Delphi(1.0/2.0)

实用编程技术

王强 周康 白晓毅

西安电子科技大学出版社

Delphi(1.0/2.0)实用编程技术

王 强 周 康 白晓毅

西安电子科技大学出版社

1997

(陕)新登字 010 号

内 容 简 介

Delphi 是 Borland 公司新近推出的一种基于 Windows 操作系统的可视化编程工具。由于其方便使用的开发环境和丰富的控件及全面支持 Win 95 新界面和 32 位特性等，因而它正在赢得越来越多的编程人员的青睐。

本书将对 Delphi 的编程环境作一个全面的介绍。其中前十章主要介绍 Delphi 开发工具的基本特征，后面四章重点介绍了 Delphi 2.0 所提供的一些强大功能和使用技巧，并有相当的范例。读者可通过本书尽快掌握 Delphi 的编程技术。

Delphi(1.0/2.0)实用编程技术

王 强 周 康 白晓毅

责任编辑 李惠萍 梁家新

西安电子科技大学出版社出版发行

地址：西安市太白南路 2 号 邮编：710071

西安电子科技大学印刷厂印刷

新华书店经销

开本 787×1092 1/16 印张 17 4/16 字数 409 千字

1997 年 5 月第 1 版 1997 年 5 月第 1 次印刷 印数 1—6 000

ISBN 7-5606-0520-6/TP·0252 定价：22.50 元

前　言

Delphi 是 Borland 公司为与微软的 Visual Basic 相抗衡而推出的一种基于 Windows 操作系统的可视化编程工具。虽然 Delphi 出现不久，但它正以其所具有的一些其它编程环境无法比拟的优点而逐渐赢得众多编程人员的青睐。

本书作为介绍 Delphi 1.0 到 2.0 版的普及书籍，将对 Delphi 编程环境作一个全面的介绍。读者可通过本书尽快掌握 Delphi 的编程技术。

1. 方便使用的开发环境。Delphi 的开发环境类似于 VB，并保留了 Turbo Pascal 和 Borland C++中提供的强大的热键和调试功能，使得编程人员无须花费太多力气即可转到 Delphi 编程环境中来。Delphi 中的调试非常方便，它具有断点调试、变量监视、值模拟、单步和多步执行等多种方式。

2. 丰富的控件。Delphi 中提供了大量的控件，而且控件都以页的形式安排得井井有条，程序员可以很方便地利用这些控件来美化程序的外观并丰富程序的功能。另外，在 Delphi 中也可以调用 VBX 控件，Delphi 将自动为其包装。这一点使得编程人员能够继续享受已经出现的、大量功能强劲的 VBX 所提供的便利。

3. 以可视化 Pascal 语言作为核心。Delphi 的语言基础是 Borland Pascal，但其中完善了面向对象的特征，已经能够完全适应可视化程序设计的要求。Pascal 结构严谨，变量类型比 VB 中要求严格，这就使得编译后的可执行文件速度大大高于经伪编译获得的 VB 可执行文件，当然这要以文件尺寸增加作为代价。

4. 其它一些优秀特征。Delphi 作为一种面向对象的程序设计语言，具有许多面向对象的特征。而且，Delphi 还具有强大的数据库功能，可以挂接常用数据库；提供现成的工具条来直接完成简单的记录移动、修改等工作，并能够在程序中直接嵌入 SQL 语言。

当然，上面仅仅是列举了 Delphi 所具有的部分优点。随着 2.0 版的推出，Delphi 已直接面向 Win95 的 32 位操作系统，全面支持 Win95 新界面和 32 位特性，并且数据库功能也更加完善，可以大大简化复杂数据库的开发过程。

本书前十章主要介绍 Delphi 开发工具的基本特征，后面四章重点介绍了 Delphi 2.0 所提供的一些强大功能。

由于 Delphi 作为一种全新的 Windows 下的语言环境，其所具有的特征之多本书不可能一一点到，所以本书的重点在于“普及”，在于抛砖引玉。对于一些细致深入的工作，还有待读者自己去探索和钻研。

作　者

1997 年 1 月 20 日

目　录

第一章　面向对象简介

“面向对象”、“基于对象编程”的概念早已为程序设计者们所熟悉。但是，到底什么是“面向对象”，为什么要使用它，以及如何使用，这些问题都可以在本书的学习中找到答案。

在深入学习 Delphi 提供的功能强大、设计精巧的编程环境之前，我们首先来学习将要使用到的一些编程的相关知识。

§1.1　程序设计发展概况

从第一台计算机诞生起，人们就一直在寻找能使人—机交流更为简便的方式。早期的电子计算机是通过前面板上的开关来输入二进制的机器指令的，比如一条指令 10011011，代表着将一个数据调入内存的操作。显然，这种方式很笨拙，也难于理解。

汇编语言的出现使得指令输入变得比较方便，原来的二进制指令被简短的字母和数字所代替，比如前面提到的调用数据的指令可简洁明了地表示为 MOV [ES:DI], AX。汇编语言仍然要求程序员学习掌握一套机器汇编指令集。

前面所提到的二进制指令以及汇编指令，都是针对某种具体的 CPU。不同的机型所用的指令集是完全不同的，这使得程序设计变得相当麻烦。

BASIC 语言的发明使得情况有所改观。最初，这种解释型语言是用来讲述通用的编程概念(不考虑使用的计算机硬件类型)，程序员可以在一种机型上编程后拿到另一种机型上运行，这就是通常所说的可移植性。

然而，BASIC 语言本身也带有许多缺点。首先，由于它是解释执行，因而速度较慢；其次，BASIC 程序要兼容于各种硬件，因而其界面相当糟糕；再次，早期的 BASIC 是作为教学工具出现的，而非商品软件开发工具，因此一个工程需要大量的程序代码，常常大得难于控制；还有一点就是早期 BASIC 的功能很不完善，有许多问题无法利用它来解决。

随后一些年里，出现了多种程序设计语言，如 FORTRAN, COBOL, ALGOL, 等等。这些语言比起 BASIC 来有两点优越性:首先，它们都是编译型语言，编译型语言可以一次性翻译成机器语言并存储下来，因此运行速度远远超过解释型语言；其次，这些新语言都有各自的专用领域，FORTRAN 主要用于数学及科学计算，COBOL 常用于商业用途，而 ALGOL 则主要是利用算法结构来解决问题。

伴随着程序设计语言的发展，许多抽象概念应运而生。C 语言的出现更使得程序设计面貌一新。前几代程序设计语言都是利用过程化方法来完成预定任务的，而 C 语言则不然，它主要是通过函数来创建结构，这也正是 C 语言最显著的特征。

C 语言及其它类似的程序设计语言有一个很突出的优点——允许将函数集成于库中。利用这一特点，程序员可以开发出一些通用函数(包括数值计算、图形界面等)，然后通过函数库来扩展程序设计语言；与此同时，应用程序的可移植性也得以提高。

为了适应逐渐提高的可移植性和可重用性的要求，最新的程序设计语言(如 C++)中

都提出了“面向对象”的概念，这一概念也正是本章下一节要介绍的内容。

§1.2 面向对象

面向对象，可以看作是一种解决问题的方法，它要求程序员创建一个自包含的代码和数据并在其构造的环境中使用。在面向对象的程序设计方法中，基本的程序结构被称为对象。下面就以汽车为例来介绍如何创建一个对象。

汽车是一个对象，人们绝不会费半天口舌去说“利用轮胎滚动，依靠马达发动并有座位的交通工具”。一提到汽车，人们心中就会想到相应的交通工具，也就会联想到汽车是有轮胎、马达和座位的。

提到汽车，人们还会进一步想到汽车的具体类型，可以通过下面几项属性来加以区别：

- 马达
- 颜色
- 车体类型
- 产地

接下来要考虑的就是“功能”。汽车将人们从A地送往B地，但怎样到达目的地却是另外一回事。有些车开得快，有些车开得慢；有些车拉人，还有些车可以拉货。

定义了汽车的属性和功能之后，我们还应针对汽车所具有的一些共性来作些特殊处理。汽车有不同的外型、尺寸、产地，但它们还具有更多相同或相似的属性。对此应如何处理呢？可以将相同(或相似)的事物组合在一起。

例如，可以根据制造商对汽车进行分类组合，像福特、通用汽车等。尽管有尺寸大小之分，功能有强弱差别，但只要是同一厂家生产的汽车均可看作同一类型。

面向对象还有一个重要的概念——继承。所谓继承，就是从祖先那里得到一些东西，比如说人们的眼睛颜色、头发、形态等等。对于对象而言，继承有更特别的含义，它意味着一个对象是另一个对象的完整拷贝，但是该对象又具有一些附加的特性，从而与原来的对象有所不同。

现在返回来看汽车的例子。1964年福特公司推出了Mustang。虽然Mustang看起来与同一时期福特的其它产品有巨大差别，但是它仍然是由Ford Falon发展而来。尽管车体外型和内部构造有所不同，但是手工制造部分、马达等部件却没多少改变。因此可以说Mustang继承了Ford Falcon的许多特性。

下面就根据以上所讲的概念来描述一个用来区别汽车类型的方法。首先，列出区分不同汽车的部分属性：

- 马达(Engine)
- 内部构造(Interior)
- 车体外形(Body Style)
- 颜色(Color)
- 轮胎(Wheels)
- 尺寸(Size)

通过列出的属性，可以创建一个名为 CAR 的父对象。考虑到汽车种类繁多，难以细数，所以可将其划分为若干类进行管理。在这里可以利用制造商来分类：

- Ford
- GM
- Chrysler
- Toyota
- Nissan
- Honda
- BMW
- Mercedes
- Alpha

分类之后，就可以忽略同一厂家生产的汽车在模式上的差别了。现在创建一个新的汽车类——FORD，该类将继承父对象 CAR 的所有属性，额外地，FORD 类还具有下面一些新增属性：

- Insurance
- Acceleration
- Maintenance
- Model

表 1.1 中列出了 CAR 和 FORD 各自的属性。

表 1.1　属 性 列 表

CAR	FORD
Engine	Engine
Interior	Interior
Body Style	Body Style
Color	Color
Wheels	Wheels
Size	Size
	Insurance
	Acceleration
	Maintenance
	Model

从表中可看出，FORD 类拥有父对象 CAR 的所有属性，同时 FORD 类还有一些反映其自身特征的一些新属性。利用同样的方法，还可以为其它制造商定义相应的类，这些类都将在继承 CAR 所有属性的基础上拥有各自的一些专有属性。

接下来引入一些新的名词。由于CAR包含了所有汽车，因而可称其为基类，其它所有的汽车类均由它衍生而来。换句话说，CAR类是生成其它汽车类的模板，它不具备任何独特的性质。

下面，我们来看看在Delphi的基础语言——Borland Pascal中是如何定义和表示类与对象的。首先，可以简单地定义CAR对象如下：

```
CAR = object
Engine: Integer;
Interior: String;
BodyStyle: String;
Color: String;
Wheels: Integer;
Size: Integer;
end;
```

然后在CAR基础上添加一些新的属性，即可定义FORD类。

```
FORD = object(CAR)
Insurance: Integer;
Model: String;
Acceleration: Real;
Maintenance: Real;
end;
```

更进一步，可在FORD类的基础上定义MUSTANG类：

```
MUSTANG=object(FORD)
Intrumentation: String;
Convertible: Boolean;
Fony Fackage: Boolean;
Saleen: Boolean;
Suspension: Integer;
Special Pricing: String;
end;
```

现在，我们已经逐步推进地定义好了MUSTANG对象。也许读者感到不太理解，下面让我们来回顾一下前面所做的工作。

我们首先定义了一个父对象——CAR，它具有以下一些属性：

```
Engine:          Integer;
Interior:        String;
BodyStyle:       String;
Color:           String;
Wheels:          Integer;
Size:            Integer;
```

接着，我们定义了一个新对象——FORD，它继承了CAR的所有属性，此外我们还为

FORD 对象增加了 4 个其特有的属性。FORD 对象的全部属性如下所示：

Inherited ford CAR：

Engine：Integer；

Interior：String；

BodyStyle：String；

Color：String；

Wheels：Integer；

Size：Interger；

New attributes to this object：

Insurance：Integer；

Model：Real；

Maintenance：Real；

最后，我们创建了第 3 个对象——MUSTANG，它继承了 FORD 的属性，加上新增加的 6 个属性，MUSTANG 共有以下一些属性：

Inherited from CAR：

Engine：Integer；

Interior：String；

BodyStyle：String；

Color：String；

Wheels：Integer；

Size：Integer；

Inherited from FORD：

Insurance：Integer；

Model：String；

Acceleration：Real；

Maintenance：Real；

New attributes to this object；

Instrumentation：String；

Convertible：Boolean；

Pony Package：Boolean；

Saleen：Boolean；

Suspension：Integer；

Special Pricing：String；

到此为止，我们已经介绍了面向对象的有关基本概念，并讲述了如何利用 Borland Pascal 构造对象的基本方法。读者应仔细体会，这将有助于后面的学习。

第二章 Delphi 开发环境

Delphi 是一个快速应用开发环境，它的与众不同之处在于主要依靠大量集成在一起的部件来实现各种功能。每个部件均作为一个对象存在于 Delphi 环境中，并且可以通过功能扩充来创建新的部件。

本章将介绍 Delphi 环境中的一些主要的功能组件。读者将接触到工程(Project)的概念，并通过一个快速生成的小应用程序来理解利用 Delphi 开发 Windows 程序是多么地简单、迅捷。

§2.1 Delphi 环境概述

起动 Delphi，屏幕上显示出几个不同的窗口，这体现了 Delphi 的集成化特征。图 2.1 即为桌面的缺省显示。

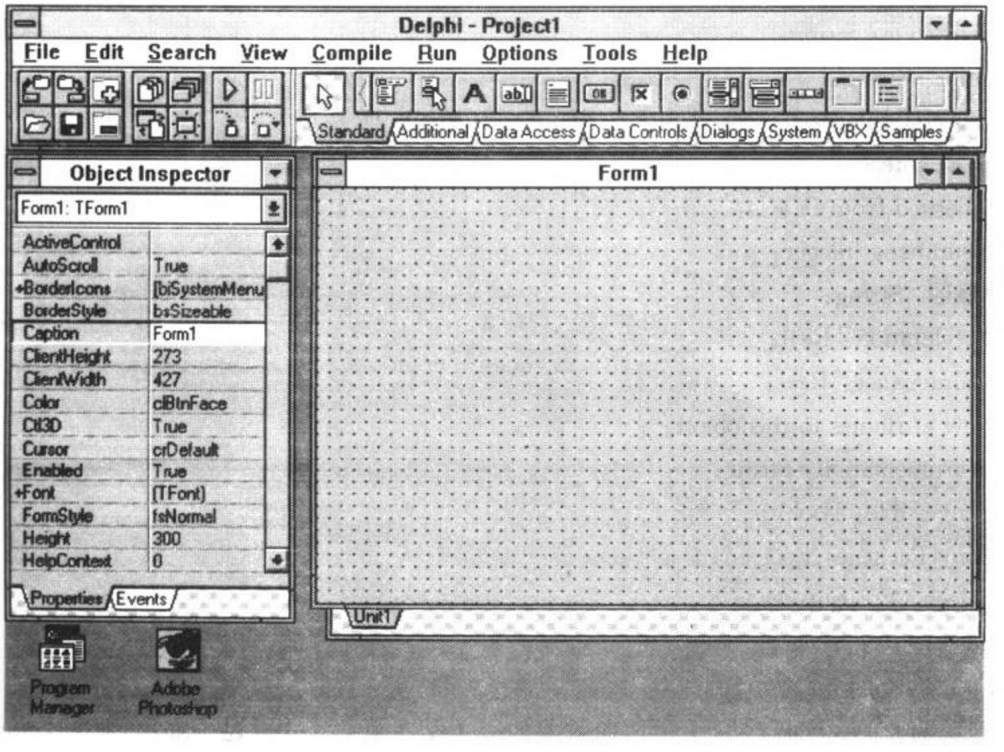

图 2.1 Delphi 集成环境

从图中可看到部件板、加速条、对象属性表、代码编辑器以及一个缺省的窗体。随着学习的逐渐深入，读者还可以根据自己的喜好来改变桌面的布局，具体方法将在后面进行

讨论。本节将分别介绍 Delphi 集成环境中的各个部分。

1. 加速条

加速条位于部件板窗口(如图 2.2)的左下角，它为一些常用的功能提供了快速执行的按钮，而这些功能都可以在主菜单中找到相应的菜单项。利用加速条，可以省却一级一级拉下菜单的繁琐操作。图 2.3 和表 2.1 分别展示了加速条的外观及各按钮的功能。

图 2.2 部件板窗口

图 2.3 加速条

表 2.1 加速条按钮功能

按 钮	功 能
Open Project	打开 Open 对话框，从中选取已建工程进行修改
Save Project	将当前工程存盘
Add File to Project	向工程中加入 Object Pascal 源文件
Select Unit from List	从源代码单元列表中选中一个代码单元，放到代码编辑器的最前端
Run	运行当前工程
Pause	暂时中止运行中的工程
Open File	利用 Open 对话框打开文件
Save File	将源文件存盘
Remove File From Project	从工程中删除源文件
Toggle From/Unit	切换显示窗体和代码单元
New Form	向工程中加入一个窗体
Trace Into	调试命令。逐步执行函数和过程调用
Step Over	调试命令。此命令将导致程序不进入调用的函数或过程，而是每次在当前代码单元中移动一行

2. 部件板

Delphi 提供了 8 个缺省的部件页(Component Page)，每页中包含不同的对象，供开发应用程序时使用(如图 2.4)。选择部件页的方法很简单，只需在代表某页的选项上点一下，

Delphi 即可将相应的部件页显示到部件板的前端，供用户查询和使用。

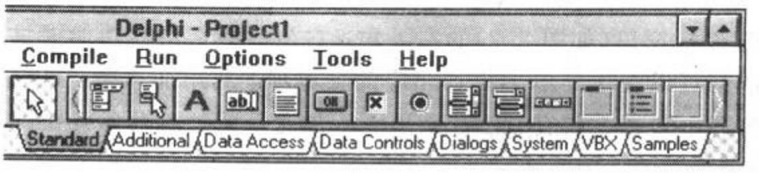

图 2.4　部件页

下面几个表格(表 2.2～2.9)分别列出了各部件板中所包含的部件及其各自的功能。

表 2.2　标准部件页(如图 2.5)

部　件	功　　能
MainMenu	调用菜单生成器来设计应用程序的主菜单，包括下拉子菜单以及加速键
PopupMenu	调用菜单生成器创建一个弹出菜单
Label	从窗体中设置文本显示，可用它来显示标题、注释以及其它有用的信息
Edit	提供一个单行编辑框编辑单行文本。这对于小型数据十分有用
Memo	提供一个多行编辑框供应用程序编辑大文本串。它具有简单文本编辑器的所有功能
Button	提供通用按钮作为一些操作的标志，如启动、停止或取消一个过程的执行。也可用按钮来激活函数、其它窗体、布尔操作以及与 Delphi 事件相关的其它操作
CheckBox	用以设置一些布尔型的条件(ON/OFF，YES/NO，TRUE/FALSE)。多个检查框的操作各自独立
RadioButton	与检查框类似，但常常是多个无线按钮一起使用，从中选中某一个
ListBox	显示一列表，供用户从中挑选一项或多项
ComboBox	操作方式与列表框类似，但它带有一个编辑框，可供用户输入选项
StrollBar	允许用户在一些显示内容超出显示空间的部件(如列表框、窗口等)中进行移动，从而显示不同的部分
GroupBox	允许组合窗体中的一些部件
RadioGroup	组合窗体中的无线按钮
Panel	组合窗体中的部件。与 GroupBox 有所不同，它可用来创建状态条、工具以及加速条等

图 2.5　标准部件页

表 2.3　附加部件页(如图 2.6)

部　件	功　　能
BitBtn	表面可带位图显示的按钮
Speed Button	多个 SpeedBar 组合在一个 Panel 对象上，可以创建一个加速条
TabSet	在窗体上创建带有立体感的条目
Notebook	提供一个多页显示对象，从而可将多层菜单和操作堆放于一个显示区域内。当选中某页时，位于该页的部件将被激活并显示出来
TabbedNote—book	基本上与 Notebook 类似，但在顶部带有条目
MaskEdit	允许格式化输入的编辑部件
Outline	显示数据的大体轮廓
String Grid	在表格中显示字符串并可进行修改
Draw Grid	显示位图、图标等
Shape	在窗体中创建简单的几何图形
Bevel	显示三维边界
Header	控制窗体的一部分来显示文本，并可用鼠标改变尺寸
ScrollBox	显示带有滚动条的区域

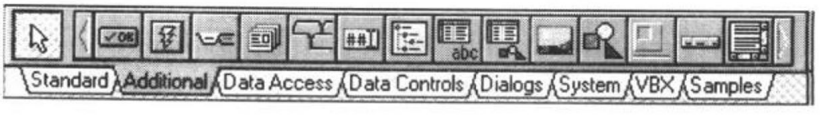

图 2.6　附加部件页

表 2.4　数据访问部件页(如图 2.7)

部　件	功　　能
DataSource	作为与数据相关部件同 Query 或 Table 之间的通道
Table	连接数据库表和 Delphi 应用程序
Query	构造并执行数据库查询
StoreProc	在本地存储远程数据的 SQL 过程
BatchMove	远程 SQL 的批处理
Report	利用 ReportSmith 来为 Delphi 应用程序生成报表

图 2.7 数据访问部件页

表 2.5 数据控制部件页(如图 2.8)

部 件	功 能
DBGrid	利用表格显示数据库中的数据
DBNavigator	提供一个工具条，允许显示、编辑数据库中的记录
DBText	Label 部件的数据板
DBEdit	Edit 部件的数据板
DBMemo	Memo 部件的数据板
DBImage	Image 部件的数据板
DBListBox	ListBox 部件的数据板
DBComboBox	ComboBox 部件的数据板
DBCheckBox	CheckBox 部件的数据板
DBRadioGroup	RadioGroup 部件的数据板
DBLookupList	从二级数据表中提取数据，在列表框中显示出来
DBLookupCombo	从二级数据表中提取数据，在混合框中显示出来

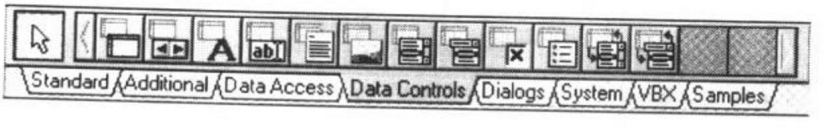

图 2.8 数据控制部件页

表 2.6 对话框部件页(如图 2.9)

部 件	功 能
OpenDialog	使用 Windows 的 Open 对话框
SaveDialog	使用 Windows 的 Save 对话框
FontDialog	使用 Windows 的 Font 对话框
ColorDialog	使用 Windows 的 Color 对话框
PrintDialog	使用 Windows 的 Print 对话框
PrintSetupDialog	使用 Windows 的 Print Setup 对话框
FindDialog	使用 Windows 的 Find 对话框
ReplaceDialog	使用 Windows 的 Replace 对话框

图 2.9　对话框部件页

表 2.7　系统部件页(如图 2.10)

部　件	功　　能
Timer	不可见部件。可以设定时钟中断以执行特殊任务，最短时间为 1 ms
PaintBox	在窗体上划出一个方形区域，用以绘图
FileListBox	利用列表框显示出当前目录下的文件
DirectoryListBox	利用列表框显示当前盘中的目录树
DriveComboBox	利用混合框显示出系统所有可用的驱动器
FileterComboBox	利用混合框来进行文件过滤
MediaPlayer	提供一个 VCR 型的 Panel，对多媒体文件进行操作
OLEContainer	在窗体中创建一个 OLE 客户区域
DDEClientConv	建立与 DDE 服务程序的连接
DDEClientItem	指定要交换的 DDE 客户数据
DDEServerConv	建立与 DDE 客户程序的连接
DDEServerItem	指定要交换的 DDE 服务器数据

图 2.10　系统部件页

表 2.8　VBX 部件页(如图 2.11)

部　件	功　　能
BitSwitch	创建一个功能与 CheckBox 类似的转换开关
BiGauge	显示过程的执行进度(%)
BiPict	显示位、图标
Chart	显示图标