前言

這本書是多年來我對專業程式員所做的 C++ 教學課程下的一個自然產物。我發現，大部份學生在一個星期的密集訓練之後，即可適應這個語言的基本架構，但要他們「將這些基礎架構以有效的方式組合運用」，我實在不感樂觀。於是我開始嘗試組織出一些簡短、明確、容易記憶的準則，做爲 C++ 高實效性程式開發過程之用。那都是經驗豐富的 C++ 程式員幾乎總是會奉行或幾乎肯定要避免的一些事情。

我最初的興趣在於整理出一些可被某種「lint-like 程式」施行的規則，最後我甚至領導一個計劃，研究某種可將 C++ 原始碼中違反使用者指定條件之處檢驗出來的工具[1]。不幸的是在我尚未完成其完整原型之前，這個研究計劃便結束了。幸運的是，目前市面上已有這類 C++ 檢驗工具（商品），而且不只一個。

雖然我最初的興趣是在研究可被（某種工具）自動實施的程式設計準則，但我很快瞭解到那個研究方向的侷限性。優秀的 C++ 程式員所奉行的準則，多數都難以「公式化」；要不就是雖然它們有許多重要的例外情況，卻被程式員盲目地奉行不渝。這使我念頭一轉：某些東西雖然不比電腦程式精準，但仍能比一本泛泛的 C++ 教科書更集中火力，更打到重點。這個念頭的結果就是你手上這本書：一本內含 50 個有效建議（如何改善你的 C++ 程式技術和你的設計思維）的書。

在這本書中，你會發現一些忠告，告訴你應該做些什麼，爲什麼如此；告訴你不應該做些什麼，又爲什麼如此。基本而言當然 whys 比 whats 更重要，但檢閱一

[1] 你可以在 Effective C++ 網站上找到此研究的一份概要報告。

列列準則，也確實比強記一本或兩本教科書更輕鬆更方便得多。

和大部份的 C++ 書籍不同，我的組織方式並非以語言特性做爲依據。也就是說我並不在某處集中討論 constructors（建構式），在另一處集中討論 virtual functions（虛擬函式），又在第三個地方集中討論 inheritance（繼承機制）。不，不是這樣，本書的每一個討論主題都剪裁合度地以一個個準則陳列出來。至於我對某特定語言性質的探討，散佈面積可能涵蓋整本書。

這種作法的優點就是比較容易反映出「特意挑選 C++ 做爲開發工具」的那些軟體系統的複雜度。在那些系統之中，光只瞭解個別語言特性是不夠的。例如，有經驗的 C++ 程式員知道，瞭解 inline 函式和瞭解 virtual destructors，並不一定表示你瞭解 inline virtual destructors。身經百戰的開發人員都認知到，理解 C++ 各個特性之間的互動關係，才是有效使用這個語言的最重要關鍵。本書組織反映出這一基本事實。

這種作法的缺點是，你恐怕必須前後交叉參考而非只看一個地方，才能發現我所說的某個 C++ 架構的全貌。爲了將不方便性降至最低，我在書中各處放了許多交叉索引，書後並有一份涵蓋全部範圍的索引。（譯註：爲了協助讀者更容易掌握 *Effective C++* 和 *More Effective C++* 二書，我以 *Effective C++ CD* 爲本，爲兩書的中文版額外加上兩書之間的交叉索引。此乃原書所無。如果文中出現像條款 M5 這樣的參考指示，M 便是代表 *More Effective C++*）

籌劃第二版期間，我改寫此書的雄心一再被恐懼所取代。成千上萬的程式員熱情擁抱 *Effective C++* 第一版，我不希望破壞吸引他們的任何東西。但是自從我寫了第一版之後，六年過去了，C++ 有了變化，C++ 程式庫有了變化（見條款 49），我對 C++ 的瞭解也有了變化，乃至於 C++ 的用途也有了變化。許許多多的變化。對我而言，重要的是我必須修訂 *Effective C++* 以反映那些變化。我嘗試一頁一頁地修改內容，但是書籍和軟體十分類似，局部加強是不夠的，唯一的機會就是系統化地重寫。本書就是重寫後的結果：*Effective C++* 2.0 版。

熟悉第一版的讀者，可能有興趣知道，書中的每一個條款都經過重新檢驗。然而我相信第一版的結構至今仍是流暢的，所以整本書的結構並沒有改變。50 個條款

中，我保留了 48 個，其中某些標題稍有變化（附隨的討論內容亦復如此）。退休下來（被取代的）兩個條款是 32 和 49，不過原條款 32 的許多資訊被我移到如今煥然一新的條款 1 中。我將條款 41 和 42 的次序做了對調，因爲這樣比較能夠適當呈現它們修訂後的內容。最後，我把上一版繼承體系圖所採用的箭頭方向顛倒過來，以符合目前幾乎已經一致的習慣：從 derived classes 指往 base classes。我的 *More Effective C++* 一書也採用相同習慣（本書最後列有該書摘要）。

本書提供的準則，離鉅細靡遺的程度還很遠，但是完成一個好的準則 — 一個幾乎可於任何時間應用於任何程式的準則，動手遠比動嘴困難得多。如果你知道其他準則，可以協助撰寫有效的 C++ 程式，我非常樂意聽到你告訴我它們的故事。

此外，說不定你會覺得本書的某些條款不適合成爲一般性忠告；或許你認爲另有比較好的方法來完成書中所說的任務；或許你認爲某些條款在技術討論方面不夠清楚，不夠完全，抑或有誤導之嫌。我衷心盼望你也能夠讓我知道你的這些想法。

Donald Knuth（譯註：經典書籍 *The Art of Computer Programming*, Volume I,II,III 的作者）長久以來爲挑出其書錯誤的熱心讀者準備有一份小小的報酬。這個故事傳爲美談。追求完美的精神令人佩服。看過那麼多倉促上市錯誤壘壘的 C++ 書籍後，我更是特別強烈地希望踵隨 Knuth 的風範。因此，如果有人挑出本書的任何錯誤並告訴我 — 不論是技術、文法、錯別字、或任何其他東西 — 我將在本書新刷的時候，把第一位挑出錯誤的讀者大名加到致謝名單中。

請將你的建議、你的見解、你的批評、以及（喔…真糟…）你的臭蟲報告，寄至：

```
Scott Meyers
c/o Publisher, Corporate and Professional Publishing
Addison Wesley Longman, Inc.
1 Jacob Way
Reading, MA 01867
U. S. A.
```

或者傳送電子郵件到 `ec++@awl.com`。

我維護有本書第一刷以來的修訂記錄，其中包括錯誤更正、文字修潤、以及技術更新。你可以從 Effective C++ 網站取得這份記錄。如果你希望擁有這份資料，但無法上網，請寄申請函到上述地址，我會郵寄一份給你。

Scott Douglas Meyers

Stafford, Oregon
July 1997