序

Greg Wilson 於 2006 年提出《美麗程式》的構想，讓頂尖軟體開發人員和電腦科學家，傳達他們領悟到的專業知識。他和編輯同事 Andy Oram，聯絡全球各種背景的專家，並收到雪片般的回應，部份原因是因為這本書的版稅已捐給美國國際特赦組織總部（Amnesty International）。這個專案的結果就呈現在本書中。

本書範圍寬廣，但也只代表這個最令人興奮的領域中、正在發生的一小部份而已。其他上千個專案，也一樣很有趣，很有教育意義，每天都由其他我們沒連繫的程式設計師往前推進。此外，很多我們邀稿的優秀實踐家都沒出現在本書中，因為他們當時太忙，無法投稿給 Amnesty International，或者有職責衝突。為了從眾位專家的專門知識中受益，我們希望未來繼續經營類似性質的其他書籍。

本書組織

第 1 章，正規運算式比對器——Brian Kernighan，說明對語言和問題的深入理解，可以得到精簡而優雅的解法。

第 2 章，Subversion 的 Delta 編輯器：只談介面——Karl Fogel，從精選的抽象介面開始討論，示範系統未來發展上的統一效應。

第 3 章，我從沒寫過的最美麗程式——Jon Bentley，建議如何評估程序，而不實際去執行。

第 4 章．找東西－－Tim Bray，抓起電腦科學裡的千頭萬緒，探索許多計算任務的一個基本問題。

第 5 章．正確＞美麗＞快速：設計 XML 查證器的教訓－－Elliotte Rusty Harold，調和完整性和良好效能的衝突目標。

第 6 章．整合測試的框架：來自脆弱性的美感－－Michael Feathers，介紹一個打破規則、自成優雅方案的範例。

第 7 章．美麗測試－－Alberto Savoia，說明採取寬廣而具創意的測試方式，不僅可消除臭蟲，還可讓你變成更棒的程式設計師。

第 8 章．動態產生影像處理程式－－Charles Petzold，往下降一層，以改良效能，同時維護可移植性。

第 9 章．由上而下運算子優先權－－Douglas Crockford，復興一個幾乎已令人忘懷的分析技巧，並說明該技巧與受歡迎的 JavaScript 有何相關性。

第 10 章．追尋加速的族群計數－－Henry S. Warren, Jr.，揭露一些聰明演算法對一個看似簡單問題的衝擊。

第 11 章．安全通訊：自由技術－－Ashish Gulhati，討論安全性訊息傳輸應用程式的演化方向，其設計是為了讓用戶可直覺存取精緻但時常令人困惑的加密演算法。

第 12 章．BioPerl 的美麗程式－－Lincoln Stein，討論具彈性的語言，加上特殊設計的模組，將如何讓具有普通程式設計技巧的使用者為資料建立強大的視覺形式。

第 13 章．Gene Sorter 的設計－－Jim Kent，結合簡單建構區塊，替基因研究人員產生健全而有價值的工具。

第 14 章．優雅程式隨硬體演化：以高斯消去法為例－－Jack Dongarra 和 Piotr Luszczek，檢視 LINPACK 和相關主要軟體套件的歷史，以說明面對新計算架構時，原始假設必須不斷重新評估。

第 15 章．美麗設計的長期益處－－Adam Kolawa，專心於良好設計原則的結果，如何協助 CERN 成為廣泛使用的數學程式庫（LINPACK 的前身），經得起數十年的考驗。

第 16 章．Linux 核心驅動程式模型：一起運作的益處－－Greg Kroah-Hartman，說明共同合作者付出多少心血以解決不同問題，因而得以讓一個複雜、多執行緒系統成功演化。

第 17 章，另一層間接層——Diomidis Spinellis，示範以許多驅動程式和檔案系統模組讓常做的運算得以抽象化，使得 FreeBSD 核心的彈性和維護性有所提昇。

第 18 章，Python 的辭典：滿足所有人的需要——Andrew Kuchling，說明謹慎設計以及替少數特殊案例做調整的結果，可讓一個語言功能支援許多不同用法。

第 19 章，NumPy 的多維迭代器——Travis E. Oliphant，帶你走過把複雜度隱藏在簡單介面下的設計步驟。

第 20 章，NASA 火星探測任務～高度可靠的企業系統——Ronald Mak，使用業界標準、最佳實務及 Java 技術，以吻合 NASA 探險需求中無可置疑的可靠度。

第 21 章，ERP5：追求最大適應力的設計——Rogerio Atem de Carvalho 和 Rafael Monnerat，說明強大的 ERP 系統如何以自由軟體工具和彈性架構開發而成。

第 22 章，一匙污水——Bryan Cantrill，陪伴作者走過一個令人心急如焚的臭蟲驚恐之旅，以及一個違反期待的聰慧解法。

第 23 章，分散式程式設計和 MapReduce——Jeff Dean 和 Sanjay Ghemawat，一個替 Google 的大規模分散式資料處理、提供易於使用程式設計抽象介面的系統，並可自動處理許多困難的分散式計算面向，包括自動平行化、負載平衡、以及失敗處理。

第 24 章，美麗的並行性——Simon Peyton Jones，經由 STM 移除平行程式的許多困難；以 Haskell 示範。

第 25 章，語法抽象：syntax-case 展開器——R. Kent Dybvig，說明 Scheme 如何保護巨集（許多語言和系統的關鍵功能），以免產生錯誤輸出。

第 26 章，省力架構：網路軟體之物件導向框架——William R. Otte 和 Douglas C. Schmidt，對分散式記錄機制，施用一些標準物件導向設計技術，諸如模式和框架，以保持系統彈性和模組化。

第 27 章，整合生意夥伴：REST 方式——Andrew Patzer，示範設計者對其程式設計師的尊敬，也就是讓 B2B 網路服務的設計滿足需求。

第 28 章，美麗除錯——Andreas Zeller，說明以遵守紀律的做法驗證程式，如何減少追蹤錯誤的時間。

第 29 章，把程式視為文章——Yukihiro Matsumoto，提出驅動他設計 Ruby 程式語言的一些挑戰性原則，而予以擴大的話，一般而言，有助於產生較佳之軟體。

第 30 章，對世界的唯一聯繫——Arun Mehta，帶你走一趟文字編輯系統中所牽涉的令人驚訝的介面設計抉擇，可讓重度肢障人士，如 Stephen Hawking 教授，得以透過電腦與外界溝通。

第 31 章，Emacspeak：完備的音訊桌面——T. V. Raman，說明 Lisp 的 advice 工具如何用於 Emacs，以解決 Emacs 環境中共同需要的一般性需求（產生豐富的語音輸出），但是又不修改較大軟體系統的底層原始碼。

第 32 章，運行中的程式碼——Laura Wingerd 和 Christopher Seiwald，列出一些簡單原則，對於程式設計準確度有意想不到的重大衝擊。

第 33 章，替「算書」寫程式——Brian Hayes，探索解決計算幾何中看似簡單的問題的沮喪，以及其令人驚訝的解答。

本書體裁

本書使用下列字型體裁。

楷體字

表示新詞彙、重要詞彙。

斜體字（Italic）

表示數學變數、URL、檔案或目錄名稱、指令。

`寬體字（Constant width）`

表示程式碼、檔案內容、輸出文字。

使用範例碼

這本書是用來幫助各位完成工作。一般而言，你可以在自己的程式或文件中使用本書的程式碼，不需要聯繫出版商取得許可，除非你更動了程式中相當重要的部分。舉例來說，撰寫使用好幾段本書範例碼的程式不需要取得許可，但販售或散播 O'Reilly 圖書的範例碼光碟絕對需要我們的允許。引用本書的說明和範例碼回答問題不需要取得許可，但把本書中的重要範例碼合併到你的產品文件則需要我們的允許。

我們希望你能放上版權宣告，但並非絕對必要。版權通常包括書名、作者、出版商和 ISBN 碼。例如：「《美麗程式》，Andy Oram 與 Greg Wilson 編輯策畫，Copyright 2008 O'Reilly Media, Inc，ISBN 碼」。

如果你覺得自己使用範例碼的程度，超出了上述的允許範圍，歡迎與我們聯繫：*permissions@oreilly.com*

批評與建議

請將你對本書的寶貴意見及問題告訴我們。來信請寄：

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

網址：*http://www.oreilly.com.tw*

電子郵件：*mail@oreilly.com.tw*

O'Reilly 的每一本書都有專屬網頁，上面有勘誤表與各種相關資訊，網址是：

http://www.oreilly.com/catalog/9780596510046/（英文書）

http://www.oreilly.com.tw/product2_tip.php?id=a231（中文書）