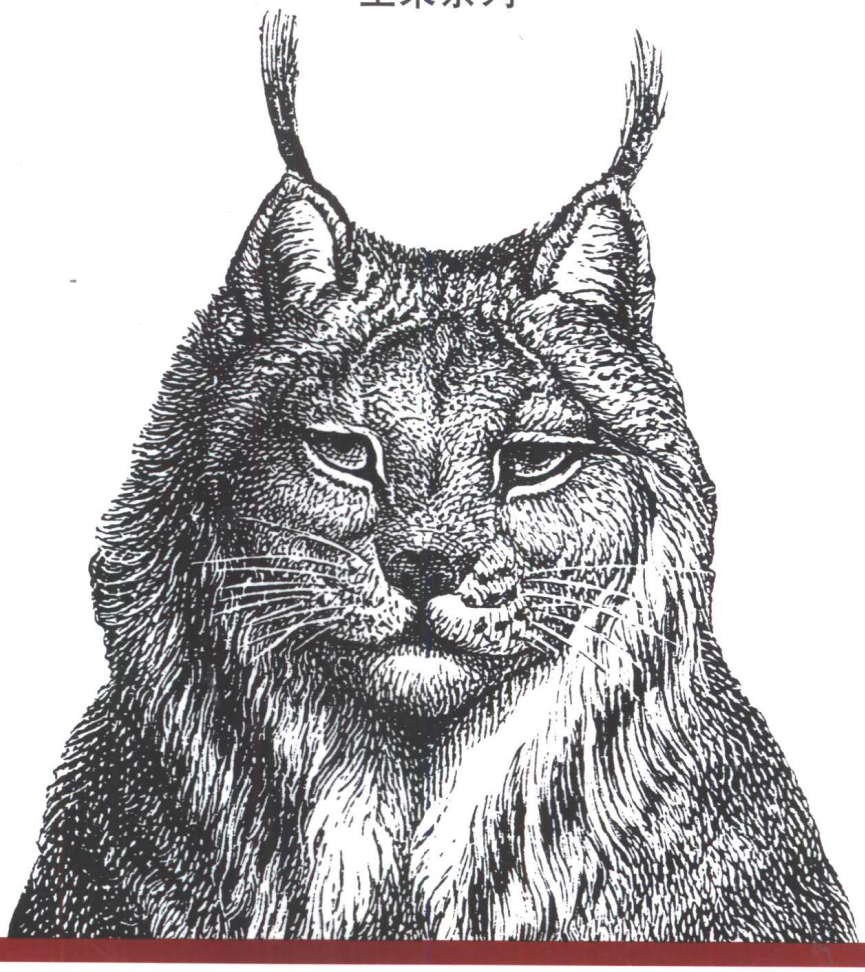
坚果系列



# DELPHI

# 技术手册

*Delphi in a Nutshell*

O'REILLY®
中国电力出版社

*Ray Lischner* 著
肖雪莲 朱腾辉 译

# DELPHI
# 技术手册

*Delphi in a Nutshell*

*Ray Lischner* 著

肖雪莲 朱腾辉 译

O REILLY®

*Beijing • Cambridge • Farnham • Köln • Paris • Sebastopol • Taipei • Tokyo*

中国电力出版社

**图书在版编目（CIP）数据**

DELPHI技术手册/（美）里希纳（Lischner, R.）编著；肖雪莲，朱腾辉译.
－北京：中国电力出版社，2001
书名原文：DELPHI IN A NUTSHELL
ISBN 7-5083-0542-6

Ⅰ.D... Ⅱ.①里... ②肖... ③朱... Ⅲ.DELPHI语言－程序设计 Ⅳ.TP312
中国版本图书馆CIP数据核字（2001）第06841号

北京市版权局著作权合同登记
图字：01-2001-0886号

© 2000 by O'Reilly & Associates, Inc.

Simplified Chinese Edition, jointly published by O'Reilly & Associates, Inc. and China Electric Power Press, 2001. Authorized translation of the English edition, 2000 O'Reilly & Associates, Inc., the owner of all rights to publish and sell the same.

All rights reserved including the rights of reproduction in whole or in part in any form.

英文原版由O'Reilly & Associates, Inc. 出版2000。

简体中文版由中国电力出版社出版2001。英文原版的翻译得到O'Reilly & Associates, Inc. 的授权。此简体中文版的出版和销售得到出版权和销售权的所有者——O'Reilly & Associates, Inc. 的许可。

版权所有，未得书面许可，本书的任何部分和全部不得以任何形式重制。

书　　名/ DELPHI技术手册
书　　号/ ISBN 7-5083-0542-6
责任编辑/ 关敏
封面设计/ Ellie Volckhausen，Hanna Dyer，张健
出版发行/ 中国电力出版社
地　　址/ 北京三里河路6号（邮政编码100044）
经　　销/ 全国新华书店
印　　刷/ 北京市地矿印刷厂
开　　本/ 787毫米×1092毫米　16开本　40.25印张　640千字
版　　次/ 2001年4月第一版　2001年4月第一次印刷
印　　数/ 0001-5000册
定　　价/ 65.00元（册）

## 作者简介

**Ray Lischner** 在Delphi社区非常有名。他为《Delphi Informant》、《Dr. Dobb's Journal》以及其他杂志撰写过无数的文章。每年的Borland/Inprise会议都有他的身影，他还曾经在全美国的Delphi用户组织中演讲。你可以给他发电子邮件，地址是nutshell@temest-sw.com。

## 封面介绍

本书封面上的动物是加拿大猞猁(Canadian lynx)。这种健壮有力的猫科动物，生活在北美的山区和北极区，在夜间捕食啮齿类和小哺乳动物。与近亲美洲山猫相比，它的爪子更大，因此在雪地里更加灵活，更能适应冬天的严寒条件。加拿大猞猁的毛发很厚，耳尖和颚有长长的须，看上去鬼鬼祟祟的。它是攀援高手，在岩石和树木间伏击猎物。

加拿大猞猁自己又是美洲狮和狼的猎物。但是人类应该为其濒临灭绝负责。两个多世纪以来，猎人为了获取它柔软的灰毛而过度捕杀，使加拿大猞猁在最南端的栖息地——科罗拉多洛基山脉的种群屠戮殆尽。为了防止加拿大猞猁的消失，美国的动物保护主义者已经行动起来。来自加拿大的四十只猞猁重新引入美国，放养在科罗拉多州的野生保护区。

# 目录


前言 ..... 1

第一章 Delphi Pascal ..... 7

单元 ..... 7
程序 ..... 11
库 ..... 12
包 ..... 14
数据类型 ..... 15
变量和常量 ..... 28
异常处理 ..... 29
文件I/O ..... 34
函数和过程 ..... 35

第二章 Delphi 对象模型 ..... 38

类和对象 ..... 38
接口 ..... 62
引用计算 ..... 67




试读结束，需要全本PDF请购买 www.ertongbook.com

消息 ..... 70
内存管理 ..... 71
旧式的对象类型 ..... 78

**第三章 运行时类型信息 ..... 80**
虚方法表 ..... 80
公布的声明 ..... 82
TypInfo 单元 ..... 88
虚方法和动态方法 ..... 96
初始化和结束化 ..... 98
自动的方法 ..... 100
接口 ..... 101
探究 RTTI ..... 101

**第四章 并发编程 ..... 105**
线程和进程 ..... 105
TThread 类 ..... 114
BeginThread 和 EndThread 函数 ..... 119
线程局部存储 ..... 120
进程 ..... 120
未来化 ..... 130

**第五章 语言参考 ..... 139**

**第六章 系统常量 ..... 488**
Variant 类型码 ..... 488
开放数组类型 ..... 489
虚方法表偏移值 ..... 490
运行时错误代码 ..... 491

第七章 运算符 ..... 494
一元运算符 ..... 494
多元运算符 ..... 497
附加运算符 ..... 498
比较运算符 ..... 499

第八章 编译器指示字 ..... 502

附录一 命令行工具 ..... 549

附录二 SysUtils 单元 ..... 566

词汇表 ..... 631

# 前言

Borland/Inprise公司的Delphi是一个组合了现代编程语言、集成开发环境（IDE）和可视化组件库（VCL）功能的强大的编程软件。Delphi的IDE很容易为任何使用过相似工具的人所熟悉。例如，你可以在一个所见即所得的窗体编辑器中简单地进行拖放操作，来可视化地设计一个窗口。更重要的是，由于Delphi编程语言的强大和弹性，其框架是面向对象、可扩展和可定制的。

Delphi的核心是Delphi Pascal编程语言，它具有支持IDE和VCL的关键特性。它具有一个现代面向对象语言的所有能力，同时保持了Pascal的优雅和简易。

《Delphi技术手册》是一本Delphi Pascal的全面参考手册。它涵盖了所有语言特性，并且突出了有效使用该语言的方法。有经验的Delphi程序员可以把本书作为以字母顺序排列的参考手册使用。Delphi新手则应该额外花一些时间在前几章上。我希望每一个人都能在这本书里面找到有价值的东西。

## 不是原来的Pascal

Delphi Pascal是Pascal的多种面向对象的变体之一。多年以来，Delphi经过发展，已经与你多年以前在学校使用的Pascal大不一样了。除基于单元的模块化编程和一个强健的类模型外，Delphi Pascal还具有许多其他现代语言的特征，包括如下：

- 接口（类似于Java™和COM接口）

- Unicode 字符串
- 属性
- 异常处理

Delphi一开始就作为一个Windows编程语言和环境，而且很多Delphi程序员（包括我自己）认为Delphi是所有Windows开发工具之中最好的。Delphi包括对COM和ActiveX的完全支持，一个面向对象的窗口部件库（称为可视化组件库，即VCL），和一个可扩展、可定制的快速应用程序开发环境。

## Linux上的Delphi

当我撰写本书的时候，Borland正在致力于把Delphi移植到Linux上（译注1）。可能在你读到这里的时候，已经可以使用Linux上的Delphi了，把它的集成开发环境带到了X-Window上，包括所见即所得的窗体编辑器、多层数据库支持，以及完全的CORBA支持。

直到Borland完成这项工作并发布Linux上的Delphi之前，我只能推测其最终产品看起来将会是如何的。（不，我并没有任何特别的内部消息。）可以相信在Linux上的Delphi和在Windows上的Delphi的核心语言是相同的，包括类、对象、接口、字符串、动态数组、异常，以及基本数据类型。大部分内置的子程序都能在Linux下（与在Windows下一样）工作。

本书中描述的一些语言特性明显只适用于Windows，例如`CmdShow`和`DllProc`变量或者`FindHInstance`函数。如果你希望编写能在Windows和Linux间移植的代码，应当避免使用这些Windows上的特性。

Windows下的Delphi是编写Windows应用程序和库最好的开发环境。为了达到这个第一的位置，Delphi合并了许多Windows特殊的特性。Borland的另一个目标是让Linux下的Delphi成为最好的Linux开发环境。为了实现这一目标，我们可以预料到Linux版的Delphi也会包括一些Linux的特性。

我仅仅是猜测而已，但我相信编写可在Windows和Linux间移植的代码是可行的。但是，你将不得不牺牲一些每个环境独特的特性。编写容易移植的组件，特别是交互式的控制，可能会成为一项令人望而生畏的任务。

---

译注1：Linux版的Delphi名为Kylix，已经发布。

# 关于本书

本书前四章介绍了关于如何有效使用Delphi的信息，后面的章节则构成了一个语言参考。

第一章“Delphi Pascal”，讨论Delphi Pascal与标准Pascal的区别。如果使用过Turbo Pascal或其他Object Pascal的变体，你可以快速阅读这一章，学习Delphi Pascal独有的新特性。同样地，如果从大学以来（而且在这之前）都没有使用过Pascal，那么你就必须仔细阅读这一章，以学习Delphi Pascal新的并且极好的特性。你可能会惊讶于多年之后，这个语言的发展如竟此之深远。

第二章“Delphi对象模型”，更深入地讨论了类和对象。如果你使用过Object Pascal的其他变体，则必须阅读本章，因为Delphi的对象模型又有很大不同。如果你对其他面向对象的编程语言已经有经验，那么阅读第二章可以学习Delphi与其他语言的不同，例如Java和C++。

第三章“运行时类型信息”，涵盖了Delphi的集成开发环境的关键。RTTI并没有被包含在Borland的正式帮助文件里，但任何编写或者使用组件的人（也就是每一个Delphi程序员）都应当理解RTTI的特性，包括它的局限性以及如何正确使用。第三章可帮助你了解RTTI的一切，以及其他一些内容。

第四章“并发编程”，是关于在一个现代化、多线程、多处理器的世界里如何使用Delphi。Delphi包括几种用于帮助编写多线程应用程序的语言特性，但这些特性可能很难使用，如果你对多线程编程的诀窍和陷阱不是太有经验的话。这一章可以让你开始有效地使用Delphi来编写现代应用程序。

第五章“语言参考”，是本书的主体。依字母顺序列出了Delphi Pascal语言和它的系统单元里的每一个关键字、指示字、子程序、类型和变量。并提供完整的例子展示如何正确有效地使用该语言。

第六章“系统常量”，包括相关联的变量的表格。第五章已经大得不能容下这些文字了，把它们放到一个单独的章节可以使整个参考更容易使用。

第七章“运算符”，描述了Delphi Pascal里面的所有算术和其他运算符。符号不好用字母排序，故把符号运算符列在它们自己的章节里，可以更容易寻找特定运算符的信息。

第八章“编译器指示字”，列出了所有可包括在源代码中的、控制Delphi编译和链接程序的注释。

附录一“命令行工具”，描述了随Delphi一起的各种命令行工具的用法和选项。这些工具与Delphi Pascal语言并没有联系，它们也经常被忽略，但它们对Delphi专业人员来说相当有用。

附录二“SysUtils单元”，列出了SysUtils单元里的所有子程序、类型和变量。这个单元并没有内置于编译器中（如同System单元被内置那样）。它不是Delphi Pascal语言的一部分。尽管如此，许多Delphi专业人员已经开始依赖SysUtils，如同它是语言的一部分一样。的确，SysUtils里的许多子程序都要优于在System单元里的等价部分（例如AnsiPos要好于Pos）。

# 本书中使用的约定

本书使用以下印刷惯例：

等宽字体（Constant width）

用于Delphi标识符和符号，包括所有关键字和指令。在语言参考中，等宽表示必须准确地如同所显示的那样使用的语法元素。例如，数组的声明要求方括号和其他符号、以及type、array和of关键字按如下所示的方式使用：

```
type Name=array[Index type, …] of Base type;
```

等宽斜体（*Constant width italic*）

在语言参考中用于表示必须由你的代码所代替的语法元素。在上面的例子中，必须提供类型*Name*，*Index type*和*Base type*。

等宽黑体（Constant width）

在更长的代码例子中用于突出显示包含被描述的语言元素的行。

斜体（*Italic*）

用于指出变量、文件名、目录名称以及词汇术语。

# 获得更多的信息

当你有关于Delphi的疑问时，应当首先参考Delphi的帮助文件。Delphi也具有大量的例子（在*Demos*目录里），它们通常比帮助文件更有用。

如果仍然不能找到所要的答案，可尝试把你的问题在众多Delphi新闻组的任一个里面提出来。有好几个标准的新闻组，而且Borland在它的服务器上维护着自己的新闻

试读结束，需要全本PDF请购买 www.ertongbook.com

组*forums.borland.com*。特别地，*borland.public.delphi.objectpascal*是有关Delphi Pascal语言问题的合适的新闻组。

如果你希望了解Delphi的集成开发环境、可视化组件库，或者其他关于Delphi编程的主题，最受欢迎的两本书是《Mastering Delphi 5》（Marco Cantu著，Sybex出版，1999年）和《Delphi 5 Developer's Guide》（Steve Teixeira和Xavier Pacheco著，Sams出版，1999年）。

如果你在本书中找到错误或疏忽之处，请发邮件到nutshell@tempset-sw.com以提醒我注意。我接收到的邮件太多，不能一一回复每条消息，但我保证所有邮件（任何通过了我的反垃圾过滤器的东西）都会过目。

# 建议与评论

本书的内容都经过测试，尽管我们做了最大的努力，但错误和疏忽仍然是在所难免的。如果你发现有什么错误，或者是对将来的版本有什么建议，请通过下面的地址告诉我们：

美国：

O'Reilly & Associates,Inc.
101 Morris Street
Sebastopol, CA 95472

中国：

100080北京市海淀区知春路49号希格玛公寓B座809室
奥莱理软件（北京）有限公司

询问技术问题或对本书的评论，请发电子邮件到：

info@mail.oreilly.com.cn

找寻勘误表和任何未来版本的计划，可以访问本书的Web站点：

http://www.oreilly.com/catalog/delphi

最后，您可以在WWW上找到我们：

http://www.oreilly.com
http://www.oreilly.com.cn

# 致谢

感谢 Tim O'Reilly 为出版 O'Reilly 的第一本 Delphi 著作而冒险。我期待着阅读和编写其他由 O'Reilly 出版的 Delphi 书籍。

技术编辑 —— Allen Bauer 和 Hallvard Vassbotn —— 为查找我的错误做了杰出的工作。Hallvard的丰富而详细的注释是无价的。任何残留的错误都是我在编辑们结束了他们彻底的工作之后误加上去的。

感谢我的编辑，Simon Hayes，以及 O'Reilly 的整个小组 —— Bob Herbstman，Troy Mott，以及设计和产品部的全体员工 —— 感谢他们把我粗陋的原稿变成了你现在看到的这本精美的书。

如果没有家庭的支持，我的任何一本书都不可能实现。感谢我的妻子，Cheryl，以及那位未来的程序员，Arthur，他使得我所有的工作都物有所值。



# 第一章

# Delphi Pascal

Delphi Pascal是传统Pascal的一个面向对象的扩展。它并不是ISO标准Pascal的一个很完整的扩展集，但如果你记得从学生时代学习的Pascal的话，现在你就可以轻易地掌握Delphi的扩展部分。Delphi并不只是一个另类的Pascal。在不使语言变得太复杂的情况下，Delphi增加了强有力的面向对象特征。Delphi具有类和对象、异常处理、多线程编程、模块化编程、动态和静态链接、OLE自动化以及其他很多很多功能。

本章描述了Delphi对Pascal的扩展。你应该已经熟悉传统Pascal或者其他一种流行的Pascal扩展，例如Object Pascal。如果你已经能区别Borland的Object Pascal和Turbo Pascal产品，你将需要学习一种新的对象模型（详见第二章），以及其他一些新特征。

Borland使用名称“Object Pascal”来指代Delphi的编程语言，但许多其他语言也使用这个名字，结果容易导致混淆。本书使用名称“Delphi Pascal”来指代Delphi编程语言，把Object Pascal留给其他多种面向对象的Pascal变体。

## 单元

Delphi Pascal是一种模块化的编程语言，其基本模块称为一个单元（*unit*）。为了编译并链接一个Delphi程序，需要一个程序源文件和源或对象形式的任意数目的附加单元。程序源文件通常被称为一个工程源文件，因为该工程可能是一个程序或者一个库——说得更精确些，一个动态链接库（DLL）。

当Delphi链接一个程序或库的时候，它可以静态地把所有单元链接为一个单独的*.exe*或*.dll*文件，或动态地链接到包里面的单元。包（package）即是一种特别的DLL，它包含一个或多个单元以及一些额外的逻辑，这些逻辑使Delphi能在一个包中隐藏静态链接单元和动态链接单元的区别。参见本章后面的“包”部分，以获得更多有关包的信息。

## 窗体和文件

窗体（*form*）是Delphi的术语，指一个可以用Delphi的GUI构造器编辑的窗口。对一个窗体的描述存储在一个*.dfm*文件中，它包含窗体的布局、内容和属性。

每一个*.dfm*文件都有一个关联的*.pas*文件，它包含该窗体的代码。窗体和*.dfm*文件是Delphi的集成开发环境（IDE）的一部分，但不是正式Delphi Pascal语言的一部分。尽管如此，该语言包含了几种只用来支持Delphi的IDE和窗体描述的特征。深入阅读第三章“运行时类型信息”可了解这些特征。

---

**注意：** 一个二进制的*.dfm*文件实际上是一个16位的*.res*（Windows资源）文件，它保持了与Delphi第一版的兼容性。版本2及以后的版本只能产生32位程序，故Delphi的链接器自动把*.dfm*资源转换为32位资源。这样，二进制*.dfm*文件通常与所有的Delphi版本都并不兼容。Delphi 5同样支持文本的*.dfm*文件。这些文件是简单的文本，与以前的Delphi版本并不兼容，至少在不具备转换回二进制格式的情况下如此。判断一个*.dfm*文件是二进制还是文本的唯一办法是打开它并检查其内容。一个简单的实现这一目的的程序化方法是测试前三个字节，在一个二进制的*.dfm*文件中它们总是$FF $0A $00。

---

表1-1简要描述了你可能在Delphi里面找到的文件以及它们的用途。标记有“(IDE)”的文件并不是Delphi Pascal的部分，但被IDE所使用。

**表1-1 Delphi文件**

| 扩展名 | 描述 |
|---|---|
| *.bpg* | 工程组（IDE） |
| *.bpl* | 编译包（特别类型的DLL） |
| *.cfg* | 命令行编译器选项 |
| *.dcp* | 编译包信息，链接一个包所需 |
| *.dcr* | 组件位图资源（IDE） |
| *.dcu* | 单元对象代码 |
| *.dfm* | 窗体描述（IDE） |
| *.dof* | 工程选项文件（IDE） |

表1-1 Delphi文件（续）

| 扩展名 | 描述 |
|---|---|
| *.dpk* | 建立一个包所需的资源文件 |
| *.dpr* | 程序或库的主体资源文件 |
| *.drc* | `resourcestring`声明的资源脚本 |
| *.dsk* | 桌面外观（IDE） |
| *.pas* | 单元资源代码 |
| *.res* | 窗口资源（每一个*.dpr*具有一个关联的*.res*文件） |

## 把接口与实现分离开来

单元具有两个部分：接口和实现。接口部分声明类型、变量、常量以及对其他单元可见的例程。实现部分提供在接口部分声明的例程的内容。实现部分可以有相对于单元的实现为私有的附加声明。这样，单元成为Delphi隐藏信息的主要手段。

一个单元可以使用另一个单元，更准确些，是从另一个单元导入其声明。对一个单元接口的改变要求所有使用了被修改单元改变后的声明的单元重新进行编译。Delphi的编译器自动处理这些，所以不需要使用makefile来编译Delphi工程。

可以在接口或实现部分使用一个单元，而且在建立一个工程时这个选择是很重要的：

- 如果单元A在它的接口部分使用单元B，单元B接口的改变就被当作对单元A接口的改变传播开来。Delphi必须重新编译所有使用单元A的单元。
- 如果单元A在它的实现部分使用单元B，则只有单元A必须为使用单元B中的新声明而重新编译。

单元不能在它们的接口部分进行循环引用。有时候会遇到两个类声明里包含相互依赖的声明。最简单的解决办法是使用一个单独的单元，但如果不得不在不同的单元声明类的话，可以在一个或两个单元都使用一个抽象基类来消除循环的依赖性。（参见第二章以获得更多信息。）

## 初始化和结束化

每一个单元都可以具有一个初始化部分和一个结束化部分。每一个初始化部分里的代码在程序或库的begin-end块之前运行。结束化部分里的代码则在程序终止或者库被卸载之后运行。Delphi使用对单元相关树的深度优先遍历来运行初始化部分。换句话说，在一个单元的初始化代码运行之前，Delphi就运行了它使用的每一个单元的

初始化部分。一个单元仅初始化一次。例 1-1 示范了 Delphi 如何初始化和结束化单元。

例 1-1：显示单元初始化的顺序

```
program Example1_1;
uses unitA;
{$AppType Console}
begin
  WriteLn('Example 1-1 main program');
end.

unit unitA;
interface
uses unitB;
implementation
initialization
  WriteLn('unitA initialization');
finalization
  WriteLn('unitA finalization');
end.

unit unitB;
interface
implementation
initialization
  WriteLn('unitB initialization');
finalization
  WriteLn('unitB finalization');
end.
```

编译和运行例1-1时，应确保它是在命令提示符而不是IDE下运行，否则控制台会在你能看到输出结果之前出现并消失，如例 1-2 所示。

例 1-2：运行例 1-1 的输出结果

```
W:\nutshell>example1_1
unitB initialization
unitA initialization
Example 1-1 main program
unitA finalization
unitB finalization
```

## System 和 SysInit 单元

System和SysInit单元自动包含在每一个单元之中，所以这些单元里面的所有声明都是 Delphi Pascal 语言的有效部分，且编译器特别熟悉 System 和 SysInit 单元里面的许多方法和过程。第五章“语言参考”将为你提供系统例程和声明的完整参考。

试读结束，需要全本PDF请购买 www.ertongbook.com