

## <<More Effective C++ ( >>

### 图书基本信息

书名：<<More Effective C++ ( 中文版 ) >>

13位ISBN编号：9787121125706

10位ISBN编号：7121125706

出版时间：2011-1

出版时间：电子工业出版社

作者：梅耶

页数：317

译者：侯捷

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

## &lt;&lt;More Effective C++ (&gt;&gt;

## 前言

C++ 是一门难学易用的语言！

C++ 的难学，不仅在其广博的语法、语法背后的语义、语义背后的深层思维、深层思维背后的对象模型；C++ 的难学，还在于它提供了4种不同（相辅相成）的编程思维模型：procedural-based, object-based, object-oriented, generic paradigm。

世上没有白吃的午餐。

又要有效率，又要弹性，又要前瞻望远，又要回溯相容，又要能治大国，又要能烹小鲜，学习起来当然就不可能太简单。

在如此庞大复杂的机制下，万千使用者前赴后继的动力是：一旦学成，妙用无穷。

C++ 相关书籍之多，车载斗量，如天上繁星，如过江之鲫。

广博如四库全书者有之（The C++ Programming Language、C++ Primer），深奥如重山复水者有之（The Annotated C++ Reference Manual, Inside the C++ Object Model），细说历史者有之（The Design and Evolution of C++, Ruminations on C++），独沽一味者有之（Polymorphism in C++, Genericity in C++），独树一帜者有之（Design Patterns, Large Scale C++ Software Design, C++ FAQs），程序库大全有之（The C++ Standard Library），另辟蹊径者有之（Generic Programming and the STL），工程经验之累积亦有之（Effective C++, More Effective C++, Exceptional C++）。

这其中，&ldquo;工程经验之累积&rdquo;对已具 C++ 相当基础的程序员而言，有着致命的吸引力与立竿见影的帮助。

Scott Meyers 的 Effective C++ 和 More Effective C++ 是此类佼佼，Herb Sutter 的 Exceptional C++ 则是后起之秀。

这类书籍的一个共同特色是轻薄短小，并且高密度地纳入作者浸淫于 C++/OOP 领域多年而广泛的经验。

它们不但开扩读者的视野，也为读者提供各种 C++/OOP 常见问题或易犯错误的解决模型。

某些小范围主题诸如&ldquo;在 base classes 中使用 virtual destructor&rdquo;、&ldquo;令 operator= 传回 \*this 的 reference&rdquo;，可能在百科型 C++ 语言书籍中亦曾概略提过，但此类书籍以深度探索的方式，让我们了解问题背后的成因、最佳的解法，以及其他可能的牵扯。

至于大范围主题，例如 smart pointers, reference counting, proxy classes, double dispatching, 基本上已属 design patterns 的层级！

这些都是经验的累积和心血的结晶！

我很高兴将以下两本优秀书籍，规划为一个系列，以郑重的形式呈现给您： 1.Effective C++ 2/e, by Scott Meyers, AW 1998 2.More Effective C++, by Scott Meyers, AW 1996 本书不但与英文版页对译，保留索引，并加上译注、交叉索引、读者服务。

这套书将对于您的程序设计生涯带来重大帮助。

翻译这套书籍的过程中，我感觉来自技术体会上的极大快乐。

我祈盼（并相信）您在阅读此书时拥有同样的心情。

## <<More Effective C++ (>>

### 内容概要

较少，页数倒是多了一些，原因是这次选材比“第一集”更高阶，尤其是第5章。

Meyers将此章命名为技术（techniques），并明白告诉你，其中都是一些patterns，例如virtual constructors，smart pointers，reference counting，proxy classes，double dispatching.....这一章的每个条款篇幅都达15~30页之多，实在让人有“山重水复疑无路，柳暗花明又一村”之叹。

虽然出版年代稍嫌久远，但本书并没有第2版，原因是当其出版之时（1996），C++ Standard已经几乎定案，本书即依当时的标准草案而写，其与现今的C++标准规范几乎相同。

而且可能变化的几个弹性之处，Meyers也都有所说明与提示。

读者可以登录作者提供的网址，看看上下两集的勘误与讨论（数量之多，令人惊恐。

幸好多是技术讨论或文字斟酌，并没有什么重大误失）。

## <<More Effective C++ (>>

### 作者简介

作者：（美国）梅耶（Scott Meyers）译者：侯捷Scott Meyers，世界顶级的C++软件开发技术权威之一。

他是两本畅销书Effective C++和More Effective C++的作者，以前曾经是C++ Report的专栏作家。

他经常为C/C++ Users Journal和Dr. Dobb's Journal撰稿，也为全球范围内的客户做咨询活动。

他也是Advisory Boards for NumeriX LLC和InfoCruiser公司的成员。

他拥有Brown University的计算机科学博士学位。

侯捷，集计算机技术读物之著、译、评于一身，是《多态与虚拟》和《STL源码剖析》的作者，《Run !

Pc》杂志、《程序员》杂志的知名专栏作家，也是极其号召力的大学教师与研讨会讲师。

他于1989年获得中国台湾清华大学动机系硕士学位。

## 书籍目录

译序 (侯捷) 导读 (Introduction) 基础议题 (Basics) 条款1: 仔细区别 pointers 和 references  
Distinguish between pointers and references. 条款2: 最好使用 C++ 转型操作符 Prefer C++-style casts.  
条款3: 绝对不要以多态 (polymorphically) 方式处理数组 Never treat arrays polymorphically. 条款4  
: 非必要不提供 default constructor Avoid gratuitous default constructors. 操作符 (Operators)  
对定制的“类型转换函数”保持警觉 Be wary of user-defined conversion functions. 条款6: 区别  
increment/decrement 操作符的前置 (prefix) 和后置 (postfix) 形式 Distinguish between prefix and  
postfix forms of increment and decrement operators. 条款7: 千万不要重载 &&, || 和 , 操作符 Never  
overload &&, ||, or ,. 条款8: 了解各种不同意义的 new 和 delete Understand the different meanings of  
new and delete 异常 (Exceptions) 条款9: 利用 destructors 避免泄漏资源 Use destructors to prevent  
resource leaks. 条款10: 在 constructors 内阻止资源泄漏 (resource leak) Prevent resource leaks in  
constructors. 条款11: 禁止异常 (exceptions) 流出 destructors 之外 Prevent exceptions from leaving  
destructors. 条款12: 了解“抛出一个 exception”与“传递一个参数”或“调用一个虚函数”之间的  
差异 Understand how throwing an exception differs from passing a parameter or calling a virtual  
function. 条款13: 以 by reference 方式捕捉 exceptions Catch exceptions by reference. 条款14: 明智运  
用 exception specifications Use exception specifications judiciously. 条款15: 了解异常处理 (exception  
handling) 的成本 Understand the costs of exception handling. 效率 (Efficiency) 条款16: 谨记 80-20 法  
则 Remember the 80-20 rule. 条款17: 考虑使用 lazy evaluation (缓式评估) Consider using lazy  
evaluation. 条款18: 分期摊还预期的计算成本 Amortize the cost of expected computations. 条款19:  
了解临时对象的来源 Understand the origin of temporary objects. 条款20: 协助完成“返回值优化  
(RVO)” Facilitate the return value optimization. 条款21: 利用重载技术 (overload) 避免隐式类型  
转换 (implicit type conversions) Overload to avoid implicit type conversions. 条款22: 考虑以操作符复  
合形式 (op=) 取代其独身形式 (op) Consider using op= instead of stand-alone op. 条款23: 考虑使  
用其他程序库 Consider alternative libraries. 条款24: 了解 virtual functions、multiple inheritance、virtual  
base classes、runtime type identification 的成本 Understand the costs of virtual functions, multiple  
inheritance, virtual base classes, and RTTI. 技术 (Techniques, Idioms, Patterns) 杂项讨论 (Miscellany) 索  
引 (一) (General Index) 索引 (二) (Index of Example Classes, Functions, and Templates)

## 章节摘录

版权页：插图：我们终于抵达了最后一站。

本章内含难以归类的准则。

一开始的两个条款讨论C++软件开发过程如何设计出能够容纳日后变化的系统。

是的，面向对象方法应用于系统构造的一个强大力量就是，它支持日后的变化。

这些条款描述了一些特定步骤，你可以用来强化你的软件工事，抵抗这个拒绝停滞的世界带来的刀戟箭弩。

接下来我将验证如何在同一个程序中结合C和C++。

这个需求导致语言上的额外考虑，不过C++毕竟生存于真实世界之中，有时候我们必须面对这样的问题。

最后，我把“C++标准规格”公开之后的各项语言变化做一番摘要整理。

在此特别涵盖标准程序库中翻天覆地的大变化（亦请参考条款E49）。

如果你未曾密切跟随标准化的脚步，对于这些变化可能会有很大的惊喜。

是的，标准程序库中有许多让人愉悦的东西。

条款32：在未来时态下发展程序 世事永远在变。

身为软件开发人员，我们可能不是知道得很多，但我们确切知道世事永远在变。

我们不一定知道改变的是什么，改变如何到来，改变何时发生，或为什么会发生，但我们真的知道：事情会改变。

好的软件对于变化有良好的适应能力。

好的软件可以容纳新的性质，可以移植到新的平台，可以适应新的需求，可以掌握新的输入。

软件具备如此的弹性、健壮性、可信赖度，并非是天上掉下来的礼物，而是那些“即使面对今天的束缚，仍然对明天可能的需求念兹在兹”的设计者和实现者共同努力的结果。

## 媒体关注与评论

《Effective c++》(Scott Meyers第一本书)的荣耀：“对于任何渴望在中阶或高阶层面精通c++的人，我慎重推荐《Effective c++》，” ——(The C / C++User's Journal)

编辑推荐

《More Effective C++:35个改善编程与设计的有效方法(中文版)》：传世经典书丛



#### 版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>