

---

# 前言

Borland/Inprise 公司的 Delphi 是一個組合了現代程式語言、整合式開發環境 (IDE) 和視覺化元件庫 (VCL) 等功能的強大程式語言。Delphi 的 IDE 很容易為任何使用過相似工具的人所熟悉。例如，你可以在一個所見即所得的表單編輯器中簡單地進行拖放操作，來視覺化地設計一個視窗。更重要的是，由於 Delphi 程式語言的強大和彈性，其架構是物件導向的、可擴展的及可自訂的。

Delphi 的核心是 Delphi Pascal 程式語言，它具有支援 IDE 和 VCL 的關鍵特性。它具有一個現代物件導向語言的所有能力，同時保持了 Pascal 的優雅和簡易。

《Delphi 技術手冊》是一本 Delphi Pascal 的全方位參考手冊。它涵蓋了所有語言特性，並且提出有效使用該語言的方法。有經驗的 Delphi 程式設計師可以把本書作為以字母順序排列的參考手冊使用。Delphi 新手則應該額外花一些時間在前幾章上。我希望每一個人都能在這本書裡面找到有價值的東西。

## 不是原來的 Pascal

Delphi Pascal 是 Pascal 的許多物件導向的變體之一。多年以來，Delphi 經過不斷的發展，已經與你多年以前在學校使用的 Pascal 大不一樣了。除了以單元為基礎的模組化程式語言和一個強健的類別模型外，Delphi Pascal 還具有許多其他現代語言的特徵，包括如下：

- 介面（類似於 Java™ 和 COM 介面）
- Unicode 字串
- 屬性
- 例外處理

Delphi 一開始就是作為一個 Windows 程式語言和環境的，而且很多 Delphi 程式設計師（包括我自己）認為 Delphi 是所有 Windows 開發工具之中最好的。Delphi 包括對 COM 和 ActiveX 的完全支援，一個物件導向的視窗元件庫（稱為視覺化元件庫，即 VCL），和一個可擴展的、可自訂的快速應用程式開發環境。

## Linux 上的 Delphi

當我撰寫本書的時候，Borland 正在致力於把 Delphi 移植到 Linux 上（譯註1）。可能在你讀到這？的時候，已經可以使用 Linux 上的 Delphi 了，把它的整合式開發環境帶到了 X-Window 上，包括所見即所得的表單編輯器、多層資料庫支援，以及完全的 CORBA 支援。

直到 Borland 完成這項工作並發佈 Linux 上的 Delphi 之前，我只能推測其最終產品的模樣。（不，我並沒有任何特別的內部消息。）可以相信在 Linux 上的 Delphi 和在 Windows 上的 Delphi 的核心語言是相同的，包括類別、物件、介面、字串、動態陣列、例外，以及基本資料型態。大部分內建的副常式都能在 Linux 下（與在 Windows 下一樣）工作。

---

譯註1 Linux 版的 Delphi 名為 Kylix，已經發行了。

本書中描述的一些語言特性明顯只適用於 Windows，例如 CmdShow 和 DIIProc 變數或者是 FindHIInstance 函數。如果你希望編寫能在 Windows 和 Linux 間互相移植的程式碼，應當避免使用這些 Windows 上的特性。

Windows 下的 Delphi 是編寫 Windows 應用程式和程式庫最好的開發環境。為了達到這個第一的位置，Delphi 合併了許多 Windows 特殊的特性。Borland 的另一個目標是讓 Linux 下的 Delphi 成為最好的 Linux 開發環境。為了實現這一目標，我們可以預料到 Linux 版的 Delphi 也會包括一些 Linux 的特性。

我僅僅是猜測而已，但我相信撰寫可在 Windows 和 Linux 間移植的程式碼是可行的。但是，你將不得不犧牲一些每個環境獨特的特性。撰寫容易移植的元件，特別是互動式的控制，可能會成為一項令人望而生畏的任務。製作一個可攜式的應用軟體可能是較容易些。

## 關於本書

本書前四章介紹了關於如何有效使用 Delphi 的資訊，後面的章節則構成了一個語言參考。

### 第一章 “Delphi Pascal”

討論 Delphi Pascal 與標準 Pascal 的區別。如果使用過 Turbo Pascal 或其他 Object Pascal 的變體，你可以快速閱讀這一章，以學習 Delphi Pascal 獨有的新特性。同樣地，如果從大學以來（而且在這之前）都沒有使用過 Pascal，那麼你就必須仔細閱讀這一章，以學習 Delphi Pascal 新的並且極好的特性。你可能會驚訝於多年之後，這個語言的發展竟如此之深遠。

### 第二章 “Delphi 物件模型”

更深入地討論了類別和物件。如果你使用過 Object Pascal 的其他變體，則必須閱讀本章，因為 Delphi 的物件模型又有很大不同。如果你對其他物件導向的程式語言已經有經驗，那麼閱讀第二章可以學習 Delphi 與其他語言的不同，例如 Java 和 C++。

### 第三章 “執行時期型態資訊”

涵蓋了 Delphi 的整合式開發環境的重點。RTTI 並沒有被包含在 Borland 的正式輔助文件？，但任何編寫或者使用元件的人（也就是每一個 Delphi 程式設計師）都應當理解 RTTI 的特性，包括它的局限性以及如何正確使用它。第三章可幫助你瞭解 RTTI 的一切，以及其他一些內容。

### 第四章 “並行程式設計”

是關於在一個現代化、多執行緒、多處理器的世界？要如何使用 Delphi。Delphi 包括幾種用於幫助編寫多執行緒應用程式的語言特性，但這些特性可能很難使用，如果你對多執行緒程式設計的訣竅和陷阱不是太有經驗的話。這一章可以讓你開始有效地使用 Delphi 來編寫現代應用程式。

### 第五章 “程式語言參考”

是本書的主體。依字母順序列出了 Delphi Pascal 語言和它的系統單元？的每一個關鍵字、指示字、副常式、型態和變數。並提供完整的例子展示如何正確有效地使用此語言。

### 第六章 “系統常數”

包括相關聯常數的表格。第五章已經大得不能容下這些文字了，把它們放到一個單獨的章節可以使整個參考更容易使用。

### 第七章 “運算子”

描述了 Delphi Pascal ？面的所有算術和其他運算子。符號不好用字母排序，故把符號運算子列在它們自己的章節？，可以更容易尋找特定運算子的資訊。

### 第八章 “編譯指示”

列出了所有可包括在原始程式碼中、控制 Delphi 編譯以及連結你的程式之特殊註解。

## 附錄 A “命令列工具”

描述了隨 Delphi 一起的各種命令列工具之用法和選項。這些工具與 Delphi Pascal 語言並沒有直接相關，它們也經常被忽略，但它們對 Delphi 專業人員來說相當有用。

## 附錄 B “SysUtils 單元”

列出了 SysUtils 單元的所有副常式、型態和變數。這個單元並沒有內建於編譯器中（如同 System 單元被內建那樣）。它不是 Delphi Pascal 語言的一部分。儘管如此，許多 Delphi 專業人員已經開始依賴 SysUtils，如同它是語言的一部分一樣。的確，SysUtils 的許多副常式都要優於在 System 單元的等價副常式（例如 AnsiPos 要好於 Pos）。

# 本書中使用的約定

本書使用以下印刷慣例：

### 等寬字體 (Constant width)

用於 Delphi 識別字和符號，包括所有關鍵字和指令。在語言參考中，等寬表示必須準確地如同所顯示的那樣使用的語法元素。例如，陣列的聲明要求方括號和其他符號、以及 type、array 和 of 關鍵字按如下所示的方式使用：

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

### 等寬斜體 (Constant width italic)

在語言參考中用於表示必須由你的代碼所代替的語法元素。在上面的例子中，必須提供類型 Name，Index type 和 Base type。

### 等寬黑體 (`Constant width`)

在更長的程式碼例子中用於突出顯示包含有被描述的語言元素之行。

### 斜體 (*Italic*)

用於指出變數、檔案名、目錄名稱以及辭彙術語。

## 獲得更多的資訊

當你有關於 Delphi 的疑問時，應當首先參考 Delphi 的輔助文件。Delphi 也具有大量的例子（在 Demos 目錄？），它們通常比輔助文件更有用。

如果仍然不能找到所要的答案，可嘗試把你的問題在許多 Delphi 新聞群組的任一個？面提出來。有好幾個標準的新聞群組，而且 Borland 在它的伺服器上維護著自己的新聞群組 `forums.borland.com`。特別地，`borland.public.delphi.objectpascal` 是有關 Delphi Pascal 語言問題的新聞群組。

如果你希望瞭解 Delphi 的整合式開發環境、視覺化元件庫，或者其他關於 Delphi 程式設計的主題，最受歡迎的兩本書是《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）。

如果你在本書中找到錯誤或疏忽之處，請發郵件到 `nutshell@tempset-sw.com` 以提醒我注意。我接收到的郵件太多，不能一一回覆每封郵件，但我保證所有郵件（任何通過了我的反垃圾篩檢程式的東西）都會過目。

## 如何連絡我們

本書的內容都經過測試，儘管我們做了最大的努力，但錯誤和疏忽仍然是在所難免的。如果你發現有什麼錯誤，或者是對將來的版本有什麼建議，請透過下面的地址告訴我們：

美國：

O'Reilly & Associates, Inc.

101 Morris Street

Sebastopol, CA 95472

台灣：

106 台北市大安區復興南路一段295巷21號1樓

美商歐萊禮股份有限公司台灣分公司

詢問技術問題或對本書的意見，請發電子郵件到：

[mail@oreilly.com.tw](mailto:mail@oreilly.com.tw)

找尋勘誤表和任何未來版本的計劃，可以造訪本書的 Web 站：

<http://www.oreilly.com.tw>

最後，您可以在 WWW 上找到我們：

<http://www.oreilly.com>

<http://www.oreilly.com.tw>

## 致謝

感謝 Tim O'Reilly 為出版 O'Reilly 的第一本 Delphi 著作而冒險。我期待著閱讀和編寫其他由 O'Reilly 出版的 Delphi 書籍。

技術編輯 —— Allen Bauer 和 Hallvard Vassbotn —— 為尋找我的錯誤做了傑出的工作。Hallvard 的豐富而詳細的注釋是無價的。任何殘留的錯誤都是我在編輯們結束了他們徹底的工作之後誤加上去的。

感謝我的編輯，Simon Hayes，以及 O'Reilly 的整個小組 —— Bob Herbstman，Troy Mott，以及設計和產品部的全體員工 —— 感謝他們把我粗陋的原稿變成了你現在看到的這本精美的書。

如果沒有家庭的支援，我的任何一本書都不可能實現。感謝我的妻子，Cheryl，以及那位未來的程式設計師，Arthur，他使得我所有的工作都是值得的。