目 錄

譯序（侯捷）　v

目錄（Contents）　vii

前言（Preface）　xiii

致謝（Acknowledgments. 中文版略）　xvii

導讀（Introduction）　001

改變舊有的 C 習慣（Shifting from C to C++）　013

條款 1：儘量以 `const` 和 `inline` 取代 `#define`　013
Prefer `const` and `inline` to `#define`.

條款 2：儘量以 `<iostream>` 取代 `<stdio.h>`　017
Prefer `<iostream>` to `<stdio.h>`.

條款 3：儘量以 `new` 和 `delete` 取代 `malloc()` 和 `free()`　019
Prefer `new` and `delete` to `malloc` and `free`.

條款 4：儘量使用 C++ 風格的註解型式　021
Prefer C++-style comments.

記憶體管理（Memory Management）　022

條款 5：使用相同型式的 `new` 和 `delete`　023
Use the same form in corresponding uses of `new` and `delete`.

條款 6：記得在 destructor 中以 `delete` 對付 pointer members　024
Use `delete` on pointer members in destructors.

條款 7：為記憶體不足的狀況預做準備　025
Be prepared for out-of-memory conditions.

條款 8：撰寫 `operator new` 和 `operator delete` 時應遵行公約 033
Adhere to convention when writing operator new and operator delete.

條款 9：避免遮掩了 `new` 的正規型式 037
Avoid hiding the "normal" form of `new`.

條款 10：如果你寫了一個 `operator new`，
請對應也寫一個 `operator delete` 039
Write `operator delete` if you write `operator new`.

建構式，解構式，和Assignment 運算子 049
Constructors, Destructors, and Assignment Operators

條款 11：如果 classes 內動態配置有記憶體，請為此 class 宣告
一個 copy constructor 和一個 assignment 運算子 049
Declare a copy constructor and an assignment operator
for classes with dynamically allocated memory.

條款 12：在 constructor 中儘量以 initialization 取代 assignment 052
Prefer initialization to assignment in constructors.

條款 13：Initialization list 中的 members 初始化排列次序應該
和其在 class 內的宣告次序相同 057
List members in an initialization list in the order
in which they are declared.

條款 14：總是讓 base class 擁有 virtual destructor 059
Make sure base classes have virtual destructors.

條款 15：令 `operator=` 傳回「`*this` 的 reference」 064
Have `operator=` return a reference to `*this`.

條款 16：在 `operator=` 中為所有的 data members 指派內容 068
Assign to all data members in `operator=`.

條款 17：在 `operator=` 中檢查是否「自己派給自己」 071
Check for assignment to self in `operator=`.

類別與函式之設計和宣告 077
Classes and Functions: Design and Declaration

條款 18：努力讓介面完滿且最小化 079
Strive for class interfaces that are complete and minimal.

條款 19：區分 member functions, non-member functions 和
friend functions 三者 084
Differentiate among member functions, non-member functions,
and friend functions.

條款 20：避免將 data members 放在公開介面中 089
Avoid data members in the public interface.

條款 21：儘可能使用 const 091
Use const whenever possible.

條款 22：儘量使用 pass-by-reference（傳址），
少用 pass-by-value（傳值） 098
Prefer pass-by-reference to pass-by-value.

條款 23：當你必須傳回 object 時，不要嘗試傳回 reference 101
Don't try to return a reference when you must return an object.

條款 24：在函式多載化（function overloading）和
參數預設化（parameter defaulting）之間，謹慎抉擇 106
Choose carefully between function overloading
and parameter defaulting.

條款 25：避免對指標型別和數值型別進行多載化 109
Avoid overloading on a pointer and a numerical type.

條款 26：防衛潛伏的 ambiguity（模稜兩可）狀態 113
Guard against potential ambiguity.

條款 27：如果不想使用編譯器暗自產生的 member functions，
就應該明白拒絕它 116
Explicitly disallow use of implicitly generated member functions
you don't want.

條款 28：嘗試切割 global namespace（全域命名空間） 117
Partition the global namespace.

類別與函式之實作 123
Classes and Functions: Implementation

條款 29：避免傳回內部資料的 handles 123
Avoid returning "handles" to internal data.

條款 30：避免寫出 member function，傳回一個 non-const pointer 或
reference 並以之指向較低存取層級的 members 129
Avoid member functions that return non-const pointers or
references to members less accessible than themselves.

條款 31：千萬不要傳回「函式內 local 物件的 reference」，
或是「函式內以 new 獲得的指標所指的物件」 131
Never return a reference to a local object or to
a dereferenced pointer initialized by new within the function.

條款 32：儘可能延緩變數定義式的出現 135
Postpone variable definitions as long as possible.

條款 33：明智地運用 inlining 137
Use inlining judiciously.

條款 34：將檔案之間的編譯依存關係（compilation dependencies）降至最低 143
Minimize compilation dependencies between files.

繼承機制與物件導向設計 153
Inheritance and Object-Oriented Design

條款 35：確定你的 public inheritance 模塑出 "isa" 的關係 154
Make sure public inheritance models "isa."

條款 36：區分「介面繼承（interface inheritance）」和「實作繼承（implementation inheritance）」 161
Differentiate between inheritance of interface and inheritance of implementation.

條款 37：絕對不要重新定義一個繼承而來的非虛擬函式 169
Never redefine an inherited nonvirtual function.

條款 38：絕對不要重新定義一個繼承而來的預設參數值 171
Never redefine an inherited default parameter value.

條款 39：避免在繼承體系中做 cast down（向下轉型）的動作 173
Avoid casts down the inheritance hierarchy.

條款 40：透過 layering（分層技術）來模塑 has-a 或 is-implemented-in-terms-of 的關係 182
Model "has-a" or "is-implemented-in-terms-of" through layering.

條款 41：區分 inheritance 和 templates 185
Differentiate between inheritance and templates.

條款 42：明智地運用 private inheritance（私有繼承） 189
Use private inheritance judiciously.

條款 43：明智地運用多重繼承（multiple inheritance，MI） 194
Use multiple inheritance judiciously.

條款 44：說出你的意思並瞭解你所說的每一句話 210
Say what you mean; understand what you're saying.

雜項討論（Miscellany） 212

條款 45：知道 C++（編譯器）默默為我們完成和呼叫哪些函式 212
Know what functions C++ silently writes and calls.

條款 46：寧願編譯和聯結時出錯，也不要執行時才錯 216
Prefer compile-time and link-time errors to runtime errors.

條款 47：使用 non-local static objects 之前先確定它已有初值 219
Ensure that non-local static objects are initialized
before they're used.

條款 48：不要對編譯器的警告訊息視如不見 223
Pay attention to compiler warnings.

條款 49：儘量讓自己熟悉 C++ 標準程式庫 224
Familiarize yourself with the standard library.

條款 50：加強自己對 C++ 的瞭解 232
Improve your understanding of C++.

後記（Afterword） 237

索引（Index） 241