

PEARSON

Broadview®  
www.broadview.com.cn

# C++ 标准库 (第2版)

The C++ Standard Library  
A Tutorial and Reference, Second Edition

[德] Nicolai M. Josuttis 著  
侯捷 译

 中国工信出版集团

 电子工业出版社  
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY  
http://www.phei.com.cn

# C++ 标准库 第2版

The C++ Standard Library  
A Tutorial and Reference, Second Edition

[德] Nicolai M. Josuttis 著  
侯捷 译

电子工业出版社  
Publishing House of Electronics Industry  
北京·BEIJING

## 内容简介

本书是全球 C++ 经典权威参考书籍时隔 12 年, 基于 C++11 标准的全新重大升级。标准库提供了一组公共类和接口, 极大地拓展了 C++ 语言核心功能。《C++ 标准库 (第2版)》详细讲解了每一标准库组件, 包括其设计目的和方法、复杂概念的剖析、实用而高效的编程细节、存在的陷阱、重要的类和函数, 又辅以大量用 C++11 标准实现的实用代码范例。除覆盖全新组件、特性外, 本书一如前版, 重点着眼于标准模板库 (STL), 涉及容器、迭代器、函数对象以及 STL 算法。此外, 本书同样关注 lambda 表达式、基于区间的 for 循环、move 语义及可变参数模板等标准库中的新式 C++ 编程风格及其影响。

Authorized translation from the English language edition, entitled THE C++ STANDARD LIBRARY: A TUTORIAL AND REFERENCE, 2E, 9780321623218 by JOSUTTIS, NICOLAI M., published by Pearson Education, Inc. publishing as Addison-Wesley Professional, Copyright © 2012 Pearson Education, Inc.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

CHINESE SIMPLIFIED language edition published by PEARSON EDUCATION ASIA LTD., and PUBLISHING HOUSE OF ELECTRONICS INDUSTRY Copyright © 2015.

本书简体中文版专有出版权由 Pearson Education 培生教育出版亚洲有限公司授予电子工业出版社。未经出版者预先书面许可, 不得以任何方式复制或抄袭本书的任何部分。

本书简体中文版贴有 Pearson Education 培生教育出版集团激光防伪标签, 无标签者不得销售。

版权贸易合同登记号 图字

### 图书在版编目 (CIP)

C++标准库: 第2版 / (德) 约祖蒂斯 (Josuttis, N.M.) 著; 侯捷译. —北京: 电子工业出版社, 2015.6

书名原文: The C++ Standard Library: A Tutorial and Reference

ISBN 978-7-121-26089-6

I. ①C… II. ①约… ②侯… III. ①C语言—程序设计IV. ①TP312

中国版本图书馆CIP数据核字(2015)第103805号

责任编辑: 白涛

印刷: 三河市鑫金马印装有限公司

装订: 三河市鑫金马印装有限公司

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开本: 787×980 1/16 印张: 70.5 字数: 1500 千字

版次: 2015年6月第1版

印次: 2015年7月第2次印刷

定价: 186.00元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zlt@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。

*To those who care  
for people and mankind*

## 第 2 版译序

“不熟悉 C++ 标准库，任何人都称不上是高生产力的 C++ 程序员。”

这是我对每一位学生说的话。

自 C++98 (1998) 至 C++11 (2011)，这个语言有了相当的变化。而变化尤其剧烈——或者你也可以说攻城略地尤其大者——是其日益包山包海的标准库 (standard library)。如今，不但昔日闪亮招牌如 STL 容器、算法、迭代器等依然熠熠发光，更添加了 Tuple、Smart Pointer、Type Trait、Clock 和 Timer、Regular Expression、Concurrency 等众多组件。脚步一个没跟上，很可能同僚写出来的代码，在你眼中都要逐渐模糊逐渐陌生了。

面对陌生，我们最大的障碍在于心中的怯弱。To be or not to be, that's the question! 不要像哈姆雷特一样犹豫不决。面对前程，必须果敢。

愈来愈多的术语，特别是组合式长术语，为翻译带来极大困扰。关于术语翻译，我的态度如下：

- 技术书籍（尤其高级技术书籍）之中译，价值不在于体现全中文化阅读。直接呈现英文术语很多时候更能营造突出的视觉效果，进而有利阅读，形成深刻印象。
- 尽可能兼顾中英词性平衡。例如“对于矢量和 deque 两种容器……”就不如“对于 vector 和 deque 两种容器……”读来顺畅。
- 本书读者不可能受制于英文单词。我们要学习的是术语代表的技术，而非术语的中文名称。
- 与 C++ 关键字重合的术语多半不译，例如 static、template、class 等以及延伸出去的 member template、function template、class template 等等。
- 任何术语第一次出现（或相隔一长段距离后再出现）时多半中英并陈。但由于你可能采取跳跃式阅读，所谓“第一次出现”或“相隔一长段距离后再次出现”也许对你意义不大。
- STL 各数据结构名称皆不译，例如 array、vector、list、deque、hash table、map、set、stack、queue、tree……
- STL 六大组件的英文术语原欲全部保留，但由于频繁出现，对版面的中英比例形成视觉威胁，因此采用以下译名：container（容器），algorithm（算法），iterator（迭代器），adapter（适配器），function object（函数对象），allocator（分配器）。这些术语有时中文呈现，有时英文呈现，有时中英并陈，取决于当时语境。
- 全书按英文版页次编排，并因此得以保留原书极其详尽的索引。索引词条皆不译（因为书中相应保留了相当分量的英文术语）。

这是个知识爆炸、信息量也爆炸的时代。网络上如山似海的信息：文章、文件、文档、文献和繁若天星的论坛、群……很多都是免费的（因而似乎特别有吸引力），但我们的精力和时间才是更宝贵的东西。审阅并且评论过许许多多 C++ 标准库相关读物后，面对这本结构和细节都臻上乘的优秀教程，我强烈推荐给你。

侯捷

2014/10/20, White Rock

## 第 2 版序言

我从未想过本书第 1 版会卖得如此长久。现在，经过 12 年的等待，出新版本的时候到了——一个涵盖最新 C++11 标准的新版本。

注意，其意义比仅仅添加新子库更多更大。C++ 已经改变了，其标准库的几乎所有典型应用如今看起来都有那么一点不同。这不是因为语言出现巨大变化，而是许多小改变造成的结果，例如运用 rvalue reference 和 move semantic、range-based for loop、auto，以及新的 template 特性。因此，本书不但展现原标准库的补充特性及新成员，书中所有例子也都重新写过，或至少局部改写。尽管如此，为了支持那些仍然使用“旧”C++ 环境的程序员，本书会在遭遇 C++ 版本差异时加以描述。

我学习 C++11 有一段艰难的历程。大约两年前我开始看 C++11，但并未追随标准化的脚步，因此确实遭遇了理解上的麻烦。但是标准委员会的人帮助我描述及表现 C++11 新特性，一如它们如今呈现的样貌。

最后，请注意，本书有个困扰：虽然分量上由 800 页成长到超过 1100 页，我还是无法完整呈现 C++ 标准库。标准库中单单针对 C++11 就有大约 750 页的描述，都是十分精练的文字，没有太多说明。基于这个原因，我必须取舍哪些特性需要描述，并决定以怎样的粒度描述之。再一次，C++ 社区中的许多人帮助我做出决定。我的企图集中在普通程序员需要的东西。若干本书舍弃的部分被我做成一份补充篇章放在本书官网 <http://www.cppstdlib.com>。即便如此，你还是会 C++ 标准规格书中找到我未曾讨论的细节。

教学的艺术，并非体现在“呈现每一面”，而是一种“沙里淘金”的艺术，从中取得最大价值。但愿我是成功的。

## 第 2 版致谢

本书呈现的想法、观念、解决方案和示例，来自许多出处。过去数年里 C++ 社区引入了许多关于 C++ 的想法、概念、提案和改进，成就了 C++11 的一部分。因此，我要再一次感谢所有协助并支持我完成这个新版本的人。

首先我要感谢 C++ 社区和 C++ 标准委员会的每一个人。不只是为了他们添加新的语言特性和标准库特性所做的努力，也为了他们对我做出的解释，这需要大量耐心和热诚。

Scott Meyers 和 Anthony Williams 允许我使用他们的教材原稿和书籍原稿，使我得以找出许多有用却尚未公开出版的程序示例。

我也要感谢每一位审阅本书并给予反馈的人，他们是：Dave Abrahams、Alberto Ganesh Barbati、Pete Becker、Thomas Becker、Hans Boehm、Walter E. Brown、Paolo Carlini、Lawrence Crowl、Beman Dawes、Doug Gregor、David Grigsby、Pablo Halpern、Howard Hinnant、John Lakos、Broniek Kozicki、Dietmar Kühl、Daniel Krügler、Mat Marcus、Jens Maurer、Alisdair Meredith、Bartosz Milewski、P. J. Plauger、Tobias Schüle、Peter Sommerlad、Jonathan Wakely 和 Anthony Williams。

有一个人做出了特别杰出的工作。每当我有疑问，Daniel Krügler 几乎总是立即回复我，带着不可置信的精准和知识。每一个经历标准化进程（standardization process）的人都知道，他就是以此方式对待每一个人的。如果没有他，不论 C++ standard 或本书都不可能今天的有质量。

非常感谢我的 Addison-Wesley 编辑团队：Peter Gordon、Kim Boedigheimer、John Fuller 和 Anna Popick。不只因为他们的支持，也因为他们找出了忍耐和压力之间的美好平衡。文字编辑 Evelyn Pyle 和校对者 Diane Freed 做出了难以置信的工作，将我的德式英文翻译为美式英文。此外，感谢 Frank Mittelbach 解决我的 L<sup>A</sup>T<sub>E</sub>X 排版问题。

最后（但不是最不重要），我把所有感谢献给 Jutta Eckstein。Jutta 有惊人的能力促使并支持人们完成他们的理想和目标。虽然大多数人只在与她共事时才能有此经验，我却是如此荣幸地在每一天的生活中都能得此幸福。



# 第 1 版序言

一开始，我不过是想写一本篇幅不大的关于C++ 标准库的德文书（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++ 标准库，使你在遭遇所有（或几乎所有）编程问题之前就能够先获得解答。然而请注意，这不是一种完整描述C++ 标准库的所有方面的书籍，我通过“在 C++ 中利用标准库进行学习及编程”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开展，然后引入日常编程所必须了解的具体细节。为了帮助你理解这些概念和细节，书中提供了详尽的范例程序。

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

# 第 1 版致谢

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及公司，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 stream (串流) 和国际化 (他曾经仅仅为了好玩写了一个 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，过去数年里他向我提供了很多宝贵的反馈。

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质量获得巨大提升。由于名单太长，以下若有任何疏漏，还请见谅。英文版的审阅者包括 Chuck Allison、Greg Comeau、James A. Crotinger、Gabriel Dos Reis、Alan Ezust、Nathan Myers、Werner Mossner、Todd Veldhuizen、Chichiang Wan、Judy Ward 和 Thomas Wikehult。德文版的审阅者包括 Ralf Boecker、Dirk Herrmann、Dietmar Kühl、Edda Lörke、Herbert Scheubner、Dominik Strasser 和 Martin Weitzel。其他投入者包括 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质量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，那是个常态性的 C++ 难题讲解专栏，播出于 `comp.lang.c++.moderated`。

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++ standard 兼容的 C++ 标准库实现工作。感谢 Andreas Hommel 和 Metrowerks，他们完成了极具价值的 CodeWarrior 程序开发环境。感谢 GNU 和 egcs 编译器的所有开发者。感谢 Microsoft，他们完成了极具价值的 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，为了他那份极具价值的 ObjectSpace library 实现。

感谢 Addison Wesley Longman 公司里与我共同工作过的每一个人。他们包括 Janet Cocker、Mike Hendrickson、Debbie Lafferty、Marina Lang、Chanda Leary、Catherine Ohala、Marty Rabinowitz、Susanne Spitzer 和 Maureen Willard 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++ 社区的所有人，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。

Have fun and be human!

# 目录

第 2 版译序	xxi
第 2 版序言	xxiii
第 2 版致谢	xxiv
第 1 版序言	xxv
第 1 版致谢	xxvi
<b>1 关于本书</b>	<b>1</b>
1.1 缘起	1
1.2 阅读前的必要基础	2
1.3 本书风格与结构	2
1.4 如何阅读本书	4
1.5 目前发展情势	5
1.6 范例代码及额外信息	5
1.7 反馈	5
<b>2 C++ 及标准库简介</b>	<b>7</b>
2.1 C++ Standard 的历史	7
2.1.1 C++11 Standard 常见疑问	8
2.1.2 C++98 和 C++11 的兼容性	9
2.2 复杂度与 Big-O 标记	10
<b>3 语言新特性</b>	<b>13</b>
3.1 C++11 语言新特性	13
3.1.1 微小但重要的语法提升	13
3.1.2 以 auto 完成类型自动推导	14
3.1.3 一致性初始化 (Uniform Initialization) 与初值列 (Initializer List)	15

3.1.4	Range-Based for 循环	17
3.1.5	Move 语义和 Rvalue Reference	19
3.1.6	新式的字符串字面常量 (String Literal)	23
3.1.7	关键字 <code>noexcept</code>	24
3.1.8	关键字 <code>constexpr</code>	26
3.1.9	崭新的 Template 特性	26
3.1.10	Lambda	28
3.1.11	关键字 <code>decltype</code>	32
3.1.12	新的函数声明语法 (New Function Declaration Syntax)	32
3.1.13	带领域的 (Scoped) Enumeration	32
3.1.14	新的基础类型 (New Fundamental Data Type)	33
3.2	虽旧犹新的语言特性	33
3.2.1	基础类型的明确初始化 (Explicit Initialization for Fundamental Type)	37
3.2.2	<code>main()</code> 定义式	37
<b>4</b>	<b>一般概念</b>	<b>39</b>
4.1	命名空间 (Namespace) <code>std</code>	39
4.2	头文件 (Header File)	40
4.3	差错和异常 (Error and Exception) 的处理	41
4.3.1	标准的 Exception Class (异常类)	41
4.3.2	异常类 (Exception Class) 的成员	44
4.3.3	以 Class <code>exception_ptr</code> 传递异常	52
4.3.4	抛出标准异常	53
4.3.5	自标准异常类派生	54
4.4	Callable Object (可被调用的对象)	54
4.5	并发与多线程	55
4.6	分配器 (Allocator)	57
<b>5</b>	<b>通用工具</b>	<b>59</b>
5.1	Pair 和 Tuple	60
5.1.1	Pair	60
5.1.2	Tuple (不定数的值组)	68
5.1.3	Tuple 的输入/输出	74
5.1.4	<code>tuple</code> 和 <code>pair</code> 转换	75
5.2	Smart Pointer (智能指针)	76
5.2.1	Class <code>shared_ptr</code>	76
5.2.2	Class <code>weak_ptr</code>	84
5.2.3	误用 Shared Pointer	89

5.2.4	细究 Shared Pointer 和 Weak Pointer	92
5.2.5	Class unique_ptr	98
5.2.6	细究 Class unique_ptr	110
5.2.7	Class auto_ptr	113
5.2.8	Smart Pointer 结语	114
5.3	数值的极值 (Numeric Limit)	115
5.4	Type Trait 和 Type Utility	122
5.4.1	Type Trait 的目的	122
5.4.2	细究 Type Trait	125
5.4.3	Reference Wrapper (外覆器)	132
5.4.4	Function Type Wrapper (外覆器)	133
5.5	辅助函数	134
5.5.1	挑选最小值和最大值	134
5.5.2	两值互换 (Swapping)	136
5.5.3	增补的“比较操作符” (Comparison Operator)	138
5.6	Class ratio<> 的编译期分数运算	140
5.7	Clock 和 Timer	143
5.7.1	Chrono 程序库概观	143
5.7.2	Duration (时间段)	144
5.7.3	Clock (时钟) 和 Timepoint (时间点)	149
5.7.4	C 和 POSIX 提供的 Date/Time 函数	157
5.7.5	以计时器停滞线程 (Blocking with Timer)	160
5.8	头文件 <cstdlib>、<stdlib> 和 <cstring>	161
5.8.1	<cstdlib> 内的各项定义	161
5.8.2	<stdlib> 内的各种定义	162
5.8.3	<cstring> 中的定义式	163
<b>6</b>	<b>标准模板库</b>	<b>165</b>
6.1	STL 组件 (Component)	165
6.2	容器 (Container)	167
6.2.1	序列式容器 (Sequence Container)	169
6.2.2	关联式容器 (Associative Container)	177
6.2.3	无序容器 (Unordered Container)	180
6.2.4	关联式数组 (Associative Array)	185
6.2.5	其他容器	187
6.2.6	容器适配器 (Container Adapter)	188
6.3	迭代器 (Iterator)	188

6.3.1	关联式 (Associative) 及无序 (Unordered) 容器的更多实例 . . . . .	193
6.3.2	迭代器种类 (Iterator Category) . . . . .	198
6.4	算法 (Algorithm) . . . . .	199
6.4.1	区间 (Range) . . . . .	203
6.4.2	处理多重区间 (Multiple Ranges) . . . . .	207
6.5	迭代器之适配器 (Iterator Adapter) . . . . .	210
6.5.1	Insert Iterator (安插型迭代器) . . . . .	210
6.5.2	Stream Iterator (串流迭代器) . . . . .	212
6.5.3	Reverse Iterator (反向迭代器) . . . . .	214
6.5.4	Move Iterator (搬移迭代器) . . . . .	216
6.6	用户自定义的泛型函数 (User-Defined Generic Function) . . . . .	216
6.7	更易型算法 (Manipulating Algorithm) . . . . .	217
6.7.1	移除 (Removing) 元素 . . . . .	218
6.7.2	更易 Associative (关联式) 和 Unordered (无序) 容器 . . . . .	221
6.7.3	算法 vs. 成员函数 . . . . .	223
6.8	以函数作为算法的实参 . . . . .	224
6.8.1	以函数作为算法实参的实例示范 . . . . .	224
6.8.2	判断式 (Predicate) . . . . .	226
6.9	使用 Lambda . . . . .	229
6.10	函数对象 (Function Object) . . . . .	233
6.10.1	定义一个函数对象 . . . . .	233
6.10.2	预定义的函数对象 . . . . .	239
6.10.3	Binder . . . . .	241
6.10.4	函数对象 vs. Lambda . . . . .	243
6.11	容器内的元素 . . . . .	244
6.11.1	容器元素的必要条件 . . . . .	244
6.11.2	Value 语义 vs. Reference 语义 . . . . .	245
6.12	STL 内部的错误和异常 . . . . .	245
6.12.1	错误处理 (Error Handling) . . . . .	246
6.12.2	异常处理 (Exception Handling) . . . . .	248
6.13	扩展 STL . . . . .	250
6.13.1	整合更多 Type . . . . .	250
6.13.2	派生自 STL Type . . . . .	251
<b>7</b>	<b>STL 容器</b> . . . . .	<b>253</b>
7.1	容器的共通能力和共通操作 . . . . .	254
7.1.1	容器的共通能力 . . . . .	254

7.1.2	容器的共通操作	254
7.1.3	容器提供的类型	260
7.2	Array	261
7.2.1	Array 的能力	261
7.2.2	Array 的操作	263
7.2.3	把array当成 C-Style Array	267
7.2.4	异常处理 (Exception Handling)	268
7.2.5	Tuple 接口	268
7.2.6	Array 运用实例	268
7.3	Vector	270
7.3.1	Vector 的能力	270
7.3.2	Vector 的操作	273
7.3.3	将 Vector 当作 C-Style Array 使用	278
7.3.4	异常处理 (Exception Handling)	278
7.3.5	Vector 使用实例	279
7.3.6	Class <code>vector&lt;bool&gt;</code>	281
7.4	Deque	283
7.4.1	Deque 的能力	284
7.4.2	Deque 的操作函数	285
7.4.3	Exception Handling	288
7.4.4	Deque 运用实例	288
7.5	List	290
7.5.1	List 的能力	290
7.5.2	List 的操作	291
7.5.3	异常处理 (Exception Handling)	296
7.5.4	List 运用实例	298
7.6	Forward List	300
7.6.1	Forward List 的能力	300
7.6.2	Forward List 的操作	302
7.6.3	异常处理 (Exception Handling)	311
7.6.4	Forward List 运用实例	312
7.7	Set 和 Multiset	314
7.7.1	Set 和 Multiset 的能力	315
7.7.2	Set 和 Multiset 的操作函数	316
7.7.3	异常处理 (Exception Handling)	325



7.7.4	Set 和 Multiset 运用实例	325
7.7.5	运行期指定排序准则	328
7.8	Map 和 Multimap	331
7.8.1	Map 和 Multimap 的能力	332
7.8.2	Map 和 Multimap 的操作函数	333
7.8.3	将 Map 视为关联式数组 (Associative Array)	343
7.8.4	异常处理 (Exception Handling)	345
7.8.5	Map 和 Multimap 运用实例	345
7.8.6	综合实例: 运用 Map、String 并于运行期指定排序准则	351
7.9	无序容器 (Unordered Container)	355
7.9.1	Unordered 容器的能力	357
7.9.2	创建和控制 Unordered 容器	359
7.9.3	Unordered 容器的其他操作	367
7.9.4	Bucket 接口	374
7.9.5	使用 Unordered Map 作为 Associative Array	374
7.9.6	异常处理 (Exception Handling)	375
7.9.7	Unordered 容器的运用实例	375
7.10	其他 STL 容器	385
7.10.1	String 作为一种 STL 容器	385
7.10.2	C-Style Array 作为一种 STL 容器	386
7.11	实现 Reference 语义	388
7.12	各种容器的使用时机	392
<b>8</b>	<b>细探 STL 容器成员</b>	<b>397</b>
8.1	容器内的类型	397
8.2	创建、复制和销毁 (Create, Copy, and Destroy)	400
8.3	非更易型操作 (Nonmodifying Operation)	403
8.3.1	大小相关操作 (Size Operation)	403
8.3.2	元素比较 (Comparison Operation)	404
8.3.3	Associative 和 Unordered 容器特有的非更易型操作	404
8.4	赋值 (Assignment)	406
8.5	元素直接访问 (Direct Element Access)	408
8.6	“产出迭代器”之各项操作	410
8.7	安插和移除 (Inserting and Removing) 元素	411
8.7.1	安插单一元素 (Inserting Single Element)	411
8.7.2	安插多重元素 (Inserting Multiple Elements)	416