

---

# 自序

本書所要探討的是系統的程式設計 — 特別是，Linux 上的系統程式設計。系統程式設計就是在撰寫系統軟體，也就是位於低階層的程式碼，可以直接跟核心與基礎系統程式庫互動。換句話說，本書的主題就是 Linux 系統呼叫以及其他低階函式，例如 C 程式庫所定義的那些。

儘管已經有許多書在探討 Unix 的系統程式設計，但是少有只將焦點放在 Linux 者，就算有，也少有在探討「最新 Linux 版本以及 Linux 獨有之進階介面」者。此外，本書的讀者將得益於我的特殊經驗：我曾為 Linux（包括核心以及系統軟體）撰寫過許多程式碼。事實上，我曾經實作過本書所探討的某些系統呼叫以及其他功能。因此，本書揭露了許多內幕知識，不僅會描述系統介面應該如何運作，也會說明它們實際的運作方式，以及你（程式設計者）如何以最有效的方式來使用它們。因此，本書是 Linux 系統程式設計的教材，也是 Linux 系統呼叫的參考手冊，亦是撰寫更聰明、更快之程式碼的權威指南。本書內容有趣且容易理解，不管你是否需要每日撰寫系統層級的程式碼，本書會教你讓你得以撰寫出較理想之程式碼的訣竅。

## 本書的讀者

本書假定讀者已經熟習 C 程式設計以及 Linux 的程式設計環境，不需要精通這些主題，但至少熟悉它們。如果你尚未讀過任何有關 C 程式語言的書籍，例如 Brian W. Kernighan 和 Dennis M. Ritchie 的經典之作《The C Programming Language》（Prentice Hall 出版；本書就是大家所熟悉的 K&R），我高度建議你閱讀這本書。如果你不習慣使用 Unix 文字編輯器，Emacs 和 vim 是最常見且備受推崇的編輯器，你可以擇一使用。你還應該熟悉 gcc、gdb、make 等等工具的基本用法。市面上可以找到許多與 Linux 程式設計有關的書籍；本書結尾的參考文獻列舉了幾本有用的參考書。

關於 Unix 或 Linux 系統程式設計，對讀者所應具備之知識做了以上假設之後。本書將從基礎開始談起，並蜿蜒前進到進階介面和優化訣竅。希望任何層級的讀者都會覺得這是一本值得閱讀以及能夠學到新東西的書。在本書寫作的過程中，我確實是這麼想的。

我也不是沒有假定本書讀者的類型以及閱讀本書的動機。本書適用於那些希望能夠在低層級寫出好程式的工程師，但是那些想在自己安身之處尋找更穩固立足點之較高層級的程式設計人員，也可以在本書找到許多對他們有用的資料。

不管你的動機為何，都祝你閱讀愉快。

## 本書的內容

本書分成 10 章、1 個附錄以及 1 個參考文獻。

### 第 1 章，介紹與基本概念

本章可做為導讀，它概述了 Linux、系統程式設計、核心、C 程式庫以及 C 編譯器。即使是進階的使用者也應該閱讀這一章 - 相信我。

### 第 2 章，檔案 I/O

本章將介紹檔案 (Unix 環境中最重要抽象概念) 以及檔案 I/O (Linux 程式設計的基本模式)，並且說明如何進行檔案的讀寫以及其他基本的檔案 I/O 操作，最後還會探討 Linux 核心實作和管理檔案的方式。

### 第 3 章，緩衝式 I/O

本章將探討基本檔案 I/O 介面的問題 - 緩衝區大小的管理 - 以及介紹一般的緩衝式 I/O (尤其是標準 I/O) 做為解決方案。

### 第 4 章，進階的檔案 I/O

本章將以進階的 I/O 介面、記憶體對映以及優化技術來結束本書對 I/O 的探討。本章還會附帶探討如何避免搜尋磁碟，以及 Linux 核心之 I/O 排程器扮演何種角色。

### 第 5 章，行程管理

本章將介紹行程 (Unix 第二個最重要的抽象概念) 以及基本行程管理的一系列系統呼叫，包括 fork。

### 第 6 章，進階的行程管理

本章將繼續探討進階的行程管理，包括即時行程。

### 第 7 章，檔案與目錄的管理

本章將探討建立、移動、複製、刪除以及其他用來管理檔案和目錄的功能。

## 第 8 章，記憶體管理

本章將說明記憶體管理。首先會介紹 Unix 的記憶體觀念，例如行程位址空間以及分頁，接著會探討從核心取得記憶空間以及將記憶空間釋回核心的介面。最後會探討進階的記憶體相關介面。

## 第 9 章，信號

本章將說明信號。首先會探討信號以及它們在 Unix 系統上所扮演的角色。然後會說明信號介面，先探討基本的部分，再探討進階的部分。

## 第 10 章，時間

本章將探討時間、修眠以及時鐘管理。本章會從基本的介面開始，一直到談起 POSIX 時鐘以及高解析計時器

## 附錄，GCC 對 C 語言的擴充

本附錄會回顧 *gcc* 和 GNU C 所提供的許多優化功能，例如 *pure* 和 *inline* 等用來標記函式的屬性。

本書最後提供了一份參考書清單，其中表列了對系統程式設計有幫助以及有論述本書未提到之必備知識的相關書籍。

# 本書所涵蓋的版本

Linux 系統介面可被定義成：Linux 核心（作業系統的中心元件）、GNU C 程式庫（*glibc*）以及 GNU C 編譯器（*gcc* - 現在被正式稱為 GNU Compiler Collection，但是我們只關心 C 編譯器）等三者所提供的應用程式二元介面（application binary interface）以及應用程式設計介面（application programming interface）。本書內容涵蓋 Linux 核心 2.6.22 版、*glibc* 2.5 版以及 *gcc* 4.2 版等三者所定義的系統介面。本書對系統介面所做的說明應該可以向下相容於較舊的版本（不含新的介面）以及向上相容於較新的版本。

如果有任何進化中的作業系統是移動的目標，Linux 就是一隻飢渴的獵豹。Linux 的進展是日數（而非年數）來度量的，頻繁地釋出核心與其他元件，並且在不斷變化中。沒有哪一本書可以永久捕獲這隻充滿活力的怪獸。

儘管如此，系統程式設計所定義的程式設計環境不再改變。核心開發者盡可能不去改變系統呼叫，*glibc* 開發者高度重視向上（forward）和向下相容性（backward compatibility），而且 Linux 工具鏈（toolchain）可以產生跨版本（特別是 C 語言）的相容程式碼。因此，儘管 Linux 不斷在變化中，Linux 系統程式設計仍能保持穩定。我想說的事情很簡單：不要擔心系統介面有任何變化，購買本書就對了！

## 本書的排版慣例

下面是本書的排版慣例：

### 斜體字

用於強調 URLs、Unix 命令與共用程式、檔案名稱、目錄名稱以及路徑名稱。

### 定寬字

用於標頭檔、變數、屬性、函式、型別、參數、物件、巨集以及其他編程結構。

### 定寬斜體字

用於指出文字（例如，一個路徑名稱元件）中可以代換成實際值的部分。



此圖示代表「訣竅」、「建議」或「一般注意事項」。

本書大部分的程式將以簡單但實用的程式碼片段來呈現，例如：

```
while (1) {
    int ret;

    ret = fork ();
    if (ret == -1)
        perror ("fork");
}
```

本書煞費苦心地提供簡潔而實用的程式碼片段。你不會看到特殊的標頭檔、沒有節制的巨集和難以辨認的捷徑。你不會看到龐大的程式實作，你所看到的是許多簡單的範例程式。本書的範例，不僅描述充分、完全可用，而且還小而明確，希望這些範例在讀者首次閱讀本書時能提供一個有用的說明，而且在之後多次閱讀本書時仍然能成為很好的參考資料。

本書所有範例幾乎都是各自獨立。這意味著你可以輕易地將它們複製到你的文字編輯器，將它們使用在實際的用途上。除非另有提及，所有的程式在建構時應該不需要用到任何特殊的編譯器旗標。（只有在少數的情況下，才需要連結特殊的程式庫。）建議以如下的命令來編譯原始碼檔案：

```
$ gcc -Wall -Wextra -O2 -g -o snippet snippet.c
```

這道命令會將原始碼檔案 *snippet.c* 編譯成可執行的二元檔案 *snippet*，開啟許多警告檢查，進行顯著但合理的優化動作以及除錯。本書的程式碼使用這道命令進行編譯應該會產生錯誤或警告 - 當然，你可能首先必須為 *snippet*（程式碼片段）建立一個骨架程式。

介紹一個新函式時，本書會採用常見的 Unix manpage 格式並以粗體字強調，如下所示：

```
#include <fcntl.h>
```

```
int posix_fadvise (int fd, off_t pos, off_t len, int advice);
```

所需要的標頭檔以及任何必要的定義都會放在開頭，後面跟著完整的呼叫原型。

## 使用範例程式

本書的宗旨是協助你搞定工作。一般而言，你可以將本書的程式碼用在你的程式裡，或是在文件裡提及，而無須要求我們的同意，除非你想大幅引用。舉例來說，使用本書範例裡的幾個程式片段，無須經過我們的同意；但如果打算將 O'Reilly 書籍裡的範例程式燒錄成光碟來銷售或散佈，則需要授權。引用本書內文或程式片段來回答問題，不需要授權；但如果大量引用本書範例到你的產品說明書裡，則需要知會我們。

如果你引用本書（內文或範例程式），我們會感謝你註明出處，但沒要求你必須得這麼做。如果你願意，請註明書名、作者、出版公司以及 ISBN。例如："Linux System Programming by Robert Love. Copyright 2007 O'Reilly Media, Inc., 978-0-596-00958-8."

## 建議和問題

歐萊禮公司是世界性的電腦資訊出版公司。我們永遠樂意聽到讀者對出版品的意見，包括如何讓本書可以更好的建議、指正本書的錯誤、或是讀者建議本書往後改版時，應該再加進來的其它主題。以下是本公司的聯絡資料：

美商歐萊禮股份有限公司台灣分公司

電話：(02) 2709-9669 傳真：(02) 2703-8802

網頁：<http://www.oreilly.com.tw>

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

與本書有關的線上資訊（可能包括勘誤、範例程式、相關連結）：

原文書

<http://www.oreilly.com/catalog/9780596009588/>

中文書

[http://www.oreilly.com.tw/product2\\_linux.php?id=a229](http://www.oreilly.com.tw/product2_linux.php?id=a229)

## 致謝

本書是在許多人的協助之下完成的。儘管可能掛一漏萬，不果我真誠地感謝那些一路上鼓勵、指導並支持我們的友人們。

Andy Oram 是一位不平凡的編輯和人物。沒有他的努力是無法完成這項艱難的工作的。難得的是，Andy 夫婦熟知如何用英語寫詩的技術。

Brian Jepson 曾做過本書的編輯，儘管編輯已經換人，他所做的努力仍繼續在本書餘波盪漾著。

本書有幸能夠找到學有專精的人士做為技術審閱者，沒有他們的努力本書將不是你現在看到的樣子。這些技術審閱者是 Robert Day、Jim Lieb、Chris Rivera、Joey Shaw 以及 Alain Williams。如果還有任何錯誤，我應該負最大的責任。

Rachel Head 善盡了她文稿編輯（copyeditor）的責任。在她的校稿之下，本書的文稿點綴著紅色的墨水 — 相信讀者一定會感激她所做的努力。基於各種理由，我想要感謝 Paul Amici、Mikey Babbitt、Keith Barbag、Jacob Berkman、Dave Camp、Chris DiBona、Larry Ewing、Nat Friedman、Albert Gator、Dustin Hall、Joyce Hawkins、Miguel de Icaza、Jimmy Krehl、Greg Kroah-Hartman、Doris Love、Jonathan Love、Linda Love、Tim O'Reilly、Aaron Matthews、John McCain、Randy O'Dowd、Salvatore Ribaud 和他的家庭、Chris Rivera、Joey Shaw、Sarah Stewart、Peter Teichman、Linus Torvalds、Jon Trowbridge、Jeremy VanDoren 和他的家庭、Luis Villa、Steve Weisberg 和他的家庭，以及 Helen Whisnant。最後要感謝我的父母 Bob 和 Elaine。

— Robert Love  
於波士頓