

Delphi(1.0~2.0)进阶指南

喻新文 史宝军 主编

電子工業出版社.

内 容 提 要

Borland 公司开发的 Delphi 是 Windows 环境下的可视化开发工具,能够快速生成程序代码和程序原型。该软件以优异的特性于 1995 年被 IDG、PC Magazine 等多家权威机构评为最佳 PC 程序开发工具。

本书介绍了 Delphi 1.0~2.0 不同版本的使用方法和技巧。其中第一至七章主要讲述 Delphi 的入门知识,包括窗体、对话框的基本设计方法;第八至十三章主要讨论 Delphi 的深层次技术,包括 DDE、数据库程序的设计等;第十四章介绍 Delphi 2.0 版的最新技术。

本书由浅入深、注重实用、结构严谨、文字流畅,可作为该软件的培训教科书及自学材料,亦可作为广大计算机工作者的参考书。

Delphi(1.0~2.0)进阶指南

喻新文 史宝军 主编

责任编辑:郭 立

*

电子工业出版社出版

北京市海淀区万寿路 173 信箱(100036)

电子工业出版社发行 各地新华书店经销

北京大中印刷厂印刷

*

开本:787×1092 毫米 1/16 印张:16.5 字数:422 千字

1996 年 10 月第一版 1996 年 10 月北京第一次印刷

印数:5000 册 定价:20.00 元

ISBN 7-5053-3689-4/TP·1547

目 录

第一章 Delphi 基础	(1)
1.1 什么是 Delphi	(1)
1.2 Delphi 的适用领域	(1)
1.2.1 Delphi 的用户范围	(1)
1.2.2 Delphi 支持的技术范围	(2)
1.2.3 可视化开发工具 Visual Basic 与 Delphi 的比较	(2)
1.3 基于 Pascal 语言的 Delphi	(3)
1.4 Delphi 的标准版和客户/服务器版比较	(3)
1.5 Delphi 对系统和应用程序开发人员的要求	(4)
1.5.1 Delphi 的系统需求	(4)
1.5.2 Delphi 对应用程序开发人员的需求	(4)
1.6 Delphi 的安装	(5)
第二章 Delphi 界面——Delphi 环境概述	(6)
2.1 用户屏幕窗口	(6)
2.1.1 标题栏(Title)	(6)
2.1.2 菜单(Menu)	(6)
2.1.3 加速条(SpeedBar)	(6)
2.1.4 元件模板(Component Palette)	(8)
2.1.5 窗体(Form)	(8)
2.1.6 对象检测器(Object Inspector)	(8)
2.1.7 代码编辑窗口(Code Edit Windows)	(9)
2.2 菜单系统	(9)
2.2.1 文件(File)菜单	(9)
2.2.2 编辑(Edit)菜单	(10)
2.2.3 搜索(Search)菜单	(11)
2.2.4 察看(View)菜单	(11)
2.2.5 选项(Options)菜单	(12)
2.2.6 工具(Tools)菜单	(12)
2.2.7 运行(Run)菜单	(13)
2.2.8 帮助(Help)菜单	(13)
2.3 其他元素	(13)
2.3.1 菜单设计工具(Menu Designer)	(13)
2.3.2 集成调试器(Integrated Debugger)	(13)
2.3.3 对象浏览器(Object Browser)	(14)
2.3.4 影像编辑器(Image Editor)	(14)
2.3.5 项目管理器(Project Manager)	(14)

第三章 Delphi 的基本元素——元件	(15)
3.1 Delphi 的元件	(15)
3.1.1 Standard 页次元件	(16)
3.1.2 Additional 页次元件	(17)
3.1.3 Data Access 页次元件	(17)
3.1.4 Data Controls 页次元件	(18)
3.1.5 Dialogs 页次元件	(19)
3.1.6 System 页次元件	(19)
3.1.7 VBX 页次元件	(20)
3.1.8 Samples 页次元件	(20)
3.2 自定义 Delphi 可视化元件库	(21)
3.3 利用元件专家(Component Expert)来产生元件	(23)
3.4 方法	(24)
3.5 常用 Delphi 元件	(24)
3.5.1 TEdit 元件的使用	(25)
3.5.2 TMemo 元件的使用	(26)
3.5.3 TCheckBox 元件的使用	(27)
3.5.4 TRadioButton 元件的使用	(28)
3.5.5 TListBox 元件的使用	(28)
3.5.6 TComboBox 元件的使用	(30)
3.5.7 TGroupBox 元件的使用	(31)
3.5.8 TRadioGroup 元件的使用	(31)
第四章 窗体设计	(32)
4.1 对象检测器	(32)
4.2 窗体	(33)
4.2.1 窗体属性	(33)
4.2.2 窗体事件	(35)
4.3 窗体中的元件操作	(36)
4.3.1 向窗体中加入元件	(36)
4.3.2 选择窗体中的元件	(36)
4.3.3 改变窗体中元件的大小	(37)
4.3.4 在窗体中加入一个元件的多个拷贝	(37)
4.3.5 组编元件的操作	(38)
4.3.6 剪贴、复制和粘贴元件	(38)
4.3.7 删除和恢复元件	(39)
4.3.8 调整元件位置	(39)
4.3.9 锁定元件位置	(40)
4.4 设置元件属性	(41)
4.4.1 显示及设定共同属性值	(41)
4.4.2 设定元件的 Tab 次序	(41)
4.4.3 激活或屏蔽元件	(42)
4.4.4 设定窗体中的焦点元件	(42)

4.5	窗体的重用	(42)
4.6	窗体模板	(43)
4.6.1	使用窗体模板	(43)
4.6.2	将一个窗体加入窗体模板中	(43)
4.7	将窗体以 ASCII 码方式存储	(44)
4.8	菜单设计	(45)
4.8.1	菜单设计过程	(45)
4.8.2	增加和删除菜单项	(45)
4.8.3	加入分隔列	(46)
4.8.4	指定加速键和快捷键	(46)
4.8.5	重新安排菜单项的位置	(47)
4.8.6	产生多层菜单	(47)
4.8.7	利用菜单模板进行菜单设计	(47)
4.8.8	将菜单存成模板	(48)
4.8.9	连接菜单事件和程序	(48)
第五章	Object Pascal 程序语言概述	(50)
5.1	编制具有自己风格的程序	(50)
5.2	赋值语句和属性设置	(50)
5.3	变量	(52)
5.4	简单数据类型	(53)
5.5	常数	(54)
5.6	判断语句	(55)
5.6.1	if 语句	(55)
5.6.2	Case 语句	(56)
5.7	循环语句	(57)
5.7.1	While 语句	(57)
5.7.2	Repeat 语句	(57)
5.7.3	For 语句	(57)
5.7.4	嵌套循环	(58)
5.8	复杂数据类型	(59)
5.8.1	枚举型	(59)
5.8.2	子界型	(60)
5.8.3	字符串型	(60)
5.8.4	数组	(61)
5.8.5	记录类型	(62)
5.8.6	集合类型	(63)
5.8.7	文件类型	(64)
5.9	Delphi 程序单元	(64)
5.10	过程和函数	(66)
5.10.1	过程	(66)
5.10.2	参数传递	(67)
5.10.3	函数	(68)
5.10.4	嵌套调用	(68)

5.11 变量和常量的应用范围	(69)
第六章 编制 Delphi 程序	(70)
6.1 Delphi 应用程序项目文件设计	(71)
6.2 窗体属性的设计	(71)
6.3 窗体中系统菜单的设计	(73)
6.4 窗体中快速按钮的设计	(80)
6.5 窗体中状态行的设计	(86)
6.6 多页面窗体的设计	(90)
6.7 多窗体 Delphi 应用程序的设计	(97)
6.7.1 调用方法	(97)
6.7.2 设计示例	(97)
第七章 对话框的使用方法	(101)
7.1 对话框函数	(101)
7.1.1 普通信息显示对话框	(101)
7.1.2 产生简单的输入窗体	(103)
7.1.3 遮盖输入字符串对话框	(104)
7.2 开发自定义对话框	(105)
7.2.1 设置对话框的窗体特性	(106)
7.2.2 在对话框中加入命令按钮	(107)
7.2.3 按钮元件的特性设置	(107)
7.2.4 标准的命令按钮	(108)
7.3 对话框元件的使用	(109)
7.3.1 打开文件对话框	(109)
7.3.2 文件存储对话框	(111)
7.3.3 字体设置对话框	(111)
7.3.4 颜色设置对话框	(112)
7.3.5 打印对话框	(113)
7.3.6 打印设置对话框	(114)
7.3.7 查找文本对话框	(114)
7.3.8 替代文本对话框	(115)
第八章 Delphi 中 DDE 的使用	(121)
8.1 DDE 的几个概念	(121)
8.1.1 DDE 的会话、用户端、服务端	(121)
8.1.2 DDE 的应用程序名称、主题和项目	(121)
8.2 DDE 主要元件介绍	(122)
8.2.1 TDDEServerConv 元件	(122)
8.2.2 TDDEServerItem 元件	(122)
8.2.3 TDDEClientConv 元件	(123)
8.2.4 TDDEClientItem 元件	(123)
8.3 DDE 用户端应用程序的设计	(124)
8.3.1 在设计阶段建立 DDE 链接	(124)

8.3.2	在程序运行阶段建立 DDE 链接	(124)
8.3.3	在 DDE 数据交换中从服务端直接取得数据	(125)
8.4	DDE 服务端应用程序的设计	(125)
8.5	DDE 设计示例	(125)
8.5.1	DDE 服务端应用程序设计示例	(126)
8.5.2	DDE 用户端应用程序设计示例	(128)
第九章	Delphi 应用程序的文件输入和文件输出	(131)
9.1	Delphi 中常用文件操作函数的功能及其用法	(131)
9.1.1	I/O 文件操作函数的功能及其用法	(131)
9.1.2	文件管理函数的功能及其用法	(132)
9.2	Delphi 文本文件的操作	(134)
9.2.1	Delphi 中常用的文本文件操作函数	(134)
9.2.2	Delphi 文本文件函数编程示例	(135)
9.2.3	字符串列表与文本文件的操作	(136)
9.3	Delphi 文件系统元件的使用	(136)
9.3.1	文件系统元件功能简介	(137)
9.3.2	文件系统元件的使用示例	(137)
9.4	文件输入/输出操作综合示例	(142)
第十章	Delphi 数据库程序的设计	(152)
10.1	Delphi 数据库访问机理	(152)
10.1.1	BDE 的安装与运行	(153)
10.1.2	建立数据库与 BDE 之间的联系	(153)
10.1.3	BDE 别名的设置及 BDE Config 的使用	(154)
10.2	Delphi 内建数据库元件介绍	(159)
10.2.1	TTable	(159)
10.2.2	TQuery	(160)
10.2.3	TDataSource	(161)
10.2.4	TDBGrid	(161)
10.2.5	TDBNavigator	(161)
10.2.6	TDBText	(162)
10.2.7	TDBEdit	(162)
10.2.8	TDBMemo	(162)
10.2.9	TDBImage	(162)
10.2.10	TDBCCheckBox	(163)
10.2.11	TDBLookupList	(163)
10.2.12	TDBLookupCombo	(163)
10.3	局域数据库应用程序的设计	(164)
10.3.1	单表格局域数据库应用程序的设计	(164)
10.3.2	多表格局域数据库应用程序的设计	(165)
10.3.3	DataSet Designer 的使用	(166)
10.3.4	Field Link Designer 的使用	(168)
10.3.5	局域数据库应用程序设计示例	(168)

10.4 SQL 应用程序的设计	(187)
10.4.1 SQL 查询概述	(188)
10.4.2 在程序设计阶段静态使用 SQL 查询	(188)
10.4.3 在程序运行阶段动态使用 SQL 查询	(189)
10.4.4 SQL 查询示例	(190)
第十一章 Delphi 中 ReportSmith 的使用	(195)
11.1 链接数据源	(195)
11.1.1 ReportSmith 支持的数据源及链接类型	(195)
11.1.2 链接数据库	(198)
11.2 设计报表的一般过程	(199)
11.2.1 选择报表类型	(199)
11.2.2 增加数据库	(199)
11.2.3 链接数据库	(201)
11.2.4 选择数据库字段	(202)
11.2.5 数据库排序	(203)
11.2.6 数据库分组	(204)
11.2.7 生成派生字段	(204)
11.2.8 使用报表变量	(205)
11.3 格式化报表	(207)
11.3.1 给报表增加背景、设置纸的大小	(207)
11.3.2 给报表中的项目增加边框	(209)
11.3.3 图形显示数据字段	(210)
11.3.4 使用绘图工具	(210)
11.3.5 输入并格式化文本	(211)
11.3.6 格式化字段	(211)
11.3.7 插入字段标签、字段、报表变量	(213)
11.3.8 报表分页	(215)
11.3.9 存储报表风格	(215)
11.4 打印报表	(216)
11.4.1 图形预览	(216)
11.4.2 简明预览方式	(217)
11.4.3 打印报表	(217)
11.5 建立交叉表	(217)
11.5.1 交叉表简介	(217)
11.5.2 在 ReportSmith 中生成交叉表的方法	(218)
11.5.3 交叉表工具框介绍	(220)
11.6 ReportSmith 与其他 Windows 应用程序之间的相互调用	(221)
11.6.1 从 ReportSmith 中存取 dBASE for Windows 数据	(221)
11.6.2 将 ReportSmith 变成 dBASE for Windows 的报表生成器	(221)
11.6.3 使用动态数据交换	(221)
11.6.4 使用 OLE 2.0	(224)
第十二章 程序调试和出错处理	(226)

12.1	调试	(226)
12.1.1	调试工具	(226)
12.1.2	错误类型	(230)
12.2	出错处理	(231)
12.2.1	确定被保护的程序	(231)
12.2.2	建立保护区程序	(231)
12.2.3	RTL 情况处理	(232)
12.2.4	元件异常处理	(234)
12.2.5	异常类型的定义	(235)
第十三章	Delphi 项目管理器的应用	(236)
13.1	Delphi 的文件类型	(236)
13.2	项目选项设置	(237)
13.2.1	Gallery 方框	(237)
13.2.2	环境(Environment)选项	(238)
13.2.3	项目(Project)选项	(238)
13.3	项目管理器(Project Manager)	(240)
13.4	编译和运行项目	(241)
13.4.1	检查原始程序的语法	(241)
13.4.2	编译项目	(241)
13.4.3	获得编译状态信息	(242)
13.5	建立和运行一个项目	(242)
第十四章	从 Delphi 1.0 到 Delphi 2.0	(244)
14.1	Delphi 2.0 新增加的数据类型	(244)
14.2	Delphi 2.0 新增加的元件简介	(245)
14.3	Delphi 1.0 应用程序如何升级到 Delphi 2.0	(246)
14.4	Delphi 1.0 集成应用环境与 Delphi 2.0 集成应用环境的比较	(247)
14.5	Delphi 2.0 的安装要求	(248)
14.6	Delphi 2.0 新增加的其他内容	(249)

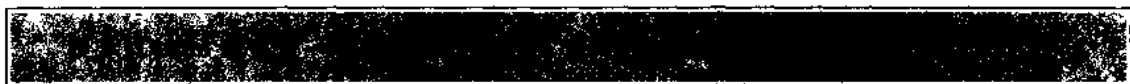
第一章 Delphi 基础

1.1 什么是 Delphi

Delphi 是一个可视化、面向对象的快速应用程序开发(RAD: Rapid Application Development)工具。它于 1994 年在 Borland 公司的国际会议上首次展示 Windows 平台的版本。1995 年推出基于 Windows 95 和 Windows NT 3.51 的 2.0 版本。其 1.0 版本又分为标准版和客户/服务器(Client/Server)版。由于其数据库技术支持 ODBC 接口,采用 BDE 引擎(Borland Database Engine),首次结合了具有高效率的原代码(Native Code)编译器、RAD 环境和可伸缩技术,因此用 Delphi 可以编写大型数据库程序。Delphi 是目前在 PC 机上最优异开发软件之一,1995 在美国软件评比中获得开发工具类第一名,领先 Visual Basic 4.0 版。

Delphi 的 2.0 版本是目前 Windows 95 上为数不多的 32 位开发语言。每一个熟悉 C 语言的读者都知道,C 语言由于最大限度地利用了 DOS 资源而成为 90 年代初全世界程序设计师的主要设计语言。同样,像 Delphi、Visual Basic 这类面向对象技术、充分发挥 Windows 平台资源的开发工具同样会成为以后微机平台上最合适的开发工具。

Delphi 发展至今已有 1.0 版和 2.0 版两种产品,由于两者之间的设计原则和操作方法的共性很多,本书的前十三章着重介绍 Delphi 1.0 版的使用,其中的多数技巧适于 Delphi 2.0 版,第十四章专门介绍 Delphi 2.0 版。



1.2 Delphi 的适用领域

1.2.1 Delphi 的用户范围

Delphi 适合于 Windows 环境下的程序开发人员,包括从程序设计的初学者到大型企业的程序设计人员以及专业开发人员。由于 Delphi 1.0 的程序码与 16 位或 32 位操作系统环境无关,所以利用 Delphi 1.0 for Windows 所开发的程序可直接在 Windows 95 上执行。而 Delphi 2.0 是直接为 Windows 95 和 Windows NT 3.51 开发的。

C++ 的开发者可以考虑使用 Delphi。专业的开发人员通常会依据手上的工作和任务来

选择合适的设计工具, Delphi 正好能满足一个多工具(multi-tool)环境,因为它具有完整的基于元件(component-based)设计与可视化的开发环境,可以让程序设计人员很快产生程序原型及用户界面。而在与 C++ 应用程序配合上, Delphi 的应用程序可调用 C++ 所产生的动态链接库(DLL: Dynamic Link Library), 同样 C++ 的应用程序也可调用 Delphi 所产生的 DLL。通过这样的交互使用, 开发者可以依据手上的工作, 选择最合适的工具来完成工作, 无需担心重复开发程序而浪费时间, 可以大大提高开发效率。

1.2.2 Delphi 支持的技术范围

因为 Delphi、dBASE 和 Paradox 都建立在 Borland 数据库引擎(BDE: Borland Database Engine)上, 它提供了直接存取 dBASE 或 Paradox 数据的能力。在网络上, Delphi 的应用程序可和 dBASE 与 Paradox 一起存取相同的数据库, 还可以通过 ODBC 接口和其他各种类型的数据库进行数据交换。开发人员也可以通过 Delphi 产生可重用(Reuse)的元件, 利用动态链接库 DLL 让 dBASE 和 Paradox 来调用。

Borland 将一些通用的开发技术加入到自己的开发工具中, 用以支持系统软件新增的各种功能, 其中包括 OLE2.0、DDE、DDL 和 VBX 等。在目前的 Delphi 中融入了基于对象(Object-based)的处理结构。

1.2.3 可视化开发工具 Visual Basic 与 Delphi 的比较

可视化编程语言是继传统的开发工具后的新一代工具。有了可视化编程工具, 编制 Windows 应用软件就不再限于专业开发人员了。可视化编程语言将 Windows 编程的复杂性隐藏起来, 向软件开发人员提供了用于构造 Windows 各个组成部分的元件, 以及好学易用的编程环境、调试器、源程序编辑器等各种工具, 使得用户不必了解 Windows 的工作原理及其属性, 不必知道事件驱动、消息循环等使用传统开发工具所必须了解的各类 Windows 编程知识, 就能够在很短的时间内编写出自己的 Windows 应用软件。

Microsoft 公司的 Visual Basic for Windows 是最早出现在软件市场上的此类编程系统, 它具有图形设计工具、结构化的事件驱动编程模式、开放的环境, 可使用户既快又简便地编制出 Windows 下的各种程序。使用 Visual Basic 编制 Windows 下的应用软件只需经过三个步骤:

- (1) 利用提供的各种元件绘制屏幕窗体(Form);
- (2) 为窗体上的元件(如按钮、文本输入框、菜单等)设置属性(如字体、颜色等);
- (3) 对各元件在一定事件下要执行的动作进行编程。

Visual Basic 的优点在于它将屏幕表格上的各个元件与相应的行为、动作结合起来, 并将各元件的属性与它们所收到的各种信息的反应分离开来, 使程序编制大为简便。因此 Visual Basic 是以往想开发可视化应用程序的一种较好选择, 但是长期以来存在以下问题:

- (1) 开发出的程序必须配合执行期的动态链接库(DLL), 程序变得庞大且执行速度很慢;
- (2) 只适合于个人开发, 无法开发中大型的程序(Basic 程序语言的限制);
- (3) 没有扩充性(无法产生 DLL、VBX 或 OLE2.0 Server), 无重用性的查询控制(Query Control), 没有表控制(Table Control)等。

Borland 公司推出的 Delphi for Windows 开发工具不但拥有和 Visual Basic 相似的开发环

境和程序设计步骤,并且所提供的元件更丰富,元件的分组组织更合理。Delphi 不仅能生成极佳的可执行程序,而且还能够编译生成 DLL 动态链接库。与 Visual Basic 相比,Delphi 的编译效率更高,编译出的可执行代码质量更好,程序设计人员使用 Delphi 可以快速开发出符合各种要求的各类 Windows 应用软件、设备驱动程序以及 DLL 和 VBX 可装载模块。

Delphi 是一个灵活多变的多功能平台,它也可以很好地支持 ODBC 以及 Client/Server 模式,能够方便地操纵本地以及大型分布式网络(如 SQL Server、Oracle、Informix 等)上的各类数据库文件。

Delphi 将软件开发中的面向对象技术与可视化编程有机地结合起来,集中到 Pascal 语言开发平台之中,使得在应用软件间能够更容易地实现代码的重复利用。可以说,Visual Basic 令人头痛的缺点正是 Delphi 被大家所赞赏的杰出特点,Delphi 的出现使应用软件的开发又向前迈进了一步。

1.3 基于 Pascal 语言的 Delphi

Delphi 所使用的程序语言为 Object Pascal。Pascal 语言具有诸多优点,它是一个结构化的面向对象的编译(Compiled)语言,正好符合 Borland 的高执行效率、可重用和易维护性的特性。它还有增强异常处理能力、增强类别(Class)的封装能力和特性以及自动的运行期间类型信息(RTTI;run-time type information)。Pascal 语言本身固有的优点,使得 Delphi 更加具有生命力和竞争力。

有些程序开发人员可能因为没有学过 Pascal,而对使用 Delphi 这一工具产生了顾虑。其实,这种担忧是大可不必的。因为 Delphi 的程序开发概念是注重在可视化开发、元件的重用性上, Object Pascal 并不是整个程序开发的主要部分,并且 Object Pascal 是一个结构化和面向对象的程序语言。如果您已有其他程序语言(如 C 语言等)的概念,相信可以很快适应这一语言环境。

1.4 Delphi 的标准版和客户/服务器版比较

Delphi 1.0 有两个产品:Delphi Desktop(标准版)和 Client/Server(客户/服务器版)。基本上两个产品均包含有:

- 高效率的原代码(native code)编译器
- 可重用与可扩充的元件结构
- 可视化的用户界面建立工具提供了双向工具(two-way tools)能力
- 可视化元件库(VCL: Visual Component Library)
- Borland Database Engine
- Local InterBase Server
- Report Smith 报表产生工具

- 支持微软系统技术规格,包括 OLE2.0、DDE、VBX 及 ODBC

而 Delphi Client/Server 主要针对大型企业、系统集成、顾问、ISV、VAR 等,是在开发高效率的工作组或企业 Client/Server 结构应用程序的需求中产生的。它除了具有 Delphi for Windows 的基本功能外,还提供:

- 对 SQL Links 驱动程序无限制的使用权 (SQL Links 是连接 Oracle、Sybase、Microsoft SQL Server、Informix、InterBase 的驱动程序)
- 可视化的查询工具 (Visual Query Builder)
- Team Development Support 开发集团式的程序控制与管理 (配合 Intersolv PVCS5.1)
- ReportSmith SQL 版本
- VCL 的源程序
- Local InterBase 开发工具,具有无限制的 Local InterBase 分发权限

本书将只介绍 Delphi 的标准版的使用方法与技巧, Delphi Client/Server 所涉及的内容请参考其他书籍。

1.5 Delphi 对系统和应用程序开发人员的要求

1.5.1 Delphi 的系统需求

Delphi 1.0 要求计算机系统必须具有:

- 80386 以上的 PC
- Microsoft Windows3.1 以上 100%兼容环境
- 8MB 或 8MB 以上扩展内存
- 30MB 以上的硬盘空间

由于现阶段中国市场上的 Delphi 软件主要为 CD-ROM 的包装,因此系统必须具有 CD-ROM 驱动器,或者利用带光盘的计算机将光盘上的 Delphi 拷贝到软盘上再安装。也可以向经销商直接购买软盘包装的软件。Delphi 2.0 对系统的需求及安装方法请参见十四章。

1.5.2 Delphi 对应用程序开发人员的要求

Delphi 是 Windows 环境下的开发工具,因此用户应对图形化的用户界面 (Graphical User Interface, GUI) Windows 较为熟悉。阅读本书的前提是您对 Windows 的基本操作方法 (例如使用鼠标选取元件,移动改变窗口的大小等) 已经非常清楚。

前面已经提到, Delphi 所使用的语言是 Object Pascal, 程序开发人员需对各元件在一定事件下要执行的动作进行编程。虽然 Pascal 不是整个程序开发的主要部分,但是开发人员在准备使用 Delphi 之前,应该先掌握 Pascal 的基本语言结构。本书假定读者已经有了初步的 Pascal 编程经验,因而将不对 Pascal 语言本身做过多介绍。

1.6 Delphi 的安装

Delphi 是基于 Windows 的软件，所以您必须在安装 Delphi 之前启动 Windows。Delphi 的安装程序会产生安装所需的目录并从光盘或磁盘中复制所需的文件。安装 Delphi 软件的步骤如下：

- (1) 首先启动 Windows；
- (2) 将装有 Delphi 的光盘放入您的光盘驱动器中；
- (3) 利用程序管理器（Program Manager）的 File | Run 菜单命令或文件管理器（File Manager）从 Delphi 的光盘中执行 \INSTALL\SETUP.EXE；
- (4) 接着按照安装程序的指示来安装。

README.TXT 文件中包含了很多关于 Delphi 的有用信息，建议您在使用 Delphi 之前认真地阅读。

第二章 Delphi 界面——Delphi 环境概述

对于初次使用 Delphi 的读者来说,这一章将有助您了解 Delphi 的窗口环境以及许多常用概念。Delphi 本身就是一个设计优秀的 Windows 应用软件,它的许多使用方法同其他 Windows 应用软件一样。

本章将向您详细介绍进入 Delphi 时所看到的主屏幕的基本元素:标题栏(Title)、菜单(Menu)、加速条(SpeedBar)、元件模板(Component Palette)、对象检测器(Object Inspector)、窗体(Form)窗口(可理解为用户设计窗口)及代码编辑窗口(经常被窗体窗口掩盖一部分)。我们还将向读者介绍在屏幕中看不见但经常被使用的元素:项目管理器、菜单设计工具、集成调试器、影象编辑器等。

2.1 用户屏幕窗口

当我们在 Windows 程序管理器的 Delphi 群组中双击 Delphi 图标时,将看到一个指示 Delphi 版本的图象,然后,看到如图 2.1 所示的图形,这也是 Delphi 进行程序设计时的界面。

下面向您介绍进入 Delphi 时所看到的主屏幕的基本元素。

2.1.1 标题栏(Title)

和所有的 Windows 应用软件一样,在屏幕的上方有一个水平方向的横条,它告诉您正在编辑或准备编辑的应用软件的名称。当您每次进入 Delphi 时,Delphi 都给您提供一个默认的项目名称 Project1,Delphi 的标题栏即显示 Delphi-Project1。

2.1.2 菜单(Menu)

Delphi 菜单为设计者提供了进行文件操作、文本编辑、查询、调试、编译、运行、各种选项设置、工具以及帮助等菜单项。本章后半部分将向读者详细介绍每一个菜单选项的功能。

2.1.3 加速条(SpeedBar)

加速条位于屏幕左上方,菜单条的下面。加速条用于放置加速按钮,加速按钮的功能是让您不经过菜单来快速选择任务,每一个加速按钮都有与它等效的键,在菜单里都可以找到相对应的项。读者在后面就可以发现,对同一个操作,Delphi 提供了多种方法,只不过选择哪一种方法是根据您的个人的偏好。将鼠标移到任一个加速按钮图标且停留时,就弹出一个提示按钮功

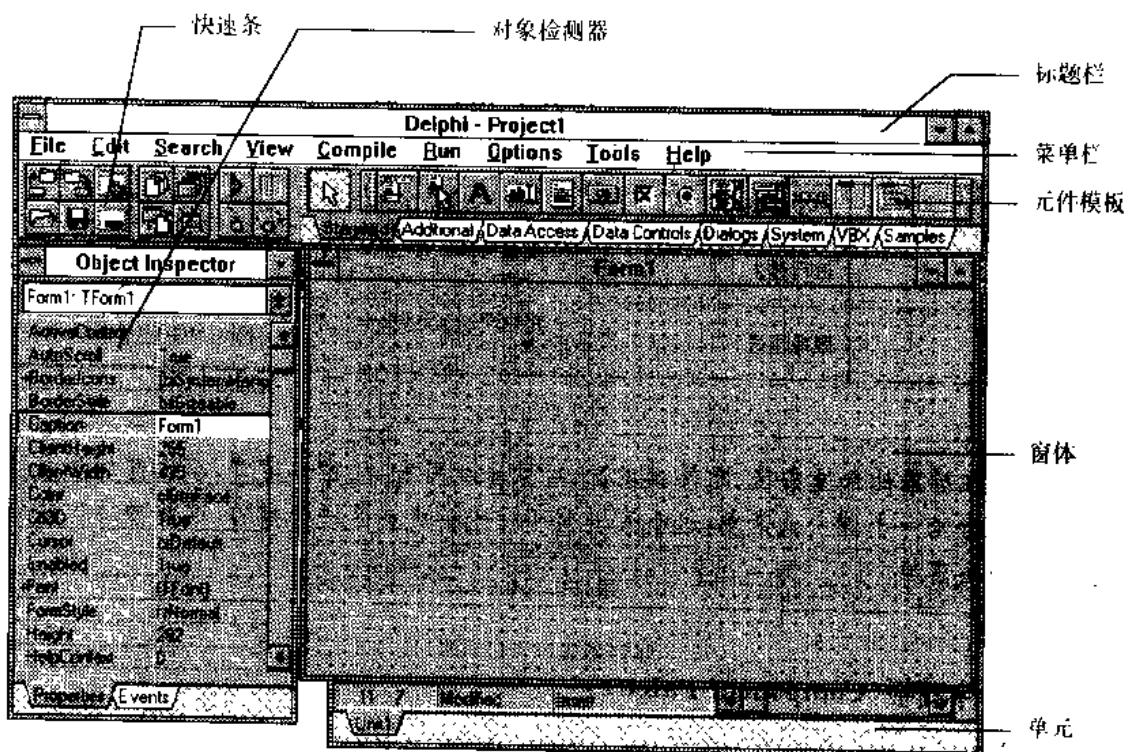


图 2.1 Delphi 界面

能的简单说明文字。下面是每一个按钮的说明。选择按钮的方法是用鼠标左键单击它。

读者也可以自行定义加速按钮。定义加速按钮的步骤是：

(1) 在加速条区中单击鼠标右键，从弹出的菜单中选择 Configure；

(2) 从弹出一个加速条编辑(SpeedBar Editor)窗口中选择一个需要加入到加速条中的功能图标，并将它拖曳到加速条区的任意一个位置，那么这个图标将覆盖以前的图标成为一个新的加速按钮。

表 2.1 加速条选项表

英文名称	中文含义	功能	等效菜单
Open Project	打开项目	打开一个已有的项目文件	File Open Project
Save Project	保存项目	保存设计者已有的项目文件	File Save Project
Add file to Project	向项目中加文件	向项目中加入已有的窗体或 Pascal 源代码文件	File Add File
Select file from List	从列表中选择观察单元	观察单元文件对话框弹出	View Units
Run	运行	运行应用软件	Run Run
Pause	暂停	暂停一个正在运行的程序	Run Program Pause
Open File	打开文件	打开一个已有的窗体文件或单元文件	File Open File
Save File	存储文件	存储一个已有的窗体文件或单元文件	File Save File

(续表)

英文名称	中文含义	功能	等效菜单
Remove file from Project	移除文件	从项目中移除一个窗体文件或单元文件	File Remove File
Toggle Form/Unit	窗体/单元切换	将窗体或单元窗口提升到最外面	View Toggle Form/Unit
New Form	产生新窗体	产生一个新的窗体文件	File New Form
Trace Into	运行过去	跟踪调试工具(第十二章介绍)	Run Trace Into
Step Over	跟踪步进	跟踪调试工具(第十二章介绍)	Run Step Over(F8)

提示：在屏幕的构造器区、元件模板区、对象检测器窗口、窗体窗口、单元窗口单击鼠标右键都可以弹出一个弹出式菜单，这个菜单项可对光标所在区进行操作，这些操作仅仅针对相应的窗口或区域。

2.1.4 元件模板(Component Palette)

Delphi 进行程序设计时最基本的元素是元件(Component)，它类似于 C 语言或其他面向对象语言中的对象，如按钮、菜单等都是一个元件。每个元件具有自己独特的属性，并且元件可对用户的一些操作通过事件进行响应。这样可以通过改变元件的属性以适应我们的需要，并且编制不同的事件处理函数来对元件发生的各种事件进行响应。元件的属性和事件设置都是通过对对象检测器来设置的。Delphi 根据元件的特性或功能将它们分布到不同的页次，元件模板下面是页次标签，单击元件页次标签就可看到不同类型的元件。初始的元件模板只有 75 个元件，在第三章中将向读者介绍更多的元件情况。

2.1.5 窗体(Form)

窗体是进行程序设计时的窗口，也是用来放置元件的窗口，可以形象地将它形容成一块“画布”。窗体文件的扩展名为 .DFM，我们可以将各种元件加入其中以形成用户界面。组成可视化窗体的文件代码允许以二进制格式存放，其他的应用软件无法读入。第四章中将详细介绍窗体。

2.1.6 对象检测器(Object Inspector)

对象检测器的作用是对放置在窗体中的元件(窗体本身也是元件)进行属性设置和编制事件处理函数。当我们在窗体中进行设计时，我们所使用元件的各种属性参数以及事件处理函数都可通过对象检测器显示出来，并进行修改。对象检测器分为属性页次和事件页次，单击属性页次或事件页次标签就可选择它们。属性(Properties)页次用来设计窗体及其元件的各种属性。事件(Events)页次用于定义与元件相关的事件处理函数。对象检测器如图 2.2 所示。