序

本書是 Python 程式設計語言的入門書。Python 是受歡迎的程式設計語言，可用於各種領域的獨立式程式和描述語言應用程式。Python 免費、可移植、功能強大、而且相當容易使用。

無論你是剛接觸程式設計語言，或者是專業開發人員，本書的目標是讓你快速掌握核心 Python 語言的基礎。讀過本書後，你就會對 Python 有足夠的瞭解，將其應用在你決定要探索的應用領域內。

關於此第三版

自從本書第二版於 2003 年末出版以來的四年內，Python 本身已有實質之變動，而我在 Python 訓練課程中介紹的主題也有所不同。雖然我試著儘可能保留前版文字，但此新版反應 Python 語言和 Python 訓練課程中許多新近之變化，以及一些結構性的改變。

有關本版 Python 語言的變動

針對此語言，本版已全面更新，反應出 Python 2.5 以及從第二版出版以來該語言之所有修改（第二版大部份奠基於 Python 2.2，而且在 2.3 版專案即將發布前，提出一些 2.3 版的功能）。此外，還會討論即將誕生的 Python 3.0 版預期的變動，而且是在合適之處整合進來。以下是一些此語言主要的議題，你將發現在本版中是嶄新的內容或有所延伸的討論：

- 新的 `B if A else C` 條件式運算式（第 12 章）。
- `with/as` 情境管理器（第 27 章）。
- `try/except/finally` 的統一（第 27 章）。
- 相對匯入語法（第 21 章）。

- 生成器運算式（第 17 章）。
- 新生成器函式功能（第 17 章）。
- 函式裝飾器（第 26 章）。
- 集合物件型態（第 5 章）。
- 新內建函式：`sorted`、`sum`、`any`、`all`、`enumerate`（第 4 章和第 13 章）。
- 十進位固定精確度物件型態（第 5 章）。
- 檔案、List comprehension（串列理解式）【譯註】、反覆器等等新的以及延伸的題材（第 13 章和第 17 章）。
- 論及新開發工具：Eclipse、`distutils`、`unittest` 和 `doctest`、IDLE 環境、Shedskin 等等（第 3 章和 29 章）。

較小語言變動會在本書內討論（例如，廣泛使用的 `True` 和 `False`，新的 `sys.exc_info` 以取出例外事件細節，以及字串式例外事件的終止，字串方法，以及 `apply` 和 `reduce` 內建函式）。此外，有些前版中屬於新的功能，也會在此版中有所擴展，包括三限制值切片（**three-limit slice**），以及包括 `apply` 的任意引數呼叫語法。

有關本版 Python 訓練課程的變動

除了這類語言變動外，本版也擴充了我最近幾年開設的 Python 訓練課程中所介紹的新主題和範例。例如，你會發現：

- 有新的一章介紹內建型態（第 4 章）。
- 有新的一章介紹敘述語法（第 10 章）。
- 全新的一章以加強涵蓋的方式，介紹動態定型（第 6 章）。
- 詳細的 OOP 簡介（第 22 章）。
- 檔案、範圍、巢層敘述、類別、例外事件等等更多的新範例。

很多的新增和變動，都是考慮為 Python 初學者所做的，而有些主題則移到於訓練課程中，已證實最易於消化的地方。例如，List comprehension 和反覆器，現在最初出現的地點是和 `for` 迴圈敘述一起，而非和稍後之功能工具一起出現。

譯註　List comprehension（串列理解式）提供了一個製造串列簡潔的方法，而不使用 `map()`、`filter()` 或 lambda 形式。其結果也比使用其他方法做出串列清楚易懂。list comprehension 通常是一個運算式跟著一個 `for` 迴圈的敘述，然後是零個或多個 `for` 或 `if`。其傳回的串列是一個由在 `for` 及 `if` 條件下執行運算式的結果。

你也會發現，很多原本核心語言主題的討論在版本中有了實質的擴展，新增了範例和新的討論內容。因為本書已成為學習核心 Python 語言的標準資源，介紹時就必須更完整，配合新的用例。

此外，整版也整合新的 Python 技巧和竅門集，這是過去 10 年從課堂上授課，以及過去 15 年以 Python 從事實際工作所收集而來的。練習題已做了更新和擴充，以反應當前 Python 最佳實務、新語言功能、以及課堂中所見初學者常犯之錯誤。總之，此版討論的核心語言內容比前版更多，不但是因為 Python 變得更大，也是因為我加了一些在實務上證實為相當重要的脈絡資訊。

版本的結構性變動

如同前版，為了配合本書現在更為完整的事實，題材已切割成能吞得下去的份量。也就是說，我把核心語言題材組織成許多章次，讓題材更易於應付。例如，型態和敘述現在是兩個頂層部份，每種主要型態和敘述主題都有一章。這種新的結構的設計，是為了讓這本書多講一點，但又不會讓讀者心生恐懼。在此過程中，練習題和「陷阱」（常犯錯誤）則移到每章結尾和每卷結尾；現在則出現在每卷上一章的結尾處。

在此第三版中，我也擴增了每卷結尾處的練習題以及每章結尾之摘要以及每章結尾之習題，以協助你在做完那些題目後，可以達到複習那些章次的效果。每章的最後都有一組問題，協助你複習和測試你對該章題材的理解。和每卷結尾練習題不同的是（解答在附錄 B），每章結尾習題的解答就在問題之後；即使你確定你已正確回答問題，我還是鼓勵你去看一看解答，因為答案本身就是一種複習。

儘管有這些新主題，本書依然是以 Python 初學者為主，其設計就是要作為程式設計師的首選 Python 書籍【註】。本書保留前兩版的許多題材、結構、以及焦點。在適當之處，我會為初學者擴充簡介內容，同時把較高階之新主題和討論主線隔離出來，以避免模糊基礎內容。再者，因為本書絕大部份是根據經過時間檢驗的訓練課程經驗、以及題材而寫成的，就像前兩版那樣，依然可作為 Python 入門課程之用，以自訂教學步調。

本版範圍變動

第三版是打算作為核心 Python 語言的教學書籍，除此無他。本書是有關深入學習這個語言，然後才是應用層次程式設計的應用。此處的介紹是由下往上以及漸進式的，但是，會完整討論整個語言，獨立於應用角色之外。

註　所謂的「程式設計師」，我是指過去以任何程式語言或描述語言寫過一列程式碼的任何人。如果這樣不包括你在內，你可能還是會覺得這本書有點用，但是，要注意一點，本書會花很多時間教導Python，而非談論程式設計之基礎。

對某些人而言，「學習 Python」就是花一兩個小時在網站上看一看教學文件。對於高階程式設計師而言，這樣多少行得通，畢竟，和其他語言相比，Python 真的簡單許多。這種走夜路手法的問題在於，實踐家最終會在不尋常情況跌倒而被卡住：變數自己變了，可變更預設引數值的變化難以理解等等。此處的目標是提供 Python 基礎之紮實地基，使得即使不尋常情況發生時，也能有所理解。

這種範圍得深思熟慮才行。我們把焦點鎖定在語言基礎，就能以令人滿意的深度研究這些主題。其他書籍則是挑選本書遺漏的，對應用層次主題和參考材料提供更完整的說明，諸如 O'Reilly 的《Programming Python》、《Python Cookbook》、《Python in a Nutshell》、以及《Python Pocket Reference》。本書的目的是全力教授 Python 本身，使你能將其應用至你碰巧在從事的任何領域上。

因為焦點有所改變，前版中有些參考資料和更高階之節次（大約前版之 15%）已被去掉，以擴充核心語言之節次。如此一來，本版的讀者會發現核心語言有更完整之說明，使得此書變成更有用的首選 Python 書籍。本版中，有加入一些較高階之範例，作為自學之程式，當作最終之練習（參考第 29 章）。

關於本書

本節強調本書的一般性重點，和本書之版本無關。沒有一本書可以滿足每位可能的讀者，所以，瞭解本書的目標何在，這一點很重要。

事前準備

事實上，沒有什麼非要不可的事前準備可講。初學者和硬底子的程式設計老兵都可成功運用此書。如果你打算學 Python，這本書可能就適合你。不過，一般而言，我發現在本書之前有任何程式設計或描述經驗，則會有所幫助，但是，並非每位讀者都得如此。

本書的設計是作為程式設計師的入門 Python 書籍。從來沒碰過電腦的人，可能就不適合（例如，我們不會花時間討論電腦是什麼東西），但是，我沒有假定你要有任何程式設計背景或教育。

另方面，我不會假定讀者是「白癡」來侮辱讀者。以 Python 來做些有用之事，這很容易，而本書就是要教你該怎麼做。本書偶爾會拿 Python 和 C、C++、Java、以及 Pascal 這些語言做比較，但是，如果你過去沒用過這類語言，當作沒看見也無所謂。

本書的範圍和其他書籍

雖然本書論及 Python 語言所有基本事項，但為了速度和篇幅考量，我還是把範圍縮小了。為了讓事情保持簡單，本書的焦點是核心概念，使用小型而自足的範例以示範重點，以及偶爾省略可在參考手冊中找到的小細節。因此，本書最佳之註腳應該是入門書，以及可前往更高階和完整的書籍的踏腳石。

例如，我們不會談太多 Python/C 之整合，這個複雜主題顯然是許多以 Python 為基礎之系統的重心所在。我們也不會談太多 Python 的歷史或發展過程。此外，受歡迎的 Python 應用程式也只簡單看一看而已，諸如 GUI、系統工具、以及網路描述機制，而有的則根本不談。想當然爾，這樣的範圍會漏掉大藍圖的一些部份。

大體而言，Python 是為了讓描述語言世界的「品質棒」再往上走幾個刻度，而有些觀念需要的背景環境，不是此處能道盡的，如果我沒有建議你讀完此書後的進修書籍，那就是我的疏忽了。我希望本書多數讀者最終都可以繼續走下去，從其他書籍完整瞭解應用層次之程式設計。

因為本書的焦點是初學者，設計上自然是和 O'Reilly 其他 Python 書籍互補。例如，我所寫的另一本書《Programming Python》，提供更大以及更完整範例，還有應用程式設計技巧的教學，而且刻意設計成讀完此書後的下一本書。粗略而言，本書和《Programming Python》當前之版本反應出作者訓練題材的兩部份：核心語言以及應用程式設計。此外，O'Reilly 的《Python Pocket Reference》也是尋找本書跳過之一些細微細節的快速參考手冊。

其他接續之書籍也可提供參考資料、其他範例、或者於特定領域內（諸如網站和 GUI）使用 Python 的細節。例如，O'Reilly 的《Python in a Nutshell》以及 Sams 的《Python Essential Reference》有提供參考資料，而 O'Reilly 的《Python Cookbook》替那些已熟識應用程式設計技巧的人，提供自足範例庫。因為書籍是很主觀的經驗，我鼓勵你自行瀏覽，以尋找適合你需求的進階書籍。不過，無論你選哪些書籍，要記住，Python 其餘故事所需研究的範例，都相當具有實務性，不是這裡的空間所能容納的。

即使這麼說，我想你會發現本書是 Python 很好的入門首選，儘管範圍有限（也許也正是因為如此）。你會學到起步所需之一切事物，以撰寫有用的獨立式 Python 程式和描述稿。當你讀完本書時，不僅學會語言本身，也學會如何將其妥善運用在你每日的任務上。此外，當你碰上時，會有能力去應付那些更高階的主題和範例。

本書的風格和結構

本書是依據三天期 Python 實用課程的訓練題材寫成的。你會在每章尾端看見每章結尾練習題，以及每卷最後一章尾端發現每卷尾端練習題。每章習題的解答就在每章裡面，而每卷練習題的解答則是在附錄 B。習題的設計是讓你複習題材，而練習題的設計是讓你以正確方式撰寫程式碼，而且通常也是該課程的重點之一。

我極力建議你去做一做習題和練習題，不只是為了獲得 Python 程式設計經驗，也是因為有些練習題會引出本書沒談及的議題。如果你碰上難關，那些章次和附錄 B 的解答，應該可以協助你（你也可以儘量偷看那些解答）。

本書的整體結構也是衍生自課堂材料。因為本書的設計是要快速介紹語言基本，我是以語言主要功能來介紹，而不是以範例為主。此處，我們採取由下至上的手法：從內建物件型態，到敘述，再到程式單元等等。每章都相當自給自足，但是，後續的章次會依賴前面章次所介紹的觀念（例如，我們談到類別時，我假定你已知道如何撰寫函式），所以，循序漸進，對多數讀者而言，應該最有意義。

一般而言，本書以由下至上的方式介紹 Python 語言。其組織是一卷談一種主要語言功能（型態、函式等等），而多數範例都很小，而且自給自足（有些人可能會說本書的範例都太做作，但是，他們是為了說明重點而設計的）。更明確的講，以下是你會看到的內容：

卷 1，入門簡介

我們一開始先概論 Python，以回答常問的一些問題：為什麼要使用這個語言、有何用處等等。首章介紹此技術底下的主要想法，讓你有些背景環境。然後，就是本書技術題材，我們會探索和 Python 執行程式的方式。此卷的目標是讓你有足夠的資訊，以追尋後續範例和練習題的腳步。

卷 2，型態和運算

接著，我們開始 Python 語言之旅，深入研究 Python 的主要內建物件型態：數字、串列、辭典等等。單用這些工具，就能以 Python 做很多事了。這是本書最實質的一卷，因為我們在此替後續章次打下地基。我們也會在此卷談動態定型及其參照值：這是善用 Python 的關鍵。

卷 3，敘述和語法

下一卷開始介紹 Python 的敘述：你輸入的程式碼，會在 Python 之中建立和處理物件。此外，我們也會介紹 Python 的一般語法模型。雖然此卷焦點為語法，但也會介紹相關工具，諸如 PyDoc 系統，並探索其他一些撰碼法。

卷 4，函式

本卷開始討論 Python 較高階程式結構工具。函式是打包程式碼以便再利用，並避免程式碼餘贅的簡單方式。在此卷中，我們要探索 Python 的範圍規則、引數傳遞技術等等。

卷 5，模組

Python 模組讓你組織敘述和函式，以變成較大之元件，而此卷會說明如何建立、使用、以及重載模組。我們也會在此看一些更高階之主題，諸如模組套件、模組重載、以及 __name__ 變數。

卷 6，類別和 OOP

此處，我們探討 Python 的物件導向程式設計（OOP）工具：類別。類別是選用的，但卻是組織程式碼以利量身打造和再利用的強大工具。你會知道，類別幾乎都是重覆利用我們在此書論及的觀念，而 Python 的 OOP 多數就是在連結物件內尋找名稱。你也會知道，OOP 在 Python 之中是選用的，但是，可以大幅削減開發時間，尤其是長期策略性專案開發。

卷 7，例外事件和工具

本書最後一卷是討論 Python 例外事件處理模型和敘述，外加簡介開發工具（當你開始撰寫較大程式時，就會有用處，例如，除錯和測試工具）。這一卷擺在最後，例外事件現在應該都是類別了。

卷 8，附錄

本書結尾是兩份附錄，論及在各種電腦上使用 Python 的平台特定技巧（附錄 A），以及提供每卷尾端練習題的解答（附錄 B）。每章結尾習題的解答就在該章之後。

注意到，索引和目錄可用於尋找細節，但本書沒有參考文獻附錄（本書是教學書籍，而非參考書）。如前所述，你可以參考《Python Pocket Reference》（O'Reilly）還有其他書籍，另外，免費的 Python 參考手冊就在 *http://www.python.org*（瞭解語法和內建工具細節）。

書籍更新

改進會持續進行（打錯字也是）。如果有本書的更新、補充、以及更正會放在下列網站之一：

http://www.oreilly.com/catalog/9780596513986/（O'Reilly 本書之網頁）

http://www.rmi.net/~lutz（作者的網站）

http://www.rmi.net/~lutz/about-lp.html（本書作者之網頁）

這三個 URL 的最後一個是指向本書之網頁，我會在此張貼更新，但是，如果鏈結失效了，一定要搜尋資訊網。如果我更有洞察力，我會盡力，但資訊網改變得比印刷書籍還要快。

關於本書的程式

本書和其內所有程式範例都是根據 Python 2.5 版。此外，雖然我沒有試著描繪未來，但是，一路上我會討論期待中的 3.0 版的一些觀念。

然而，因為本書的焦點是核心語言，你可以相當肯定，多數內容，在未來 Python 版本中，不會有太多變化。本書多數內容也適用早期 Python 版本，除非不適用：當然，如果你試著使用在你所用版本之後才新增的延伸功能，那當然行不通啦。

原則就是，最新的 Python 就是最好的 Python。因為本書焦點是核心語言，多數內容也適用第 2 章提及的 Jython（Java 版的 Python 語言實作品）以及其他 Python 實作品。

本書範例的原始碼以及練習題解答都可從本書網站取得：http://www.oreilly.com/catalog/9780596513986/。那麼，你怎麼執行範例？我們會在第 3 章研究啟動細節，所以，要有點耐心，以瞭解細節。中文版在 http://www.oreilly.com.tw/product2_tip.php?id=a240

準備迎接 Python 3.0

Python 3.0 首次 alpha 版就在本書寫好還沒出版前問世。正式來講，本書此版是根據 Python 2.x 線的版本（明確講，就是 2.5 版），但是，也增加了很多有關 Python 3.0 版即將來臨的修改說明。

本書出版後大約一年內，3.0 版不會正式推出，而且至少兩年內，也不會廣為使用。然而，如果你在 3.0 廣為使用後挑上此書，本部份提供了，你將碰到此語言的一些修改的簡潔說明，以協助你渡過版本差異。

雖然會有些例外，但 3.0 版 Python 語言的多數部份都會和本書所說明的相同，而且對典型和實務應用程式的衝擊也會比較小。也就是說，本書介紹的 Python 基礎不會隨版本而異，而讀者處理版本特定細節前，先研讀這些基礎，自然能有所受益。

不過，為了協助你未來轉換程式碼，下列清單列出 Python 3.0 主要的差異。本版章次之指標，不是討論這些修改，就是指出受到這些修改的衝擊，因此，清單有配合相關之章次。有些修改已可在 Python 2.5 之內撰碼，但有些則不行。因為此時下列清單對多數讀者而言都沒有意義，我建議你先研讀本書，以學習 Python 基礎，然後再回來看這份清單以瞭解 3.0 之中修改。Python 3.0 之中：

- 當前的 `execfile()` 內建函式已被移除；改用 `exec()`（參考第 3 章）。
- `reload()` 內建函式會被移除；另一個替代函式尚未知（參考第 3 章和第 19 章）。
- `'X'` 倒引號字串轉換運算式已被移除；使用 `repr(X)`（參考第 5 章）。
- `X <> Y` 多餘的不相等運算式已被移除：使用 `X != Y`（參考第 5 章）。
- 集合能以新的實字語法 `{1, 3, 2}` 建立，對等於當前的 `set([1, 3, 2])`（參考第 5 章）。

- 集合理解式（set comprehension）能寫成：`{f(x) for x in S if P(x)}`，相當於當前生成器運算式：`set(f(x) for x in S if P(x))`（參考第 5 章）。
- 真實除法已支援：`X / Y` 一定會傳回保留餘數的浮點數，即使是整數亦然；使用 `X // Y` 以啟用當前的截短除法（參考第 5 章）。
- 只有一種整數型態 `int`，支援當前長整數型態之任意精確度（參考第 5 章）。
- 八進位和二進位實字：當前八進位值 `0666` 變成錯誤：改用 `0o666`。而 `oct()` 函式的結果也會據以修改：`0b1010` 現在等於 `10`，而 `bin(10)` 會傳回 `"0b1010"`（參考第 5 章）。
- 字串型態 `str` 支援寬字元 Unicode 文字。新的 `bytes` 型態會代表短字元字串（例如，以二進位模式從檔案載入時）；`bytes` 是小整數構成之可變更序列，和 `str` 之介面有點不同（參考第 7 章）。
- 有個新的、選用的字串格式技術：`"See {0}, {1} and {foo}".format("A", "B", foo="C")` 會變成 `"See A, B and C"`（參考第 7 章）。
- 辭典 `D.has_key(X)` 方法會被移除：改用 `X in D` 成員關係（參考第 4 章和第 8 章）。
- 混合非數值型態之比較（經由代理之排序函式）會引發例外事件，改用當前任意但固定之跨型態次序（參考第 8 章和第 9 章）。
- 辭典方法 `.keys()`、`.items()`、以及 `.values()` 會傳回類似可繞行之「檢視」物件，而非串列；有需要時，使用 `list()` 以強迫建立串列（參考第 8 章）。
- 因為前一項，這種撰碼樣式不能再使用：`k = D.keys(); k.sort();`，要改用 `k = sorted(D)`（參考第 4 章和第 8 章）。
- `file()` 內建函式會被移除；改用 `open()`（參考第 9 章）。
- `raw_input()` 內建函式會被更名為 `input()`；使用 `eval(input())` 以得到今日之 input() 函式的行為（參考第 10 章）。
- `exec` 程式碼字串執行敘述會再次變成內建函式（參考第 10 章）。
- `as`、`with`、以及 `nonlocal` 會變成新保留字；因前一項，`exec` 不再保留（參考第 11 章）。
- `print` 變成函式以支援更多功能，不再是敘述：使用 `print(x, y)`，而非 `print x, y`，以及使用新函式的關鍵字引數，自訂出列印行為：`file=sys.stdout, sep=" "`、以及 `end="\n"`（參考第 11 章）。
- 擴充之可繞行物件解開機制：支援通用序列指定之敘述現在可以用了，諸如 `a, b, *rest = some_sequence`，如同 `*rest, a = stuff`；於是，指定敘述左右兩側的項目數不必吻合（參考第 11 章）。

- 函式經由序列指定以讓 tuple（元組）參數自動解開的機制移除了；你無法再寫 `def foo(a, (b, c)):`，而必須使用 `def foo(a, bc): b, c = bc` 以明確進行序列指定（參考第 11 章）。
- 當前 `xrange()` 內建函式更名為 `range()`；也就是說，只有 `range()` 了（參考第 13 章）。
- 在反覆器協定中，`X.next()` 方法更名為 `X.__next__()`，而新的內建函式 `next(X)` 會對物件呼叫 `X.__next__()` 方法（參考第 13 章和第 17 章）。
- 內建函式 `zip()`、`map()`、以及 `filter()` 會傳回反覆器；使用 `list()` 以強迫建立串列結果（參考第 13 章和第 17 章）。
- 函式可以包括註解說明引數和結果（選用）：`def foo(x: "spam", y: list(range(3))) -> 42*2:` 只會在執行期貼附至函式物件的 `foo.func_annotations` 辭典屬性：`{'x': "spam", 'y': [0, 1, 2], "return": 84}`（參考第 15 章）。
- 新的 `nonlocal x, y` 敘述可以讓你擁有封閉函式範圍內變數之寫入權（參考第 16 章）。
- `apply(func, args, kws)` 函式會被移除；改用 `func(*args, **kws)` 呼叫語法（參考第 16 章和第 17 章）。
- `reduce()` 內建函式會被移除；以本書所示之方式撰寫迴圈；`lambda`、`map()`、以及 `filter()` 在 3.0 之中保留（參考第 17 章）。
- 所有匯入預設成為絕對的，而且會跳過套件自己的目錄：使用新語法 `from . import` *name* 以啟用當前相對之匯入（參考第 21 章）。
- 所有類別都會是新形式之類別，而且會支援當前新形式之延伸（參考第 26 章）。
- 當前新形式類別所需要的類別 `Spam(object)` 衍生將不再需要；在 3.0 之中，當前獨立式之「傳統」和衍生之「新形式」類別都會自動視為當前所謂之新形式（參考第 26 章）。
- 在 `try` 敘述中，`except name, value:` 形式，將變成 `except name as value:`（參考第 27 章）。
- 在 `raise` 敘述中，`raise E, V` 必須寫成 `raise E(V)`，以刻意製作實體（參考第 27 章）。
- 本書說明的 `with/as` 例外事件環境管理器功能已啟用了（參考第 27 章）。
- 所有用戶定義和內建之例外事件都視為類別，不再是字串（參考第 28 章）。

- 用戶定義之例外事件必須衍生自內建之 `BaseException`，也就是例外事件階層之根（`Exception` 是其子類別，足以作為你的根類別）；內建的 `StandardException` 類別已被移除（參考第 28 章）。
- 標準程式庫的套件結構會實質重組（參考 Python 3.0 版的附註）。

雖然這份清單乍看之下有點嚇人，但記住，本書說明的多數核心 Python 語言在 3.0 之中都會維持不變；事實上，上列多數內容都是相當細微的枝節，不會對程式設計師產生太多衝擊。

此外，注意到，本書撰寫時，這份清單依然存疑，最終可能不完整也不準確，所以，一定要看看 Python 3.0 版本附註，以瞭解實情。如果你替 Python 2.x 線版本撰寫程式碼，也可以看一看 2to3 這個自動把 2.x 轉成 3.0 的程式碼轉換命令稿（Python 3.0 會提供）。

關於此系列

O'Reilly 學習手冊書籍，是針對任何想採取結構化手法學習以養成新技能的人，所撰寫和設計的。本系列的每本書籍都利用我們（透過你們的協助）認為最佳之學習原理，讓你具備參與新專案的知識，以應付來自你的經理的意外指派，或者迅速學習新語言。

要善用學習手冊系列的任何一本書，我們建議你循序讀完每一章。你會發現，閱讀每章標題，就可立刻掌握該章之內容。你也可以使用摘要，事先看一看每章的關鍵重點，並複習你已學過的事物。最後，為了協助你精通每一章的題材，我們還搭配了腦力激盪一節，包含習題在內。每卷也有實用之練習題。

學習手冊書籍會伴隨在你身旁，一如你從值得信賴之同事或導師那裡所期待的那樣，我們努力要讓你的學習經驗更為愉快。我們表現得如何，請讓我們知道，無論是讚賞、批評、或改進之建言，都可寄到 *learning@oreilly.com*。

使用程式範例

本書是協助你把工作做好。一般而言，你可以在程式和說明文件中使用本書的程式碼。你不需要連絡我們取得許可，除非你是要重製程式碼大量的部份。例如，撰寫程式，使用本書好幾段程式碼，就需要許可。販賣和散佈 O'Reilly 範例光碟片也需要許可。引用此書和範例程式以回答問題，不需要許可。把本書大量範例程式整合至你的產品說明文件則需要許可。

雖然並非必要，但表明歸屬權，我們會很感激。歸屬權通常包括標題、作者、出版商、以及 ISBN。例如：「Learning Python, Third Edition, by Mark Lutz. Copyright 2008 O'Reilly Media Inc., 978-0-596-51398-6.」。

如果你覺得你對程式範例的運用超出合理使用或者上列許可狀況之外，可以連絡我們：*permissions@oreilly.com*。

印刷體裁

本書使用下列印刷體裁：

斜體

用於電子信箱、URL、檔名、以及路徑。

定寬體

用於檔案內容以及命令輸出，表明模組、方法、敘述、以及命令。

定寬粗體

用於程式節段中，以顯示應該由用戶輸入的命令或文字，而偶爾則用於強調程式碼的一些部份。

定寬斜體

用於程式節段中可替換部份以及一些註解。

< 定寬體 >

指出應該以真實程式碼取代的語法單元。

指出和附近本文相關之技巧、建言、或一般註解。

指出和附近本文相關之警言和注意事項。

本書範例中，系統命令列開頭的 % 字元指的是系統提示符號，這得視你的機器而定（例如，DOS 視窗是 `C:\Python25>`）。不要自行輸入 % 字元。同樣地，在直譯器互動列表中，也不要輸入文字列開端的 >>> 和 … 字元；這些是 Python 顯示的提示符號。只要輸入這些提示符號之後的文字就行了。為了協助你記住此事，本書中，用戶輸入都以粗體顯示。此外，你一般也不需要輸入列表中以 # 開頭的文字；學了之後，你就會知道，這些是註解，不是可執行之程式碼。

建議與問題

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，應該再加進來的其他主題。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　（書籍內容的問題）

與本書相關的線上資訊，假如有勘誤、範例程式、相關連結等，可參訪：

原文書

http://www.oreilly.com/catalog/9780596513986

中文書

http://www.oreilly.com.tw/product2_tip.php?id=a240

誌謝

當我在 2007 年寫本書第三版時，我一直有種「任務完成」的心態。我已經使用和提倡 Python 有 15 年了，而且也教 Python 教了 10 年了。儘管時間和世事流逝，我依然驚訝於 Python 這些年來的成功。Python 的成長，是我們多數人在 1992 年時難以想像的。所以，也許我得冒著被當成沒救、自言自語的作者的風險，但是，你要體諒一下，我得在這裡說一些回憶、恭賀、以及感謝之語。

這是冗長而蜿蜒的道路。今日回首，當我在 1992 年首次發現 Python 時，我根本不知道那對我將來 15 年的生活會有什麼衝擊。1995 年撰寫《Programming Python》初版後的兩年，我開始在全國和全世界旅行，對初學者和專家教授 Python。自從於 1999 年完成本書初版後，我變成全職、獨立的 Python 訓練師和作家；這得感謝 Python 指數式成長的歡迎度。

我在 2007 年中寫下這些話時，已經寫了九本 Python 書籍；我已經教授 Python 超過 10 年了，而且在美國、歐洲、加拿大、以及墨西哥教過 200 個 Python 短期訓練課程，一路上碰過 3,000 位以上的學生。除了時常累計飛行哩程外，這些課程也協助我精煉本書以及其他 Python 書籍的內容。幾年下來，教學磨練了書籍，而書籍又磨練了教學。事實上，你在讀的這本書大部份都是衍生自上課內容。

因此，我很感謝過去 10 年來參加我的課程的所有學生。除了 Python 本身有變之外，你們的意見在形塑本書上，也扮演重大角色（沒有看著 3,000 位學生重覆犯下初學者錯誤更具有啟發性的事了！）。本版的修正主要歸因於 2003 年之後的課程，不過，從 1997 年起以來的每堂課，都對本書之精煉有或多或少之協助。我要特別感謝那些在都伯林、墨西哥市、巴賽隆納、倫敦、艾得蒙頓、以及波多黎各開課的那些客戶；我無法想像更好的地點了。

我也想對每位參與本書製作的人表達感謝。參與此專案的編輯：本版的 Tatiana Apandi，以及前幾版的許多人。感謝 Liza Daly 負責本書技術審查。感謝 O'Reilly 讓我有這個機會寫出這九本書；真的很有趣（感覺上有點像電影《今天暫時停止》）。

我想感謝最初的共同作者 David Ascher 對本書前幾版的協助。David 在前幾版中貢獻「外層」部份，但是在本版中，為了讓新核心語言材料有多一點的空間，我們忍痛割捨了。我在本版中加了一些更高階的程式，以作為自行研讀的最終練習，但是，這無法補償被裁掉的所有題材。如果你對那些材料無法忘懷，可以參考序文中先前有關應用層次內容的相關說明。

做了這麼有趣和有用的語言，我得特別感謝 Guido van Rossum 和 Python 社群其餘之人。如同多數開源碼系統，Python 是許多英雄心血的成果。即使有 15 年的 Python 程式設計經驗，我還是覺得 Python 很有趣。我的特權就是看著 Python 從描述語言的小孩成長成廣為使用之工具，幾乎每個撰寫軟體的機構都以某種方式在運用。這是令人興奮的成果，我想感謝和恭賀整個 Python 社群，做了很棒的事。

我也想感謝我在 O'Reilly 原本的編輯：已故的 Frank Willison。這本書大部份都是 Frank 的想法，反應出他那會感染人群的願景。回顧時，Frank 對我的職涯和 Python 本身都有很深的衝擊。Python 剛出來時能有這麼多樂趣和成功，可以說都是 Frank 的功勞，這一點都不誇張。我們還是很想念他。

最後，還有些人值得感謝。感謝 OQO 這麼好的玩具。感謝已故的 Carl Sagan，讓來自威斯康新州的 18 歲小伙子得到啟發。感謝 Jon Stewart 和 Michael Moore 這些愛國者。感謝我這幾年遇過的所有大型公司，提醒我，我有多麼幸運，可以自己當老闆。

感謝我的小孩 Mike、Sammy、以及 Roxy，無論他們將來選擇要做什麼。我開始用 Python 時，你們都還小，而你們似乎也這樣長大了；我以你們為榮。生活會逼迫我們一路走下去，但總是有回家的路。

而我最要感謝 Vera，我最好的朋友，女友，以及老婆。我最好的日子就是我終於遇見你的那一天。我不知道接下來 50 年會怎樣，但我知道，我想把所有時間都用來擁抱你。

—Mark Lutz
Berthoud, Colorado
2007 年 7 月