.6 命令按钮 …… 62
2.7 复选框 …… 63
2.8 单选框 …… 65
2.9 列表框 …… 67

2.10 组合框 …… 75
2.11 滚 杆 …… 78
2.12 分组框 …… 81
2.13 单选分组框 …… 81
2.14 窗 格 …… 83
2.15 动作列表 …… 85
2.15.1 动作列表机制的三个环节 …… 85
2.15.2 管理动作列表 …… 85
2.15.3 为客户指定一个动作 …… 87
2.16 位图按钮 …… 87
2.17 快捷按钮 …… 89
2.18 按格式输入编辑框 …… 90
2.19 自绘栅格 …… 93
2.20 字符串栅格 …… 98
2.21 图 像 …… 100
2.22 几何图形 …… 102
2.23 分 界 …… 103
2.24 滚动箱 …… 104
2.25 带复选框的列表框 …… 106
2.26 尺寸调节杆 …… 107
2.27 静态文本 …… 110
2.28 TControlBar …… 111

第三章
公共对话框

3.1 TCommonDialog …… 113
3.2 “打开”对话框 …… 114
3.3 “另存为”对话框 …… 119
3.4 能预览图像的“打开”对话框 …… 119
3.5 能预览图像“另存为”对话框 …… 120
3.6 “字体”对话框 …… 121
3.7 “颜色”对话框 …… 123
3.8 “打印”对话框 …… 124
3.9 “打印设置”对话框 …… 127
3.10 “查找”对话框 …… 128
3.11 “替换”对话框 …… 130

第四章
实现系统控制功能

4.1 定时器 …… 133
4.2 画 板 …… 134
4.3 媒体播放器 …… 135
4.4 OLE 客户 …… 144
4.4.1 创建 OLE 客户的一般步骤 …… 145
4.4.2 TOleContainer 元件的特性 …… 147
4.4.3 TOleContainer 元件的方法 …… 150
4.4.4 TOleContainer 元件的事件 …… 155
4.5 动态数据交换 …… 156
4.5.1 开发 DDE 程序的一般步骤 …… 156
4.5.2 TDDEClientConv 元件 …… 157
4.5.3 TDDEClientItem 元件 …… 160
4.5.4 TDDEServerConv 元件 …… 161
4.5.5 TDDEServerItem 元件 …… 162
4.6 文件列表框 …… 163
4.7 目录列表框 …… 166
4.8 驱动器组合框 …… 169
4.9 文件类型过滤器 …… 170

第五章
Win32 公共控件

5.1 TAB 控件 …… 173
5.2 多页控件 …… 177
5.2.1 Win31 风格的多页控件 …… 177
5.2.2 Win95/98 风格的多页控件 …… 178
5.2.3 TPageControl 元件的特性、方法和事件 …… 179
5.2.4 TTabSheet 对象 …… 181
5.2.5 在两个多页控件之间拖放页 …… 182
5.3 图像列表 …… 184
5.3.1 怎样在设计期建立图像列表 …… 184
5.3.2 怎样在运行期动态建立图像列表 …… 185
5.3.3 TImageList 的特性、方法和事件 …… 185
5.3.4 用 TImageList 元件实现屏幕捕捉器 …… 191
5.4 RTF 编辑器 …… 191
5.4.1 TRichEdit 元件的特性和方法 …… 192
5.4.2 TTextAttributes 对象 …… 198
5.4.3 怎样在运行期设置字符格式 …… 198
5.4.4 TParaAttributes 对象 …… 199

5.4.5 动态显示当前插入点所在的行和列 …… 200
5.5 跟踪条 …… 201
5.6 进程条 …… 202
5.7 加/减控件 …… 205
5.8 热键控件 …… 207
5.9 AVI 播放器 …… 208
5.10 日期和时间控件 …… 211
5.11 月 历 …… 214
5.12 树状视图 …… 216
5.12.1 TTreeView 元件的特性、方法和事件 …… 216
5.12.2 TTreeNode 对象 …… 225
5.12.3 TTreeNodes 对象 …… 232
5.12.4 树状视图与 Master/Detail 数据库 …… 234
5.12.5 用树状视图描述类的继承关系 …… 235
5.13 列表视图 …… 236
5.14 表头控件 …… 250
5.15 状态栏 …… 253
5.15.1 怎样用 TPanel 元件制作状态栏 …… 254
5.15.2 怎样用 TStatusBar 元件制作状态栏 …… 254
5.15.3 TStatusBar 元件的特性、方法和事件 …… 255
5.16 工具栏 …… 257
5.16.1 用 TPanel 元件制作工具栏 …… 258
5.16.2 用 TToolBar 元件制作工具栏 …… 258
5.16.3 怎样在运行期操纵工具栏上的快捷按钮 …… 259
5.16.4 TToolBar 元件特性、方法和事件 …… 260
5.16.5 TToolButton 对象 …… 262
5.17 “酷” …… 263
5.17.1 在设计期建立段 …… 264
5.17.2 在运行期建立段 …… 264
5.17.3 TCoolBar 元件的特性、方法和事件 …… 265
5.17.4 TCoolBand 对象 …… 267
5.18 TPageScroller …… 269

第六章
操纵 Form 和应用程序

6.1 TScrollingWinControl …… 271
6.2 TCustomForm …… 272
6.3 TForm …… 285
6.4 记忆 Form 关闭前的状态 …… 286
6.5 MDI 程序 …… 287

6.5.1 MDI 程序的“父”Form …… 288
6.5.2 MDI 程序的“子”Form …… 288
6.5.3 自动创建“子”Form 的实例 …… 288
6.5.4 怎样在运行期生成“子”Form 的实例 …… 289
6.5.5 合并菜单 …… 289
6.5.6 排列打开的子窗口 …… 290
6.6 控制台程序 …… 291
6.7 操纵应用程序 …… 293
6.7.1 TApplication 的特性 …… 294
6.7.2 TApplication 的方法 …… 297
6.7.3 怎样响应运行期元件的事件 …… 302
6.7.4 TApplication 的事件 …… 303
6.8 应用程序的实例 …… 307
6.9 服务程序 …… 308

第七章
操纵图像
7.1 TCanvas …… 313
7.2 TPen …… 323
7.3 TBrush …… 327
7.4 TPicture …… 329
7.5 TBitmap …… 330
7.6 TMetafile …… 336
7.7 TMetafileCanvas …… 337
7.8 如何设计一个作图软件 …… 338

第八章
操纵屏幕和打印机
8.1 TScreen …… 345
8.2 显示和打印的一致性 …… 351
8.3 TPrinter 对象 …… 352
8 4 Writeln 过程 …… 356
8.5 DEVMODE 结构 …… 357
8.6 打印机控制码 …… 358

第九章
操纵字体、字符串列表、
注册表和剪贴板
9.1 TFont …… 361
9.2 TStrings …… 364

9.3 TStringList …… 370
9.4 TRegistry …… 372
9.5 TClipboard …… 378

第十章
多线程应用程序

10.1 多线程概述 …… 383
10.2 创建线程对象 …… 384
10.3 设置线程的优先级 …… 385
10.4 定义线程函数 …… 386
10.4.1 访问 VCL …… 386
10.4.2 线程局部变量 …… 387
10.4.3 检查 Terminated 特性 …… 388
10.5 锁定和阻塞 …… 388
10.6 依赖另一个线程的执行结果 …… 389
10.7 挂起和唤醒 …… 391
10.8 一个典型的多线程应用程序 …… 392

第十一章
Open Tools API

11.1 怎样创建专家(Expert) …… 399
11.2 怎样注册专家 …… 403
11.3 IDE 的服务接口 …… 404
11.4 标准型专家的示例 …… 405
11.5 加载型专家的示例 …… 410

第十二章
编写自己的元件

12.1 选择祖先类 …… 413
12.1.1 公共祖先类 …… 413
12.1.2 现有的元件 …… 414
12.1.3 元件模板 …… 414
12.1.4 我们的建议 …… 414
12.2 建立元件框架 …… 415
12.3 手工建立元件框架 …… 416
12.4 加入特性 …… 417
12.4.1 加入简单型的特性 …… 417
12.4.2 加入枚举型的特性 …… 418
12.4.3 加入集合型的特性 …… 419
12.4.4 加入对象型特性 …… 420

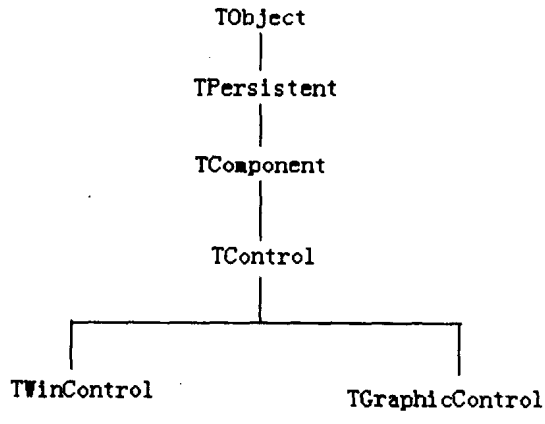
12.4.5 加入数组型特性 …… 422
12.4.6 给出特性的默认值 …… 423
12.5 加入方法 …… 424
12.5.1 方法的可见性 …… 424
12.5.2 避免内部相关性 …… 424
12.5.3 给方法命名 …… 424
12.5.4 加入标准的方法 …… 424
12.5.5 加入虚拟方法 …… 425
12.5.6 加入动态方法 …… 425
12.5.7 加入抽象方法 …… 426
12.6 加入事件 …… 426
12.7 编写特性编辑器 …… 429
12.7.1 选择合适的祖先 …… 429
12.7.2 重载某些方法 …… 430
12.7.3 注册特性编辑器 …… 433
12.8 如何编写元件编辑器 …… 434
12.9 把元件加到 IDE 中 …… 437

第一章
VCL 的结构

Delphi 4 是开发图形用户界面(GUI)的最佳编程工具，这主要得益于它的符合工业标准的 VCL。从本章开始将详细介绍 VCL 中的元件。尽管这些元件千差万别，但它们都是从几个共同的祖先类继承下来的。因此，这些元件具有某些相同或相似的基本特征。

1.1 概 述

VCL(Visual Component Library 的缩写)是 Delphi 4 的核心，它是完全面向对象的。VCL 中的所有对象都存在着继承与被继承的关系，下图是 VCL 结构的示意：



从示意图可以看出，最顶层的是 TObject，它是一切对象的祖先类。

TPersistent 是 TObject 的下一级继承者。它是一个抽象类，提供了对象之间相互赋值和读写流的能力。

TComponent 又是 TPersistent 的继承者。这个类是 VCL 中所有元件的祖先类。TComponent 定义了所有元件最基本的行为。

尽管所有元件都是从 TComponent 继承下来的，但直接继承的只有几个非可视的元件，如 TTimer 和 TDataSource 等。其他元件则是从 TComponent 的下级 TControl 继承下来的。从 TControl 继承下来的元件是可视的，可视元件也称为控件。

TWinControl 和 TGraphicControl 这两个类都是从 TControl 继承下来的。这两个类的区别正如它们的名字一样，从 TWinControl 继承下来的主要是按钮、对话框、列表框等有窗口句柄的控件，这些控件占用 Windows 的资源，并且允许用户输入。从 TGraphicControl 继承下来的控件，例如 TLabel、TSpeedButton 等则没有窗口句柄，不占用 Windows 资源，也不能接受键盘的输入，使用这一类控件的好处在于节约资源。

VCL 的面向对象还体现在它的可扩展性，可以选择其中一个元件作为祖先类，派生出一个新的元件，并把新创建的元件加入到 VCL 中。

1.2 TObject

TObject 是 VCL 中所有对象的祖先类，它定义了操纵对象的基本方法。其中，有的是类方法，用于返回对象的类型信息；有的是虚拟方法，能够在派生类中重载。

ClassInfo 类方法

声明：class Function ClassInfo: Pointer;

这个类方法返回一个指针，这个指针指向对象的运行期类型信息表(RTTI)。注意：类型信息一般是由 Delphi 4 的 IDE 使用的，应用程序一般不必与它打交道。

ClassName 类方法

声明：class Function ClassName: ShortString;

这个类方法返回一个对象实例的类名。注意：可以用祖先类的变量来引用派生类的对象实例。用此变量来调用 ClassName 时，返回的是派生类的类名，而不是变量本身的类型。

ClassNameIs 类方法

声明：class Function ClassNameIs(const Name: string): Boolean;

这个类方法用于判断对象的类名是否与 Name 参数匹配，如果匹配，就返回 True。

ClassParent 类方法

声明：class Function ClassParent: TClass;

这个类方法返回类的祖先类，对于 TObject 来说，结果肯定是 NIL，因为 TObject 已经是最顶层的了，它没有“父”。应用程序一般不要直接调用此方法。

下面这个示范程序演示了怎样在一个列表框中列出一个元件的所有祖先类：

```
Procedure TForm1.Button1Click(Sender: TObject);
```

```
Var
    ClassRef: TClass;
Begin
    ListBox1.Clear;
    ClassRef := Sender.ClassType;
    While ClassRef <> NIL Do
    Begin
        ListBox1.Items.Add(ClassRef.ClassName);
        ClassRef := ClassRef.ClassParent;
    End;
End;
```

上述程序执行后，列表框中将依次出现 TButton、TButtonControl、TWinControl、TControl、TComponent、TPersistent、TObject。

Create 构造

声明：constructor Create;

构造的作用是建立类的对象实例，并且对其进行初始化。

DefaultHandler 过程

声明：procedure DefaultHandler(var Message); virtual;

在消息处理句柄中，如果调用 Inherited，但祖先类没有提供处理消息的默认句柄，就会自动调用 DefaultHandler。DefaultHandler 什么也没干，但 TObject 的派生类可以重载它。

Destroy 析构

声明：destructor Destroy; virtual;

析构的作用是删除对象实例。TObject 的 Destroy 是虚拟的，派生类通常要对它重载，针对特定的对象做一些“清场”的工作。重载 Destroy 时，一般在最后要加上这么一句：

Inherited Destroy;

不过，最好调用 Free 而不要直接调用 Destroy 来删除对象实例，因为 Free 在删除对象实例之前会检查对象实例是否存在，这样就能避免对值为 NIL 的指针操作。

Dispatch 过程

声明：procedure Dispatch(var Message);

这个过程用于调用对象的消息处理方法。究竟调用哪个消息处理方法，由 Message 参数指定的消息来决定。如果对象对该消息没有提供处理方法，Dispatch 就会从祖先类中逐级查找，如果祖先类中也没有找到，就调用 DefaultHandler。

Message 是个无类型的参数。从语法上讲，可以传递任何类型的数据，不过，最好是 TMessage 类型，这是 Object Pascal 语言的记录类型。

Free 过程

声明：Procedure Free;

这个过程用于删除对象实例，释放对象占用的内存。正如前面提到的那样，Free 能够检查对象实例是否存在，因此，调用 Free 是比较安全的。

对于 Form 来说，最好调用 Release 来删除，因为 Release 能够保证等 Form 和 Form 上元件的所有事件句柄都执行完毕后再删除 Form 自己。

FreeInstance 过程

声明：Procedure FreeInstance; virtual;

FreeInstance 是与 NewInstance 配对使用的，NewInstance 用于建立一个新的对象实例，而 FreeInstance 用于删除 NewInstance 建立的对象实例。

注意：FreeInstance 一般是由析构自动调用的，应用程序一般不要直接调用它。除非派生类重载了 NewInstance，才需要相应地重载 FreeInstance。

GetInterfaceTable 类方法

声明：class Function GetInterfaceTable: PInterfaceTable;

其中，PInterfaceTable 的声明如下：

```
Type
    PInterfaceTable = ^TInterfaceTable;
    TInterfaceTable = Record
        EntryCount: Integer;
        Entries: array[0..9999] of TInterfaceEntry;
    End;
```

这个类方法返回一个记录的指针，这个记录描述了类所实现的所有接口(不包括祖先类实现的接口)。如果要获得祖先类实现的接口，首先要调用 GetParentClass 返回祖先类，再由祖先类调用 GetInterfaceTable。

InheritsFrom 类方法

声明：class Function InheritsFrom(AClass: TClass): Boolean;

这个类方法用于判断两个类之间的继承关系，如果 AClass 参数指定的类是对象的祖先类，这个类方法就返回 True，否则就返回 False。

InitInstance 类方法

声明：class Procedure InitInstance(Instance: Pointer): TObject;

这个类方法一般不需要直接调用，它是由 NewInstance 自动调用的，用于对新建立的对象实例初始化。注意：InitInstance 不是虚拟方法，派生类不能重载它，但派生类可以重载 NewInstance，在重载 NewInstance 时要确保最后一句是 InitInstance。

NewInstance 类方法

声明：class Function NewInstance: TObject; virtual;