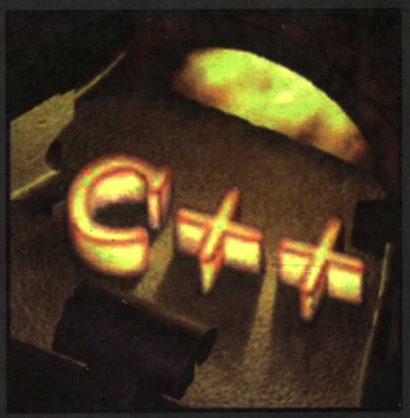
HUSTP

C++ 标准程序库

- 自修教程与参考手册 -

The C++ Standard Library

A Tutorial and Reference



Nicolai M.Josuttis

华中科技大学出版社
http://press.hust.edu.cn

侯捷/孟岩 译

C++ 标准程序库

The C++ Standard Library

自修教程与参考手册（A Tutorial and Reference）

Nicolai M. Josuttis 著

侯捷 / 孟岩 译

C++ 标准程序库

The C++ Standard Library

Nicolai M. Josuttis

图书在版编目(CIP)数据

C++标准程序库/(德)Nicolai M. Josuttis 著；侯 捷/孟 岩 译

武汉：华中科技大学出版社，2002 年 9 月

ISBN 7-5609-2782-3

Ⅰ. C…

Ⅱ. ①N… ②侯… ③孟…

Ⅲ. C++-程序设计

Ⅳ. TP312

责任编辑：周 筠(http://yeka.xilubbs.com junzhou@public.wh.hb.cn)

技术编辑：孟 岩

出版发行：华中科技大学出版社 （武昌喻家山 邮编：430074）

录 排：华中科技大学惠友科技文印中心

印 刷：湖北新华印务有限公司

开 本：787×1092 1/16 印 张：51.75 字 数：800 000

版 次：2002 年 9 月第 1 版 印 次：2002 年 9 月第 1 次印刷

印 数：1—12 000 定 价：108.00 元

ISBN 7-5609-2782-3/TP・478

巨细靡遗 井然有序

（侯捷译序）

自从 1998 年 C++ *Standard* 定案以后，C++ 程序库便有了大幅扩充。原先为大家所熟知、标准规格定案前酝酿已久的 STL（Standard Template Library，标准模板程序库），不再被单独对待，而是被纳入整个 C++ 标准程序库（Standard Library）。同时，原有的程序库（如 iostream）也根据泛型技术（generics）在内部做了很大的修改。可以说，C++ *Standard* 的发布对 C++ 社群带来了翻天覆地的大变动——不是来自语言本身，而是来自标准程序库。这个变动，影响 C++ 程序编写风格至巨，C++之父 Bjarne Stroustrup 并因此写了一篇文章：*Learning Standard C++ as a New Language*（载于 C/C++ User's Journal, 1999/05）。

我个人于1998年开始潜心研究泛型技术和STL，本书英文版《***The C++ Standard Library***》甫一出版便成为我学习 C++ 标准程序库的最重要案头工具书之一。小有心得之后，我写过数篇相关技术文章，从来离不开本书的影响和帮助。我曾经把 STL（代表泛型技术目前最被广泛运用的一个成熟产品，也是 C++ 标准程序库的绝大成分）的学习比喻为三个境界（或层次）：

- 第一境界：熟用 STL
- 第二境界：了解泛型技术的内涵与 STL 的学理乃至实作
- 第三境界：扩充 STL

不论哪一个阶段，你都能够从本书获得不同程度的帮助。

第一阶段（对最大多数程序员有立竿见影之效），我们需要一本全面而详尽的教程，附带大量设计良好的范例，带领我们认识十数个 STL 容器（containers）、数十个 STL 算法（algorithms）、许许多多的迭代器（iteartors）、配接器（adapters）、仿函数（functors）……的各种特性和用途。这些为数繁多的组件必须经过良好的编排组织和索引，才能成就一本效果良好、富教育性又可供长久查阅的案头工具书。

在这一阶段里，本书表现极为优异。书中运用许多图表，对所有 STL 组件的成员做了极其详尽的整理。更值得称道的是书中交叉参考（cross reference）做得非常好，在许多关键地点告诉读者当下可参见哪一章哪一节哪一页，对于阅读和学习带来很大的帮助（本中文版以页页对译方式保留了所有交叉参考和索引）。

第二阶段（从 STL 的运用晋升至泛型技术的学习），我们需要一些关键的 STL 源代码（或伪码, pseudo code），帮助我们理解关键的数据结构、关键的编程技术。认识这些关键源代码（或伪码）同时也有助提升第一阶段的运用深度（学会使用一样东西，却不知道它的道理，不高明☺[1]）。

本书很多地方都提供了 C++ 标准程序库的关键源代码。不全面，但很关键。

第三阶段（成为一位泛型技术专家；打造自己的 STL 兼容组件），我们需要深入了解 STL 的设计理念和组织架构[2]，并深入（且全面地）了解 STL 实作手法[3]。是的，不入虎穴，不能得虎子；彻底了解 STL 如何被打造出来之后，你才能写出和 STL 水乳交融、完美整合的自定义组件（user-defined components）。

本书对第二阶段的学习也有相当帮助。虽然没能提供全面的 STL 源码并分析其技术（那需要另外 800 页☺），却提供了为数不少的订制型组件实作范例：p191, p213 提供了一个执行期指定排序准则并运用不同排序准则的实例，p219 提供了一个自定义容器（虽然只是个简单的包装类别），p222 提供了一个“reference 语意”示范作法，p285 提供了一个针对迭代器而设计的泛型算法，p288 提供了一个用于关联式容器的定制型 inserter，p294 有一个自定的排序准则，p441 有一个自定的（安全的）stack，p450 有一个自定的（安全的）queue，p504 有一个自定的 traits class for string，p614 有一个自定的 stream 操控器，p663 有一个自定的 stream 缓冲区，p735 有一个自定的内存配置器（allocator）。

---

1 这是乍见之下令人错愕的一句话。看电视需要先了解电视的原理吗？呵呵，话讲白了就没意思了。这句话当然是对技术人员说的。

2 这方面我推荐你看《*Generic Programming and the STL - Using and Extending the C++ Standard Template Library*》, by Matthew H. Austern, Addison Wesley 1998。详见稍后说明。中译本《泛型程序设计与 STL》,侯捷/黄俊尧合译，碁峰, 2001。

3 这方面我推荐你看《STL 源码剖析, *The Annotated STL Sources*》by 侯捷，碁峰, 2002。详见稍后说明。

除了众所瞩目的 STL，本书也涵盖了一般不被归类为 STL 的 String 程序库，以及一般不被视为关键的 IOStream 和 Locale 程序库[4]。三部分互有关连，以 IOStream 为主干。在 GUI（图形使用接口）和 application framework（应用程序框架）当道的今天，IOStream 提供的输出输入可能对大部分人失去了价值，但如果你希望开拓 OO 技术视野，IOStream 是一颗沉睡的珠宝。

❖ ❖ ❖

泛型技术不仅在 C++ 被发扬光大，在 Java 上也有发展[5]，在 C# 上亦被众人期待。从目前的势头看，泛型技术（Generics）或许是面向对象（Object Oriented）技术以来程序编写方面的又一个巨大冲击。新一代 C++ 标准程序库[6]将采用更多更复杂更具威力的泛型技术，提供给 C++ 社群更多更好更具复用价值的组件。

不论你要不要、想不想、有没有兴趣在你的程序编写过程中直接用上泛型技术，至少，在 C++ 程序编写过程中你已经不可或缺于泛型技术带来的成熟产品：C++ 标准程序库。只要你具备 C++ 语言基础，本书便可以带领你漂亮地运用 C++ 标准程序库，漂亮地提升你的编程效率和程序品质。

面对陌生，程序员最大的障碍在于心中的怯懦。To be or not to be, that is the question! 不要像哈姆雷特一样犹豫不决。面对光明的技术，必须果敢。

❖ ❖ ❖

关于术语的处理，本书大致原则如下：

1. STL 各种数据结构名称皆不译，例如 array, vector, list, deque, hast table, map, set, stack, queue, tree...。虽然其中某些已有约定俗成的中文术语，但另一些没有既标准又被普遍运用的中文名称，强译之会令读者瞠目以对，部分译部分不译则阅读时词性平衡感不佳（例如“面对向量和 **deque** 两种容器...”就不如“面对 **vector** 和 **deque** 两种容器...”读起来顺畅）。因此，数据结构名称全部不译。直接呈现这些简短的英文术语，可能营造更突出的视觉效果，反而有利阅读。技术书籍的翻译不是为了建立全中文化阅读环境，我们的读者水

4 这方面我见过的唯一专著是《*Standard C++ IOStreams and Locales - Advanced Programmer's and Reference*》, by Angelika Langer and Klaus Kreft, Addison Wesley 2000.

5 (1) *GJ : A Generic Java*, by Philip Wadler, Dr. Dobb's Journal February 2000.
(2) *JSR- 000014 : Adding Generics to the Java Programming Language*, http://jcp.org/aboutJava/communityprocess/review/jsr014/index.html.

6 请参考 http://www.boost.org/，这个程序库据称将成为下一代 C++ 标准。

平也不可能受制于这些英文单词。

2. STL 六大组件的英文名称原打算全部保留，但由于处处出现，对版面的中英文比例形成视觉威胁，因此全部采用以下译名：**container** 容器，**algorithm** 算法，**iterator** 迭代器，**adapter** 配接器，**functor** 仿函数[7]，**allocator** 配置器。
3. 任何一个被保留的英文关键术语，其第一次（或前数次）出现时尽可能带上中文名称。同样地，任何关键的中文术语，我也会时而让它中英并陈。

❖ ❖ ❖

关于编排，本书原则如下：

1. 全书按英文版页次编排，并因而得以保留原书索引。索引词条皆不译。
2. 中文版采用之程序代码字体（Courier New 8.5）比文本字体（细明体 9.5）小，英文版之程序代码字体却比其文本字体大，且行距宽。因此中文版遇有大篇幅程序行表时，为保持和英文版页次相应，便会出现较多留白。根据我个人对书籍的经验，去除这些留白的最后结果亦不能为全书节省五页十页；填满每一处空白却丧失许多立即可享的好处，为智者不取☺。

❖ ❖ ❖

一旦你从本书获得了对 C++ 标准程序库运用层面的全盘掌握与实践经验之后，可能希望对 STL 原理乃至实作技术做更深的研究，或甚至对泛型编程（Generic Programming）产生无比狂热。在众多相关书籍之中，下面是我认为非常值得继续进修的四本书：

1. 《*Generic Programming and the STL - Using and Extending the C++ Standard Template Library*》, by Matthew H. Austern, Addison Wesley 1998。本书第一篇（前五章）谈论 STL 的设计哲学、程序库背后的严密架构和严谨定义。其中对于 STL 之异于一般程序库，有许多重要立论。其余部分（第二篇、第三篇）是 STL 的完整规格（分别从 concepts 的角度和 components 的角度来阐述），并附范例程序。

[7] 原书大部分时候使用 **function object**（函数对象）一词，为求精简及突出，中文版全部改用其另一个名称 **functor**（仿函数）（见第 8 章译注）。

2. 《STL 源码剖析, *The Annotated STL Sources*》by 侯捷, 碁峰, 2002。本书剖析 STL 实作技法，详实揭示并注释 STL 六大组件的底层实作，并以公认最严谨的 SGI（Silicon Graphics Inc.）STL 版本为剖析对象。附许多精彩分析图，对于高度精巧的内存配置策略、各种数据结构、各种算法、乃至极为“不可思议”的配接器（adapter）实作手法，都有深入的剖析。
3. 《*Effective STL*》, by Scott Meyers, Addison Wesley 2001。本书定位为 STL 的深层运用。在深层运用的过程中，你会遇到一些难解的问题和效率的考虑，你需要知道什么该做、什么该避免。本书提供 50 个专家条款。请注意，深层运用和效率调校，可能需要读者先对底部机制有相当程度的了解。
4. 《*Modren C++ Design*》by Andrei Alexandrescu, Addison Wesley 2001。将泛型技术发挥到淋漓尽致、令人目瞪口呆的一本书籍。企图将泛型技术和设计模式（design patterns）结合在一起。是领先时代开创先河的一本书。

❖ ❖ ❖

本书由我和孟岩先生共同完成。孟岩在大陆技术论坛以 C++/OO/Generics 驰名，见解深隽，文笔不凡。我很高兴和他共同完成这部作品。所谓合译，我们两人对全书都有完整的参与（而非你一半我一半的对拆法），最终由我定稿。本书同时发行繁体版和简体版，基于两岸计算机术语的差异性，简体版由孟岩负责必要转换。

侯捷 2002/05/23 于新竹

http://www.jjhou.com（繁体网站）
http://jjhou.csdn.net（简体网站）
jjhou@jjhou.com（个人电子邮箱）

孟岩译序

IT 技术书籍市场，历来是春秋战国。一般来说，同一个技术领域里总会有那么数本、十数本、甚至数十本定位相似的书籍相互激烈竞争。其中会有一些大师之作脱颖而出，面南背北，黄袍加身。通常还会有后来者不断挑战，企图以独到特色赢得自己的一片天地。比如说在算法与数据结构领域，D. E. Knuth 的那套《*The Art of Computer Programming*》一至三卷，当然是日出东方，惟我独尊。但是他老人家的学生 Robert Sedgewick 凭着一套更贴近实用的《*Algorithms in C*》系列，也打出了自己的一片天下，成为很多推荐列表上的首选。就 C++ 应用经验类书籍来说，Scott Meyers 的《*Effective C++*》称王称霸已经多年，不过其好友 Herb Sutter 也能用一本《*Exceptional C++*》获得几乎并驾齐驱的地位。嗨，这不是很正常的事吗？技术类书籍毕竟不是诗词歌赋。苏轼一首“明月几时有，把酒问青天”，可以达到“咏中秋者，自东坡西江月后，余词尽废”的程度，但怎么可能想象一本技术著作达到“我欲乘风归去，又恐琼楼玉宇，高处不胜寒”的境界！谁能够写出一本技术书，让同一领域后来者望而却步，叹为观止，那才是大大的奇迹！

然而，您手上这本《*The C++ Standard Library*》，作为 C++ 标准程序库教学和参考类书籍的定音之作，已经将这个奇迹维持了三年之久。按照 IT 出版界时钟，三年的时间几乎就是半个世纪，足以锤炼又一传世经典！

1998 年 *C++ Standard* 通过之后，整个 C++ 社群面临的最紧迫任务，就是学习和理解这份标准给我们带来的新观念和新技术。而其中对于 C++ 标准程序库的学习需求，最为迫切。C++ 第二号人物 Andrew Koenig 曾经就 C++ 的特点指出：“语言设计就是程序库设计，程序库设计就是语言设计”[1]。*C++ Standard* 对程序库所作的巨大扩充和明确规范，实际上即相当于对 C++ 语言的能力作了全面提升与扩展，意味着你可以站在无数超一流专家的肩上，将最出色的思想、设计与技术纳入囊中，

[1] “Language design is library design, library design is language design”，参见 Andrew Koenig, Barbara Moo 合著《Ruminations on C++》第 25, 26 章标题。

让经过千锤百炼的精美代码成为自己软件大厦的坚实基础。可以说，对于大多数程序员来说，标准 C++ 较之于“ARM 时代”之最大进步，不是语言本身，而恰恰是标准程序库。因此，我们可以想象当时人们对 C++ 标准程序库教学类书籍的企盼，是何等热切！

一方面是已经标准化了的成熟技术，另一方面是万众期待的眼神，我们完全有理由认为，历史上理应爆发一场鱼龙混杂的图书大战。它的典型过程应该是这样：先是一批快刀手以迅雷不及掩耳盗铃之势[2]推出一堆敛财废纸，然后在漫长的唾骂与期待中，大师之作渐渐脱颖而出。大浪淘沙，最后产生数本经典被人传颂。后虽偶有新作面世，但波光点点已是波澜不兴。

然而，这一幕终究没有在“C++ 标准程序库教学与参考书籍”领域内出现。时至今日，中外技术书籍市场上这一领域内的书籍为数寥寥，与堆积如山的 C++ 语言教学类书籍形成鲜明对比。究其原因，大概有二，一是这个领域里的东西毕竟份量太重，快刀手虽然善斩乱麻，对于 C++ 标准程序库这样严整而精致的目标，一时也难以下手。更重要的原因则恐怕是 1999 年 8 月《*The C++ Standard Library*》问世，直如横刀立马，震慑天下。自推出之日起至今，本书在所有关于 C++ 标准程序库的评论与推荐列表上，始终高居榜首，在 Amazon 的销量排行榜上名列所有 C++ 相关书籍之最前列。作者仅凭一书而为天下知，成为号召力可与 Stan Lippman, Hurb Sutter 等“经典”C++ 作家比肩的人物。此书之后，虽然仍有不少著作，或深入探讨标准程序库的某些组件，或极力扩展标准库倡导的思想与技术，但是与《*The C++ Standard Library*》持同一路线的书籍，再没有出现过。所谓泰山北斗已现，后来者已然无心恋战。

于是有了这样的评论：“如果你只需要一本讲述 C++ 标准程序库和 STL 的书籍，我推荐 Nicolai Josuttis 的《*The C++ Standard Library*》。它是你能得到的唯一一本全面讲述 C++ 标准程序库的书，也是你能想象的最好的一本书。”这种奇异情形在当今技术书坛，虽然不是绝无仅有，也是极为罕见。

究竟这本书好到什么程度，可以获得这么高的评价？

我正是带着这份疑问，接受侯捷先生的邀请，着手翻译这本经典之作。随着翻译过程的推进，我也逐渐解开了心中的疑惑。在我看来，这本书的特点有四：内容详实，组织严密，态度诚恳，深入浅出。

---

2 此处非笔误，而是大陆流行的一句“新俚语”，意思十分明显，就是“迅雷不及掩耳”地“掩耳盗铃”。

首先，作为一本程序库参考手册，内容详实全面是一项基本要求。但是，本书在这方面所达到的高度可以说树立了一个典范。本书作者一开始就提出一个极高的目标，要帮助读者解决“使用 C++ 标准程序库过程中所遇到的所有问题”。众所周知，C++ 标准程序库是庞然大物，每一部分又有很精深的思想和技术，既不能有所遗漏，又不能漫无边际地深入下去，何取何舍，何去何从，难度之大可想而知！作者在大局上涵盖了 C++ 标准程序库的全部内容，在此基础上又对所有组件都进行细致的、立体式的讲解。所谓立体式讲解，就是对于一个具体组件，作者首先从概念上讲解其道理，然后通过漂亮的范例说明其用法，申明其要点，最后再以图表或详解方式给出参考描述。有如钱塘江潮，层层叠叠，反反复复，不厌其烦。读完此书，我想您会和我一样感受冲击，并且完全体认作者付出的巨大心血。

C++ 标准程序库本身就是一个巨大的有机整体，加上这本书的立体讲解方式，前后组织和对应的工作如果做不好，很容易会使整部书显得散乱。令人钦佩的是，这本书在组织方面极其严密，几无漏洞。相关内容的照应、交叉索引、前后对应，无一不处理得妥善曼妙。整体上看，整本书就像一张大网，各部分内容之间组织严谨，契合密切，却又头绪清晰，脉络分明，着实难能可贵。我在阅读和翻译过程中，常常诧异于其内容组织的精密程度，简直像德国精密机械一样分毫不差——后来才想到，本书作者 Nicolai Josuttis 就是德国人，精密是德意志民族的性格烙印，真是名不虚传！

说起德意志民族，他们的另一个典型性格就是诚实坦率。这一点在这本书同样有精彩的展现。身为 C++ 标准程序库委员会成员，作者对于 C++ 标准程序库的理解至深，不但清楚知道其优点何在，更对其缺陷、不足、不完备和不一致的地方了如指掌。可贵的是，在这些地方，作者全不避讳，开诚布公，直言不讳，事实是什么样就是什么样，绝不文过饰非，绝不含混过关。作为读者，我们不仅得以学到好东西，而且学到如何绕开陷阱和障碍。一个最典型的例子就是对于 valarray 的介绍，作者先是清清楚楚地告诉读者，由于负责该组件设计的人中途退场，这个组件没有经过细致的设计，最好不要使用。然后作者一如既往，详细介绍 valarray 的使用，完全没有因为前面的话而稍微有所懈怠。并且在必要的地方将 valarray 的设计缺陷原原本本地指出来，让读者口服心服。读到这些地方，将心比心，我不禁感叹作者的坦诚与无私，专精与严谨。

本书最具特色之处，就是其内容选取上独具匠心，可谓深入浅出。本书的目的除了作为手册使用，更是一本供学习者阅读学习的 "tutorial"（自学教本）。也就是说，除了当手册查阅，你也可以捧着它一篇一篇地阅读学习，获得系统化的坚实知识。一本书兼作 "tutorial" 和 "reference"，就好像一本字典兼作“作文指南”，没

有极高的组织能力和精当的内容选择，简直难以想象会搞成什么样子。了不起的是本书不仅做到了，而且让你感觉，学习时它是一本最好的 "tutorial"，查阅时它是一本最好的 "reference"，我要说，这是个奇迹！单从学习角度来说，本书极为实用，通过大量鲜明的例子和切中要害的讲解让你迅速入门，而且绝不仅仅浅尝辄止，而是不失时机地深入进去，把组件的实作技术和扩展方法都展现给读者。单以 STL 而论，我经常以侯捷先生提出的“STL 学习三境界”来描述一本书的定位层次，这本书就像一座金字塔，扎根于实用，尖锋直达“明理”和“扩展”层次。从中你可以学到“reference 语意”的 STL 容器、smart pointer（智能指针）的数种实现、扩充的组合型仿函数（composing function object）、STL 和 IOStream 的扩展方法、定制型的配置器（allocator）设计思路等等高级技术，也能学到大量的实践经验，比如 vector 的使用技巧，STL 容器的选择，`basic_string<>` 作为容器的注意事项等等。可以这么说，这本书足以将你从入门带到高手层次，可谓深入浅出，精彩至极！

我很高兴自己第一次进行技术书籍翻译，就能够碰到这样一本好书，这里要深深感谢侯捷先生给我一辈子都庆幸的机会。翻译过程出乎意料地艰辛，前后持续将近 10 个月。我逐字逐句地阅读原文，消化理解，译成自以为合适的中文，然后再由侯先生逐字逐句地阅读原文，对照我的粗糙译文，进行修订和润色，反复品味形成最终译稿。作为译者，侯先生和我所追求的是，原书技术的忠实呈现加上中文化、中国式的表达。我们为此花费了巨大的心力，对我来说，付出的心血远远超过一般翻译一本书的范畴。虽然最终结果需要广大读者评论，但今天面对这厚厚的书稿，我问心无愧地享受这份满足感。我最大的希望是，每一位读者在学习和查阅这本中文版的时候，完全忘掉译者曾经的存在，感觉不到语言的隔阂，自由地获取知识和技术。对于一个初涉技术翻译的人来说，这个目标未免太贪心了，可是这始终会是我心里的愿望。一个译者应该是为了被忽略而努力的。

最后，感谢侯先生一直以来对我的欣赏和帮助，感谢您给我的机会，我十分荣幸！感谢华中科技大学出版社的周筠老师，您始终友好地关注着我，鼓励着我。感谢 CSDN 的蒋涛先生，您的热情鼓励始终是我的动力。感谢我的父母、弟弟，你们是我最爱的人，是我最坚强的支柱！感谢曾经帮助过我，曾经关心过我的每一个人，无论你现在怎样，我为曾经拥有过的，仍然拥有着的每一片快乐和成果，衷心地感谢你！

祝各位读书快乐！

孟岩　2002 年 5 月于北京

前言
Preface

一开始，我只不过想写一本篇幅不大的有关于 C++ 标准程序库的德文书（也就 400 多页吧）。萌生这个想法是在 1993 年。而在 1999 年的今天，您看到了这个想法的成果：一本英文书，厚达 800 多页，其中包含大量的文字描述、图片和范例。我的目标是，详尽讲解 C++ 标准程序库，使你的所有（或几乎所有）编程问题都能够在你遇到之前就先给你解答。然而，请注意，这不是一种完整描述 C++ 标准程序库的所有方面的书籍，我通过“在 C++ 中利用标准程序库进行学习和程序编写”的形式，表现出最重要的主题。

每一个主题都是以一般性概念为基础而开展，然后导入日常程序编写工作所必须了解的具体细节。为了帮助你理解这些概念和细节，书中提供详尽的范例程序。

这就是我的前言——言简意赅！撰写此书的过程中，我得到了很多乐趣，希望你阅读本书时，能够像我一样快乐。请享用！

致　谢

Acknowledgments

这本书表达的观点、概念、解决方案和范例，来源十分广泛。从这个意义上讲，封面只列我一个人的名字，未免不公平。所以我愿在此向过去数年来帮助和支持我的人和公司，表示诚挚的谢意。

我第一个要感谢的是 Dietmar Kühl。Dietmar 是一位 C++专家，尤其精通 I/O streams（输入输出流）和国际化（他曾经仅仅为了好玩而写了一个 I/O stream library）。他不仅将本书的大部分从德文译为英文，还亲自动笔，发挥专长，为本书撰写了数节内容。除此之外，在过去的数年里，他还向我提供了很多宝贵的反馈意见。

其次，我要感谢所有审阅者和那些向我表达过意见的人。他们的努力使本书的品质获得巨大提升。由于名单太长，以下如有任何疏漏，还请见谅。英文版的检阅者包括：Chuck Allison, Greg Comeau, James A. Crotinger, Gabriel Dos Reis, Alan Ezust, Nathan Myers, Werner Mossner, Todd Veldhuizen, Chichiang Wan, Judy Ward, Thomas Wike-hult。德文版的检阅者包括：Ralf Boecker, Dirk Herrmann, Dietmar Kühl, Edda Lörke, Herbert Scheubner, Dominik Strasser, Martin Weitzel。其它人包括：Matt Austern, Valentin Bonnard, Greg Colvin, Beman Dawes, Bill Gibbons, Lois Goldthwaite, Andrew Koenig, Steve Rumsby, Bjarne Stroustrup 和 David Vandevoorde。

我要特别感谢 Dave Abrahams, Janet Cocker, Catherine Ohala 和 Maureen Willard，他们对全书进行了非常细致的审阅和编辑。他们的反馈意见让本书的品质获得了难以置信的提升。

我也要特别感谢我的“活字典”Herb Sutter，他是著名的 “Guru of the Week” 的创始人，这是一个常设的 C++ 难题讲解专栏，在 comp.std.c++.moderated 上发表。

我还要感谢一些公司和个人，他们的帮助使我有机会在各个不同的平台上，使用各种不同的编译器来测试自己的范例程序。非常感谢来自 EDG 的 Steve Adamczyk，Mike Anderson 和 John Spicer，他们的编译器真是太棒了，在 C++ 标

准化过程和本书写作过程中，提供了巨大的支持。感谢 P. J. Plauger 和 Dinkumware, Ltd，他们很早以来就持续进行与 C++ 标准规格兼容的 C++ 标准程序库开发工作。感谢 Andreas Hommel 和 Metrowerks，他们完成了深具价值的 CodeWarrior 程序开发环境。感谢 GNU 和 egcs 编译器的所有开发者。感谢 Microsoft，他们完成了深具价值的 Visual C++。感谢 Siemens Nixdorf Informations Systems AG 的 Roland Hartinger，他提供了一份他们的 C++ 编译器测试版本。感谢 Topjects GmbH，为了他那一份深具价值的 ObjectSpace library 实作品。

感谢 Addison Wesley Longman 公司里头与我共同工作过的每一个人，包括 Janet Cocker, Mike Hendrickson, Debbie Lafferty, Marina Lang, Chanda Leary, Catherine Ohala, Marty Rabinowitz, Susanne Spitzer, 和 Maureen Willard 等等。这项工作真是太有趣了。

此外，我还要感谢 BREDEX GmbH 的人们，感谢 C++ 社群中的所有人，特别是那些参与标准化过程的人，感谢他们的支持和耐心（有时候我问的问题确实挺傻）。

最后，也是最重要的，我要将我的感谢（附上一个亲吻）送给我的家人：Ulli, Lucas, Anica 和 Frederic。为了这本书，我很长时间没有好好陪他们了。

但愿各位能从这本书获得乐趣，另外，请保持宽厚。
（译注：上句原文为 Have fun and be human!。抱歉，译者对其精确含义并无十足把握）

目录