

计算机技术

入门

提高

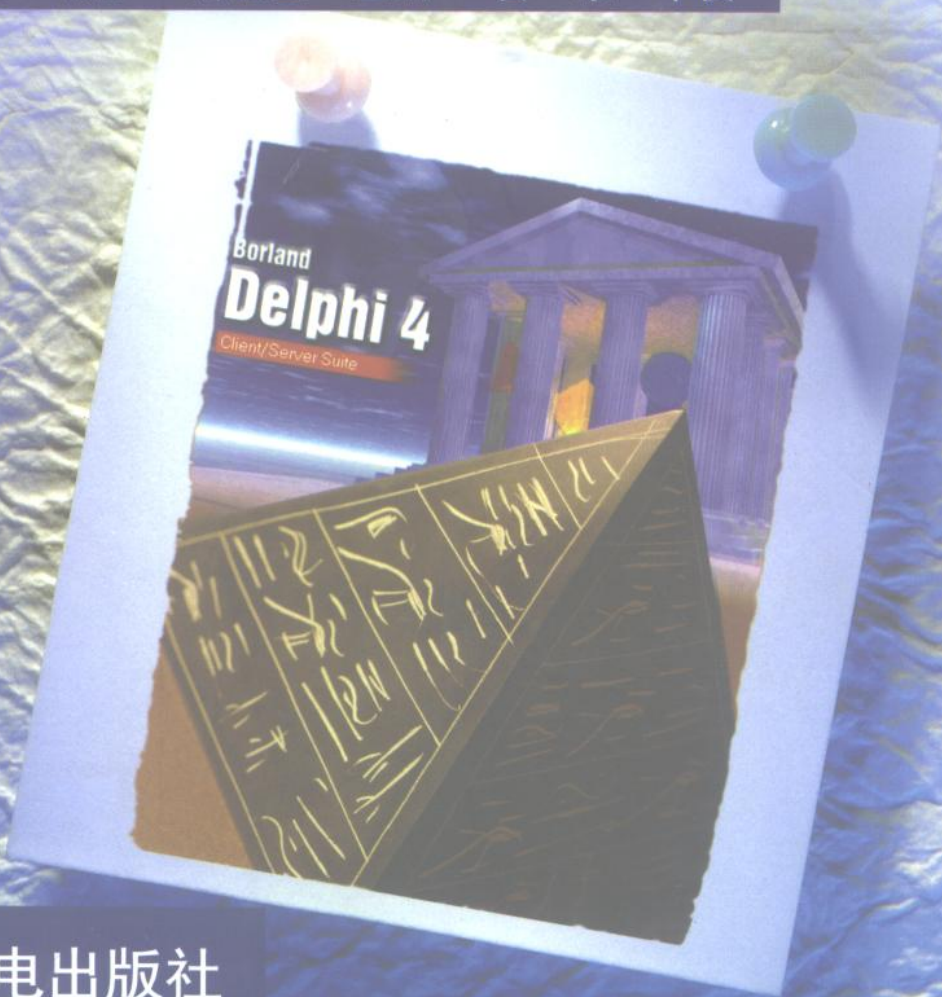
精通

系列丛书

Delphi

4 实用开发指南

康博创作室 编著 郝启堂 主编 曹 康 审校



人民邮电出版社

计算机技术入门提高精通系列丛书

Delphi 4 实用开发指南

康博创作室 编著

郝启堂 主编

曹 康 审校

人民邮电出版社

JS.66/27

计算机技术入门提高精通系列丛书

Delphi 4 实用开发指南

◆ 编 著 康博创作室

主 编 郝启堂

审 校 曹 康

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: 19.5

字数: 478 千字

1999 年 8 月第 1 版

印数: 1 - 5 000 册

1999 年 8 月北京第 1 次印刷

ISBN 7-115-07882-3/TP·1153

定价: 31.00 元

本书从 Delphi 的编程语言 Object Pascal 开始,向读者详细阐述了 Delphi 应用程序的编程元素,同时介绍了 Delphi 4 与其以前版本相比所具有的新增特性。根据当前 Windows 应用程序所具有的特性,本书就用户界面的构造、文件的操作、重要事件的响应、数据库以及 Internet 方面的应用进行了简明扼要的介绍,从而使本书既适用于想利用 Delphi 语言编写应用程序的初学者,也适用于想在短时间内就能编写出专业水准应用程序的高级用户。

前 言

Delphi 是一种可视化应用程序开发工具,其基础语言为 Object Pascal。由于 Object Pascal 是一种强类型语言,与其他语言相比,它提供了一种快速的编译器。优化编译模式在很大程度上提高了代码质量,所以这种语言一直都受到人们的青睐。自从 Delphi 1.0 推出以来,相继出现了 Delphi 的 2.0、3.0 和 4.0 版本。Delphi 4 与其以前的版本相比,虽然它们的集成开发环境看上去极为相似,但二者之间存在一些较大的差别。Delphi 4 在好几个方面都作了改进,主要体现在数据库体系结构与连接、VCL 组件增强、Object Pascal 语言的扩展、工程管理器、模块管理器等方面。这些改进使 Delphi 4 的功能更加强大,使用起来更加灵活和方便,大大地提高了应用程序的开发效率。

本书由浅入深、循序渐进地介绍了 Delphi 4 的语言基础和程序开发技巧。在内容编写和结构编排上既照顾到 Delphi 语言自发功能的内在逻辑,又充分考虑到国内用户现有的实际应用水平。各章内容既相对独立,又前后呼应,初学者可以按顺序阅读本书前几章的内容,以便快速掌握 Delphi 的编程基础和语法特征。中高级用户可以直接阅读自己感兴趣的编程议题。另外,为了使读者尽快利用 Delphi 这个强大的工具构建出专业水准的 Windows 应用程序,本书以丰富的例子,说明性很强的插图向读者演示了应用程序各组成部分的编程方法,提出了关键要素的处理技巧,使读者在短时间内即可达到相当不错的学习效果。

本书由康博创作室编著,郝启堂博士主编,曹康博士审校。由于时间仓促,加之作者水平有限,本书难免有疏漏之处,恳请广大读者指正。

康博创作室

1999 年 4 月

第一章 Delphi 4 简介	(1)
1.1 Delphi 4 的新特性	(1)
1.1.1 语言扩展	(1)
1.1.2 项目管理器(Project Manager)	(2)
1.1.3 用户界面的增强功能	(2)
1.1.4 AppBrowser	(3)
1.1.5 CORBA 支持	(3)
1.1.6 ActionLists	(3)
1.2 Delphi 4 的安装	(4)
1.2.1 Delphi 4 的版本	(4)
1.2.2 系统最低需求	(4)
1.2.3 安装步骤	(4)
1.3 开发环境简介	(8)
1.3.1 主窗口	(8)
1.3.2 对象编辑器	(10)
1.3.3 代码编辑器	(11)
1.3.4 窗体(Form)	(12)
第二章 编写第一个 Delphi 程序	(13)
2.1 Delphi 的编程特点	(13)
2.2 编写第一个程序	(13)
2.2.1 设计思路	(13)
2.2.2 程序设计	(14)
2.2.3 程序运行	(21)
2.2.4 程序的存储	(21)
2.3 工程简介	(22)
2.3.1 工程文件	(22)
2.3.2 窗体文件	(23)
2.3.3 单元文件	(23)
2.3.4 Delphi 生成的其它文件	(24)
2.4 工程管理	(25)
2.4.1 工程管理器简介	(25)
2.4.2 给工程增加文件	(26)
2.4.3 从工程中删除文件	(27)

2.5 工程的编译和运行	(27)
2.5.1 语法检查	(28)
2.5.2 编译链接	(28)
2.5.3 重新编译链接	(28)
2.5.4 工程的运行	(29)
2.6 工程环境设置	(29)
2.6.1 Forms 选项卡	(29)
2.6.2 Application 选项卡	(30)
2.6.3 Compiler 选项卡	(31)
2.6.4 Linker 选项卡	(33)
2.6.5 Directories/Conditionals 选项卡	(34)
2.6.6 VersionInfo 选项卡	(35)
2.6.7 Packages 选项卡	(36)
2.7 集成调试器	(37)
2.7.1 何时使用调试器	(37)
2.7.2 调试前的准备	(37)
2.7.3 准备调试用例	(38)
2.7.4 单步执行程序	(40)
2.7.5 设置断点	(40)
2.7.6 监视变量	(41)
第三章 Pascal 语言	(45)
3.1 Pascal 语言简介	(45)
3.2 数据类型	(45)
3.2.1 变量	(46)
3.2.2 常量	(46)
3.2.3 资源字符串常量	(48)
3.2.4 Delphi 的数据类型	(48)
3.2.5 用户定义的数据类型	(56)
3.2.6 指针类型	(60)
3.2.7 文件类型	(61)
3.2.8 Delphi 中的字符串	(61)
3.3 编程风格	(66)
3.3.1 注释	(66)
3.3.2 大写的使用	(67)
3.3.3 空白	(67)
3.3.4 美观的打印	(67)
3.3.5 语法的突出	(68)
3.4 Pascal 语句	(68)

3.4.1	表达式和运算符	(68)
3.4.2	简单语句和复合语句	(70)
3.4.3	条件语句	(70)
3.4.4	Pascal 中的循环	(71)
3.4.5	with 语句	(72)
3.5	过程和函数	(73)
3.5.1	引用参数	(74)
3.5.2	常量参数	(74)
3.5.3	不定数组参数	(75)
3.5.4	无类型不定数组参数	(75)
3.5.5	Delphi 调用约定	(76)
3.5.6	对象方法	(76)
3.5.7	Forward 声明	(76)
3.5.8	外部声明	(77)
3.5.9	过程类型	(77)
3.5.10	WINDOWS 回调函数	(78)
第四章	面向对象的语言 Object Pascal	(79)
4.1	类和对象	(79)
4.1.1	类的定义	(79)
4.1.2	创建 Delphi 的对象	(81)
4.1.3	在 Delphi 中 TMember 类的使用	(81)
4.1.4	构造函数的声明	(82)
4.2	类的封装性	(83)
4.2.1	专用、保护和公共部分	(84)
4.2.2	公共和 Published 部分	(84)
4.3	关于单元	(85)
4.3.1	类的接口部分	(86)
4.3.2	单元和作用域	(86)
4.3.3	单元和名称冲突	(87)
4.4	类的继承	(88)
4.4.1	继承和类型兼容	(88)
4.4.2	类的多态性	(90)
4.4.3	重载和重定义对象方法	(92)
4.4.4	动态方法	(93)
4.4.5	抽象方法	(93)
4.5	异常处理	(93)

第五章 高级 Object Pascal	(97)
5.1 Self 关键字	(97)
5.2 类方法与类的数据	(99)
5.3 对象方法指针	(102)
5.4 类的引用	(103)
第六章 键盘与鼠标的编程	(105)
6.1 Delphi 事件与消息	(105)
6.1.1 Delphi 事件	(105)
6.1.2 Delphi 消息	(106)
6.2 键盘的编程	(107)
6.2.1 虚拟键	(107)
6.2.2 键盘事件与响应	(107)
6.2.3 键盘事件预览	(109)
6.2.4 键盘编程例子	(109)
6.3 鼠标控制与响应	(111)
6.3.1 鼠标事件	(111)
6.3.2 Microsoft 智能鼠标	(112)
6.3.3 鼠标拖放	(113)
6.3.4 鼠标光标	(116)
6.4 直接处理事件	(116)
第七章 菜单的创建与处理	(119)
7.1 VCL 控件	(119)
7.2 菜单技术基础	(119)
7.2.1 创建菜单	(119)
7.2.2 菜单设计	(120)
7.2.3 图形化菜单设计	(123)
7.2.4 菜单模板	(123)
7.2.5 菜单事件响应与命令模拟技术	(124)
7.3 智能菜单技术与设计	(124)
7.3.1 菜单项灰化	(124)
7.3.2 动态菜单	(125)
7.3.3 多重菜单的合并与分离	(127)
第八章 按钮和复选框的添加	(129)
8.1 VCL 控件	(129)
8.2 基本按钮	(130)

8.2.1 命令按钮 —— Button	(130)
8.2.2 复选框 —— CheckBox	(131)
8.2.3 单选按钮 —— RadioButton	(131)
8.3 高级按钮	(131)
8.3.1 位图按钮 —— BitBtn	(131)
8.3.2 加速按钮 —— SpeedButton	(133)
8.3.3 增减按钮 —— UpDown	(133)
8.4 按钮组合技术	(134)
8.4.1 使用操作面板 —— Panel	(135)
8.4.2 使用组框 —— GroupBox	(136)
8.4.3 使用单选组框 —— RadioGroup	(136)
第九章 工具栏、状态栏与酷条设计	(139)
9.1 VCL 控件	(139)
9.2 设计工具栏	(140)
9.2.1 加入工具栏前的考虑	(140)
9.2.2 工具按钮图标的制作	(140)
9.2.3 加入工具栏	(142)
9.2.4 加入工具按钮	(142)
9.2.5 浮动工具栏	(143)
9.2.6 定制工具栏	(144)
9.3 状态栏设计	(146)
9.3.1 加入状态栏	(146)
9.3.2 设计状态栏	(147)
9.4 创建酷条	(148)
9.4.1 包容所有控件还是部分	(148)
9.4.2 创建酷条	(148)
9.4.3 加入其它控件	(149)
9.5 设计酷条	(150)
9.5.1 菜单栏	(150)
9.5.2 酷条上图案设计	(151)
9.5.3 酷条上的动画	(151)
9.5.4 停靠工具栏 —— Delphi 新特性	(151)
9.6 控制条	(152)
第十章 列表框设计	(155)
10.1 VCL 控件	(155)
10.2 字符串类 —— TString	(155)
10.3 列表控件剖析	(156)

10.3.1 字符串存储	(156)
10.3.2 表项选择	(157)
10.3.3 字符串其它操作	(157)
10.4 字符串网格设计	(159)
10.4.1 字符串网格剖析	(159)
10.4.2 设计实例	(161)
10.5 列表视图	(163)
10.5.1 列表视图的数据 —— TListItem	(163)
10.5.2 列表视图控件	(164)
10.5.3 列表视图的程序设计	(164)
10.6 树视图	(167)
10.6.1 树视图的内容 —— TTreeNode	(167)
10.6.2 树视图控件	(167)
10.6.3 树视图控件的程序设计	(168)
10.6.4 外部数据与拖放	(170)
第十一章 字符串与文本	(175)
11.1 VCL 控件	(175)
11.2 字符串类型	(175)
11.2.1 Object Pascal 字符串类型	(175)
11.2.2 字符串函数	(176)
11.2.3 PChar 字符串	(178)
11.2.4 Unicode 支持	(179)
11.3 单行文本处理	(180)
11.3.1 Label 与 StaticText	(180)
11.3.2 EditBox 与 MaskEdit	(181)
11.4 多行文本处理	(182)
11.4.1 创建 Memo 对象	(182)
11.4.2 文本输入	(183)
11.4.3 文本编辑与更新	(186)
第十二章 目录与文件	(187)
12.1 VCL 控件	(187)
12.2 公共文件对话框	(188)
12.3 目录与文件浏览	(190)
12.4 文件的拖放	(193)
12.4.1 Windows 的拖放	(193)
12.4.2 文件的拖放	(193)
12.4.3 程序实例	(195)

12.5 文件的读写	(196)
12.5.1 文件类型	(197)
12.5.2 基本函数	(198)
12.5.3 读写文件	(200)
12.5.4 VCL 对象的文件读写	(200)
第十三章 对话框	(203)
13.1 VCL 控件	(203)
13.2 对话框的模态	(204)
13.3 公共对话框	(204)
13.3.1 颜色与 ColorDialog	(205)
13.3.2 字体与 FontDialog	(206)
13.3.3 SearchDialog 与 ReplaceDialog	(207)
13.3.4 PrintDialog 与 PrinterSetupDialog	(208)
13.4 消息对话框与其它对话框	(210)
13.4.1 Delphi 消息对话框	(210)
13.4.2 Delphi 输入框	(212)
13.4.3 Windows 消息对话框	(213)
13.4.4 其它 Windows 对话框	(214)
13.5 对话框的一般设计过程	(215)
13.5.1 自述对话框的设计	(215)
13.5.2 动态创建对话框	(216)
13.6 多页对话框的设计	(217)
13.6.1 页面控制器 —— PageControl	(217)
13.6.2 TabControl 和 NoteBook	(219)
13.6.3 Win3.X 下运行	(220)
13.7 对话框的停靠技术	(221)
第十四章 MDI 应用程序	(223)
14.1 MDI 基础	(223)
14.2 Action 对象	(224)
14.3 MDI 应用程序设计	(226)
14.3.1 使用 Delphi 模板	(226)
14.3.2 手工创建	(227)
14.3.3 加入第二种类型子窗口	(227)
第十五章 图形与图像应用程序	(231)
15.1 Delphi 图形设计基础 —— Canvas	(231)
15.1.1 位置与像素	(231)

15.1.2	基本图元的绘制	(232)
15.1.3	控制图元属性	(233)
15.1.4	绘制文本	(235)
15.2	Delphi 图形设计	(235)
15.2.1	使用窗体	(235)
15.2.2	使用 PaintBox	(236)
15.2.3	使用 Shape	(236)
15.2.4	Delphi 控件表面图形的自绘制程序设计	(236)
15.2.5	创建 Canvas 对象	(237)
15.3	图像处理	(238)
15.3.1	图像格式与文件	(238)
15.3.2	TMetafile 与 TMetafileCanvas	(239)
15.4	利用 Chart 控件设计商业图形	(240)
15.5	3D 图形程序设计	(242)
15.5.1	OpenGL 简介	(243)
15.5.2	OpenGL 函数	(243)
15.5.3	OpenGL 程序设计	(243)
第十六章	打印机应用程序设计	(251)
16.1	Windows 打印技术	(251)
16.2	Delphi 打印编程	(251)
16.2.1	TPrinter 类	(252)
16.2.2	程序设计	(252)
16.2.3	窗体的打印	(256)
16.2.4	打印预览	(256)
16.3	文本的打印	(257)
第十七章	数据库编程	(259)
17.1	数据库控件	(259)
17.2	数据库应用程序初步	(260)
17.2.1	数据库基本术语	(260)
17.2.2	创建最小的数据库应用程序	(260)
17.2.3	使用 Database Form Wizard	(262)
17.2.4	数据模块	(265)
17.2.5	数据表的链接	(266)
17.3	数据库维护	(267)
17.3.1	浏览数据库	(267)
17.3.2	读写字段	(268)
17.3.3	查找记录	(269)

17.3.4	筛选记录	(269)
17.3.5	加入和删除记录	(270)
17.3.6	编辑记录	(271)
17.3.7	书签	(271)
17.4	SQL 查询	(272)
17.4.1	Query 控件	(272)
17.4.2	SQL 属性	(274)
17.4.3	参数化查询	(274)
第十八章	剪贴板、DDE 与 OLE	(277)
18.1	剪贴板	(277)
18.1.1	TClipboard 类	(278)
18.1.2	文本内容的复制与粘贴	(279)
18.1.3	图像的复制与粘贴	(280)
18.1.4	Delphi 对象与剪贴板	(281)
18.2	动态数据交换(DDE)	(283)
18.2.1	动态数据交换过程	(284)
18.2.2	VCL 控件	(284)
18.2.3	DDE 初步	(285)
18.2.4	DDE 服务器与客户机程序	(287)
18.3	对象链接与嵌入(OLE)	(291)
18.3.1	对象链接与嵌入	(292)
18.3.2	OLE 包容器类(TOleContainer)	(292)
18.3.3	OLE 包容器程序设计	(294)
18.3.4	OLE 自动化(OLE Automation)	(295)
18.3.5	OLE 自动化客户机程序举例	(296)

Delphi 4 简介

Borland 公司(现已更名为 Inprise)被誉为“开发工具专家”,它推出的 Delphi 自 1994 年面世以来一直备受称道,其简便的操作、强劲的功能和高效率为它赢得越来越多的用户。Delphi 4 是它的最新杰作。在第一章中我们主要介绍 Delphi 4 的新特性、安装步骤和开发环境。

1.1 Delphi 4 的新特性

1.1.1 语言扩展

Delphi 使用 Object Pascal —— 一种真正面向对象的程序开发语言,可以方便地实现面向对象的开发。与以前的版本相比,Delphi 4 提供了许多新的语言特性,使开发人员有了更多的选择,可以更方便地实现多种功能。

一、动态数组(Dynamic arrays)

与传统的数组概念不同,动态数组使开发人员在使用数组时只需指定数据类型和维数,而不必指定元素数,在运行时定义数组长度或动态地改变它。例如,可以用如下的定义:

```
var A: Array of Integer;
```

然后在程序中用 `SetLength` 指定它的长度。下面语句将数组 A 的长度设为 10 个元素。

```
Begin  
    ...  
    SetLength(A, 10);  
    ...  
End.
```

二、方法重载(Method overloading)

方法重载允许对象有多个同名的方法,这些方法由它们所带的不同参数区分。例如,可以有下面的方法:

```
function hoo(I: Integer): String; Overload;  
function hoo(S: String): String; Overload;
```

这两个同名的函数有不同的处理过程。当调用时,编译器会根据传入参数的不同而调用不同的方法。这使得开发人员可以编写更灵活、更易于维护的程序,特别是在建立那些使用第三方厂商提供的企业对象或组件时显得尤为重要。实际上,方法重载是一种类似C++语言的特性,它的根本目的是为了使得 Delphi、C++ Builder 和 Jbuilder 的程序开发更容易相互组合。

三、默认参数(Default parameters)

Delphi 4 允许在定义方法的同时指定参数的默认值,请看下面的例子:

```
Procedure Sam(A: integer = 3);  
Begin  
ShowMessage( IntToStr(A) );  
End;
```

如果在程序中用无参数的形式调用 Sam:

```
Begin  
...  
Sam;  
...  
End.
```

则默认参数值“3”会被引用,注意这并不是允许改变参数数目,Sam 在任何时候都要接受一个参数,本例中使用的是默认值。

四、64 位整数

Delphi 4 对 64 位整数的支持提供了更强的处理精确计算的能力,并支持更大范围的数值,在开发科学计算、工程应用,尤其是金融处理时这是非常重要的。

1.1.2 项目管理器(Project Manager)

Delphi 4 全新的项目管理器(见图 1-1)可以同时管理多个项目,可以从同一源代码中在同一时间生成组件、动态链接和可执行文件,而且版本号是绝不会重复的。开发者可以更多地应用开发、更少地进行项目管理,从而在应用程序体积膨胀的今天获取更多的主动。

1.1.3 用户界面的增强功能

Delphi 的用户界面一直非常友好,适合开发人员快速高效地进行应用开发。在 Delphi 4 中,又新增加了对 Docking(坞式)窗口与窗体的支持,使得代码窗口、窗体和调试窗口可方便地重新分组、排列,从而在项目开发的各个阶段选择更合适的桌面。

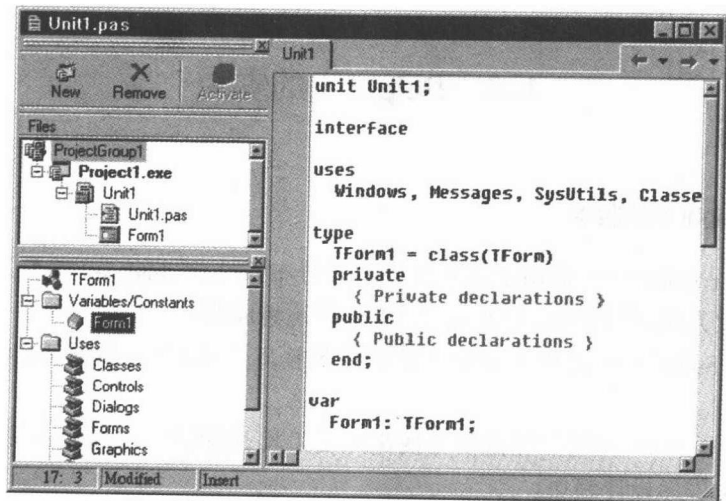


图 1-1 全新的项目管理器可同时管理多个项目

1.1.4 AppBrowser

AppBrowser 是 Inprise 开发实验室的最新成果,它采用先进的后台分割来理解各个部分代码(如方法调用、定义和执行)的内在关系,然后只需要单击鼠标按键就可以找到所需的源代码。

1.1.5 CORBA 支持

Delphi 4 支持一步到位的开发 COM 和 CORBA 组件,可以让一个组件支持 COM 和 CORBA 两种标准。

Delphi 4 Client/Server Suite 是第一个支持 OMG(Object Management Group)组织的 CORBA 分布式对象的快速开发工具。它内含了 VisiBroker CORBA ORB 并提供可视化编程工具,使开发人员不需要学习撰写 CORBA IDL,即能快速生成可供异构平台环境的 C++、Java、Smalltalk、COBOL 等程序语言使用的 CORBA 应用对象。

1.1.6 ActionLists

ActionLists 是继可视化开发工具诞生以来在用户界面开发方面最有意义的概念,它使各种用户界面元件可以在不同对象中实现统一功能。想象一下,如果你在一个应用程序的开发中有几个不同的方式实现一个操作,比方说,“保存文件”的功能,可以从菜单中执行,也可以在快捷工具栏中执行,或者还可以从别的什么情况下被调用,然而当没有文件生成或调入时,保存功能是不需要的。用户需要将它们“变灰”(使按钮失效),传统的思路是编写代码处理所有涉及到的元件。当调入文件时,又要做相反的事情,使它们一一激活。如果是在更复杂的情况下,许多元件关联时,那才是真正令人头疼。Delphi 4 的解决方案是将所有这些元件都连接到 ActionList 的一个列表项上,这样在相应情况下只需对 ActionList 的列表项作修改即可完成任务。