
序

關於這本書

本書以簡明的內容介紹 Python，讓你能很快了解並學會 Python 這個程式語言。Python 是廣受歡迎的物件導向語言，無論寫獨立自主的程式，或者發展命令稿的應用軟體，Python 都能提供最佳的服務方案，而且可應用的領域相當廣泛。Python 不僅可以免費取得，移植能力優良而且威力強大，更重要的是，Python 易學易用，這才是 Python 得以迅速流傳開來的原因。本書的主要目的就是帶著你很快的看過一遍 Python 的核心，不論你是新手或專業人士，開卷有益，不過在那之前我們希望你先把序言的部份讀完，知道這本書的編排方法，心中才能有個概要。

本書內容

關於 Python 的基本觀念你都可以在這本書裡找到，不過，我們將重點放在強調程式執行的速度以及如何讓程式碼寫得愈短愈好，內容著重於 Python 的核心概念，可能有些部份會令你覺得太過簡潔了。因此，我們將這本書歸類為入門的指引以及進階自修的跳板。

例如，Python/C 的整合就不會有大篇幅的報導。一般而言，倘若系統以 Python 為主，對系統的中樞部份而言，C 語言往往是多餘的配角。就事實而論，Python/C 的整合原本就相當複雜，果真要舉例說明，恐怕程式範例會大到嚇跑入門的讀者。此外，Python 的歷史、Python 愛用者的組織以及有關的軟體開發理論等等都不會佔太多的篇幅。其它諸如 GUI、系統工具、網路命令稿和數值設計等等，雖然都是 Python 引以為傲的應用程式，然而基於入門的編寫原則，我們只會在本書的結尾給予扼要的說明。看來，許多環節都遺漏了。

大體而言，Python 對命令稿語言的世界帶來了品質提昇的衝擊。Python 實作的觀念很多不是書中隻字片語可以道盡的，因此，除了這本書的內容之外，如果我們沒有進一步指出更高階的教材和資料，那只能說是兩位作者失職，沒有盡到應盡的本份。我們誠摯的希望讀者完成此書的巡禮之後，能再勇往直前，找更多的資料以得到更完整的理解和認識。如歐萊禮出版的 Programming Python 一書，就是絕佳的參考教材，書中有關於 Python 更深入的主題【註】。學習 Python 的不二法門就是多讀多看多寫，本書列舉的範例著實有限，讀者可以到網站上搜尋眾家提供的各類程式碼，從中吸取菁華，才能舉一反三，學以致用。

然而，也因為此書的侷限，反而使這本書成為最佳的 Python 入門教材。你會學到所有必備的基礎知識，並有能力開發獨立執行的 Python 程式。你會從這本書學到 Python 的核心、常見的工作任務、以及究竟可以應用在那些方面。看完這本書，就配備了足夠的火力，任何進階的議題相信都再也難不倒你了。

本書架構

本書的材料絕大多數來自於 Python 訓練課程的教材。各章的問題集都可在附錄 C 找到的解答。編纂問題集的目的在於訓練讀者的思考能力以及養成上機寫程式的習慣。我們建議讀者一定要親自做一做問題集的題目，不僅是為了得到寫作程式的經驗，同時也是因為問題集提出來的討論議題多半是本文無法涵蓋的範圍。如果思想淤塞，附錄 C 的解答不妨翻它一翻，解不解得出來並不代表智商的高低。當然啦，讀者要先上網把 Python 抓下來安裝，才能上機操作書中的範例並研究問題集的解答。

註 你可以到 <http://www.ora.com> 及 <http://www.python.org> 找到更多與 Python 相關的詳細資料。Programming Python 的作者是本書作者之一，書中詳細探討關於程式設計的實際議題。

為了快速的介紹 Python 的基礎與知識，因此書本的編排方式會以 Python 的特性做為章節區分的準繩，而非以範例為基調來介紹 Python 語言。我們會沿著語言的組織體系，由內建物件型態開始討論起，再到敘述，再到程式單元等等（如果你看過 Programming Python 的附錄 E，你可能會覺得有些內容似曾相識）。每一章的內容皆各自獨立，然而觀念卻彼此前後關聯（例如：當我們在紹類別時，假定你已經會寫函式了）。因此，以一章一章循序漸進的方式閱讀，最適合這本書。整體而言，此書分成下列三個部份：

第一部：核心語言

第一部介紹 Python 語言，由小地方講起，一路過關斬將，直達神奇的流程控制為止。每一種語言的特性都獨立成一章加以介紹，如型態、函式等等。各章列舉的範例都盡量保持簡短獨立，並充分說明文章的重點。以下簡要的說明各章節的內容。

第一章 入門指引

簡單的介紹 Python 以及執行 Python 程式的方法，如此讀者才能立刻開始練習程式碼編寫，並享受成果。

第二章 物件型態和運算子

探討 Python 主要的內建物件型態：數值、串列、辭典等等。只要能善加利用這些內建的資料型態物件，很多複雜的工作都能輕鬆完成。

第三章 基本敘述

討論 Python 的敘述。物件的建立和操作都得透過敘述才能進行。

第四章 函式

函式是 Python 高階程式架構的工具之一，程式碼如要再使用，函式是不可多得的好幫手。

第五章 模組

Python 的模組可以把敘述和函式組織成較大的元件，以檔案儲存之。本章教導讀者如何建立模組、使用模組以及重載模組等等。

第六章 類別

探討 Python 物件導向程式設計 (OOP) 的工具，也就是類別。OOP 對 Python 而言，多用於在互相連結的物件中搜尋名稱。

第七章 例外事件

第一部結束的終曲是 Python 的例外事件，例外事件擺在類別之後介紹是因為例外事件可以是類別。

第二部：進階技巧

第二部強調的重點在於 Python 的實用價值，列舉 Python 提供的內建工具，並以範例說明內建工具的用途。

第八章 內建工具

這一章介紹了許多模組和函式，都歸屬於 Python 的標準工具組，安裝 Python 時，這些工具都會隨附而來，故名為內建工具。瞭解 Python 提供的眾多標準工具，絕對可以節省許多無謂的研發時間。

第九章 常見任務

這一章提供幾支有用的程式範例，利用它們可將第一部學到的紮實基礎以及第八章所討論的內建工具順利的整合應用。本章內容重點是 Python 愛用者最感興趣的文字處理、系統介面以及一些簡單的小任務，千萬別錯過。

第十章 系統架構和應用程式

這一章教導讀者如何建立實際的軟體系統，及使用程式庫 (多指 Python 標準配備的程式庫和協力廠的免費程式庫)。本章的程式是書中最複雜的範例，但是，對實務設計而言，也是最真實的說明案例。我們特地以 JPython 作為最後的一個大型範例，JPython 就是 Java 的 Python，你可以由範例瞭解到 JPython 的強大功能。

第三部：附錄

這本書最後有三篇附錄。附錄 A 列出 Python 現有的資源網站；附錄 B 討論一些和作業平台相關的議題，如 Unix/Linux、Windows 和 Macintosh 的作業平台，這些都是本書內容主要提及的幾種系統；最後的附錄 C 則是各章問題集的解答。三個附錄是很好的補充資料。此外，美商歐萊禮出版的 Python Pocket Reference 一書也是相當優良的課外讀物；網站上也有相當豐富而且免費的 Python 文件資源，詳情請參照網站 <http://www.python.org>。

閱讀需知

其實沒什麼好說的了。這是一本入門書，總不能預期讀者都先成為 Python 的高手，再來讀這本書吧。然而，如果你從未碰過電腦，連開機是什麼都得大費周章好好研究一下，恐怕這本書就不太適合你來閱讀。但是，我們並沒有事先假定讀者一定要有某種程式語言的背景知識才讀得懂這本書，當然，本書的水準也絕對不會太小兒科，有程式寫作經驗的人大可放心購買。Python 能做的事很多，許多看似複雜的事情以 Python 來做都可遊刃有餘，我們希望透過這本書的指引，能讓眾人明白 Python 的這點好處。書中論述 Python 的特性時，會不時的拿 C 和 C++ 語言來做比較，如果你從未聽過 C/C++，大可略過這些論述的文字。

有件事應該稍微提一下：Python 的創始人是 Guido van Rossum，Python 的命名是來自於 BBC 的喜劇影集：Monty Python's Flying Circus，因此書中有些範例會扯到這個節目的內容，例如：“spam”、“eggs”。那部喜劇片究竟在演什麼無需在意，畢竟符號終究只是符號而已。然而，大蟒蛇（python）和 Python 一點關係也扯不上。

最新書訊

本書會在網站上持續不斷的修編，可能是隨 Python 的版本更新做昇級的修正，或者加上更多的註解材料，或者修正錯誤等等。下列各站都可以找到本書最新的消息：

- <http://www.oreilly.com> (美商歐萊禮)
- <http://rmi.net/~lutz> (Mark 的網頁)
- <http://starship.skyport.net/~da> (David 的網頁)
- <http://www.python.org> (Python 的主網站)

我們很希望能具備更敏銳的洞察力，可惜兩位作者都只是平凡的人，網站資源變更的速度實在太快了，印刷文件終究無法跟不上腳步。

書中的範例

本書寫作的依據以及範例的測試都是以 Python 1.5 版為主。然而，由於書中的重點擺在語言的核心部份，我們有理由相信即使未來 Python 有更新的版本問世，書中的內容也不會淘汰得太快，大部分的內容也適用於較早的版本【註】，不過如果你想將新版的延伸套件用在舊的版本那當然是行不通的。總之記住，最新版本就是最好的。本書內容著重於核心語言，因此大部分都適用於 JPython —— 一個以 Java 寫成的 Python。

書中的範例程式以及附錄 C 的解答都可以上網下載：<http://www.oreilly.com/catalog/lpython>。

範例程式怎麼執行？第一章會就此簡介一番。然而，第一步是安裝 Python。最新版的 Python 可從 <http://www.python.org> 下載，你可以下載預先編譯好的 Python，也可以下載原始碼自行編譯。Walnut Greek 有販售 Python 的 CD-ROM，搭配 Linux 發行軟體出售。此外，也可以搭配大本的 Python 手冊一起買。安裝執行檔或者原始碼檔都有完整的說明文件可以參考，我們不用再多費唇舌講解不必要講解的東西（Programming Python 一書有介紹安裝的程序）。

註 嗯，這話應該言之成理。從“Programming Python”一書出版到現在已經有幾年的時間了，Python 語言變更的地方著實不多，而且即使真的有變，通常和舊版也都能相容。幾年下來，Guido 的確添加了不少東西，不過原本就有的，卻很少變動。週邊的工具，諸如 Python/C API 和 Thinkter GUI 介面似乎比較容易改變，只是這本書幾乎不談這些高等的技巧。然而，無論如何，讀者仍然有義務自行上網查詢最新的版本資訊。

與我們連絡

如果你對此書有任何的指教和建議，請把建言寄到：

O'Reilly & Associates
101 Morris Street
Sebastopol, CA 95472
1-800-998-9938
1-707-829-0515
1-707-829-0104 (傳真)

你也可以寄電子信件給我們。如果希望加入通信論壇或者索取書目，請寫信到：
mail@ora.com.tw。

對此書有任何技術上的問題或建言，請寫信到：bookquestions@oreil.ly.com。

誌謝

我們對所有曾參與這本書的人致上最深的謝意，只是限於篇幅，無法在此詳列諸公的大名。特別要感謝我們的編輯 Frank Willison，以及 O'Reilly 的同仁，沒有他們，Python 的叢書無法順利誕生。謝謝 Eric Raymond、Guido van Rossum、Just van Rossum、Andrew Kuchling、Dennis Allison、Greg Ward、Jennifer Tanksley 校閱這本書。當然不能忘記 Guido 以及其他曾為 Python 付出心力的人，謝謝他們創出這麼有用又這麼好玩的一個語言。Python 和其他免費軟體一樣，是許多幕後英雄共同努力的結果。

Mark 的話：

打從撰寫“Programming Python”那時候開始，我便常常遊走各地教導一些 Python 的初學者，除了累計哩程之外，這些課程也幫助我更加認識 Python 的核心，我將這些新的體認寫在書中的第一部。謝謝參與這些課程的學生，他們的回饋是我參與這本書一個很主要的原因。也謝謝 Softronex 今年夏天提供的工作機會（我不可能再拿到那麼高的酬勞了）。

最後，有幾位人士也是功不可沒，需要記上一筆：這本書的另一位作者 David Ascher，由於他的努力不懈以及耐心和毅力，這本書才有可能完成；謝謝我在 Lockheed Martin 的工作夥伴：Linda Cordova、Carl Sagan、Lao Tzu、Denver Broncos。最重要的是，要謝謝我的妻子 Lisa 和小孩 Michael、Samantha、Roxanne，感謝他們的支持，忍受書本撰寫計畫的苦楚。我還欠 Roxanne 一趟 Wally World 之旅。

於科羅拉多

David 的話：

除了上述需要感謝的對象之外，我想把感謝的名單再加長一點。

首先，當然是 Mark Lutz，沒有 Mark 盛情的邀請，我沒有機會參予這本書的編寫，也不可能成為 Python 的訓練技師。另外，幾名 Python 的大師 Guido、Tim Peters、Don Beaudry 以及 Andrew Mullhaupt 等等，我也有說不完的感激之情，沒有他們的提攜和鼓勵，我很難在 Python 語言和週邊工具的貢獻上有所作為。

如同 Mark 一樣，我自己也發展出了一套課程，做為 Python 和 JPython 的教學材料。這些課程的學生幫我再一次澄清了 Python 的一些很戲劇性又很技巧性的部份，很幸運的，這些成份很少，學起來一點也不困難。同時，也讓我時時謹記在心，為什麼 Python 會如此受歡迎，眾人都愛用的原因為何。我很感謝這些學生的回饋作為。我也想感謝幾位特別的人士，沒有他們，這些教學課程是沒有機會問世的：Jim Anderson (Brown University)、Cliff Dutton (Distributed Data System)、Geoff Philbric (Hibbitt、Karlson & Sorensen)、Paul Dubois (Lawrence Livermore National labs) 以及 Ken Swiz (KLA-Tencor)。

感謝我的科學導師 Jim Anderson、Leslie Welch 以及 Norberto Grzywacz。他們都很親切的支援我的努力，不論是 Python 或本書。不單單是瞭解為什麼我要這麼做，而且也完全信賴無所質疑。

我的第一批受難學生每個人都值得獲得一面耐力金牌獎，當時我只有憑藉著一份無以倫比的熱情（引用學生說的話，已經有點走火入魔了），卻無實際的教學經驗：Thanassi Protopapas、Gary Strangman 以及 Steven Finney。Thanassi 對此書的草稿有許多不錯的建議。

最後，我想感謝我的家人：我的父母 JacSue 以及 Philippe，他們總是鼓勵我，要我做我想做的事情；我的兄弟 Ivan，他會提醒我以前碰到的程式設計的問題；我的妻子 Emily，她總是無尤的支持著我，而且認為寫書是一件很神聖的工作。我的兒子 Hugo，總會讓出一點時間讓我有機會使用電腦的鍵盤，而且在這本書的計畫結束之後，學會了把電腦關掉，這也值得嘉許一番吧。當我收到 Mark 的第一封討論到這本書的信時，Hugo 不過才出生3天。現在，Hugo 已經 8 個月大了。這一年半載的日子的確耐人尋味呀。

這本書的讀者也值得感謝。我希望你們能沉浸在讀書的學習樂趣，把 Python 語言融入於生活情境中。Python 帶給我的成果早已不可量估，在整個計算領域？，Python 所發揮的威力遠超乎原本的觀想。我努力寫書的目的就是期望本書的讀者也能領略其中的況味兒。如果您學習 Python 的目的是為了解決某種特定的問題，我希望 Python 對您能完全透明，毫無掛礙，彷彿消失了蹤影一般，您只要全心全意專注在核心問題的解決方案即可。然而，我不知道讀者能否同我一樣一看到 Python，就驚覺出 Python 的諸多優點，進而花費時間學習認同。如果您也是門道中人，請務必瞭解一點，要透澈瞭解 Python 不是短時間就能辦到的。我自己也算不清究竟花了多少時間和 Python 相處。只是，現在的我仍然遊蕩在 Python 的四周圍，仍然能從中得到樂趣。您呢？

於 舊金山

