



C++ 17



Professional C++, 4th Edition

C++高级编程

(第4版)

[美] 马克·葛瑞格尔(Marc Gregoire) 著
徐志超 曹瑜 译

清华大学出版社



C++高级编程

(第4版)

[美] 马克·葛瑞格尔(Marc Gregoire) 著

徐志超 曹瑜

译

清华大学出版社

北 京

Professional C++, 4th Edition

Marc Gregoire

EISBN: 978-1-119-42130-6

Copyright © 2018 by John Wiley & Sons, Inc., Indianapolis, Indiana

All Rights Reserved. This translation published under license.

Trademarks: Wiley, the Wiley logo, Wrox, the Wrox logo, Programmer to Programmer, and related trade dress are trademarks or registered trademarks of John Wiley & Sons, Inc. and/or its affiliates, in the United States and other countries, and may not be used without written permission. Visual Studio is a registered trademark of Microsoft Corporation. All other trademarks are the property of their respective owners. John Wiley & Sons, Inc., is not associated with any product or vendor mentioned in this book.

本书中文简体字版由 Wiley Publishing, Inc. 授权清华大学出版社出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

北京市版权局著作权合同登记号 图字：01-2018-4099

本书封面贴有 Wiley 公司防伪标签，无标签者不得销售。

版权所有，侵权必究。侵权举报电话：010-62782989 13701121933

图书在版编目(CIP)数据

C++高级编程：第4版 / (美)马克·葛瑞格尔(Marc Gregoire) 著；徐志超，曹瑜 译。—北京：清华大学出版社，2019

书名原文：Professional C++, 4th Edition

ISBN 978-7-302-52631-5

I. ①C… II. ①马… ②徐… ③曹… III. ①C++语言—程序设计 IV. ①TP312.8

中国版本图书馆 CIP 数据核字(2019)第 046888 号

责任编辑：王 军

装帧设计：孔祥峰

责任校对：成凤进

责任印制：刘海龙

出版发行：清华大学出版社

网 址：<http://www.tup.com.cn>, <http://www.wqbook.com>

地 址：北京清华大学学研大厦 A 座 邮 编：100084

社 总 机：010-62770175 邮 购：010-62786544

投稿与读者服务：010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈：010-62772015, zhiliang@tup.tsinghua.edu.cn

印 装 者：清华大学印刷厂

经 销：全国新华书店

开 本：190mm×260mm 印 张：46.75 字 数：1580 千字

版 次：2019 年 4 月第 1 版 印 次：2019 年 4 月第 1 次印刷

定 价：138.00 元

产品编号：078948-01

作者简介

Marc Gregoire是一位Microsoft Visual C++ MVP, 是一名软件工程师和开发人员, 是比利时C++用户组的创始人, 现供职于Nikon Metrology, 负责开发3D激光扫描软件。Marc曾在Siemens 和Nokia Siemens Networks开发关键的2G和3G电信软件。Marc是本书前两版的作者, 也曾担任其他多本书籍的技术编辑。

译者序

在中国大陆的程序员圈子中，常将 C++ 读作“C 加加”；而西方国家的程序员常将 C++ 读作“C plus plus”或“CPP”。C++ 最初作为 C 语言的增强版出现，此后不断融入新特性，比如虚函数、运算符重载、多重继承、模板、异常、RTTI、结构化绑定、嵌套的名称空间等。

多年来，C++ 都是编写性能卓越、功能强大的企业级面向对象程序的事实标准语言。尽管 C++ 语言已经风靡全球，但这门语言却极难完全掌握，即使是经验丰富的 C++ 程序员也未必了解 C++ 中某些很有用的特性。

要成为一名专业的 C++ 程序员，必须扎实理解 C++ 语言的工作原理，了解不同编程方法论和软件开发过程——从设计和编码，到测试、调试等，以便更好地和团队协作。要知道良好设计的重要性，领略面向对象编程的理论，掌握卓越的调试技能，探索可重用的库和常用的设计模式，了解可重用的思想，以大幅提升日常工作效率。

这本讲解 C++17 的著作将帮助读者全面透彻地掌握 C++ 语言的功能，包罗 C++ 语言的一切，分享真实范例，展现 C++17 的新工具和功能，详述如何在真实世界中使用 C++，揭示最新版 C++ 带来的显著变化，解密 C++ 中鲜为人知的特性，探索编程方法论、可重用的设计模式和良好的编程风格，阐述如何设计可充分利用 C++ 语言功能的高效解决方案。本书深入探讨 C++ 语言功能集的更复杂元素，并讲解避开常见陷阱的技巧。本书提供详尽的编程指南，紧贴实际，是编程人员深入挖掘 C++ 的理想工具。

本书包括 5 部分。第 I 部分是 C++ 基础速成教程，第 II 部分介绍 C++ 设计方法论，第 III 部分从专业角度分析 C++ 编程技术，第 IV 部分讲解如何真正掌握 C++ 的高级功能，第 V 部分重点介绍 C++ 软件工程技术。最后的附录 A 列出了在 C++ 技术面试中取得成功的指南，附录 B 列出带注解的参考文献，附录 C 总结标准的 C++ 头文件，附录 D 则简要介绍 UML。

对于这本经典之作，译者在翻译过程中力求忠于原文，再现原文风格，但是鉴于译者水平有限，错误和失误在所难免，如有任何意见和建议，请不吝指正。除封面署名的译者外，何益枝、吴秀莲、张继春、段治勋、郑玉花、兰翠平、段国鑫也参与了本书的翻译工作。

最后，希望读者通过阅读本书能早日步入 C++ 编程的殿堂，领略 C++ 语言之美！

作者简介

Marc Gregoire 是一名软件工程师，毕业于比利时鲁文大学，拥有计算机科学工程硕士学位。之后，他在鲁文大学获得人工智能专业的优等硕士学位。完成学业后，他开始为软件咨询公司 Ordina Belgium 工作。他曾在 Siemens 和 Nokia Siemens Networks 为大型电信运营商提供有关在 Solaris 上运行关键 2G 和 3G 软件的咨询服务。这份工作要求与来自南美、美国、欧洲、中东、非洲和亚洲的国际团队合作。Marc 目前担任 Nikon Metrology (www.nikonmetrology.com) 的软件架构师；Nikon Metrology 是 Nikon 的分公司，是领先的精密光学仪器和 3D 扫描软件供应商。

Marc 的主要技术专长是 C/C++，特别是 Microsoft VC++ 和 MFC 框架。他还擅长在 Windows 和 Linux 平台上开发 24x7 小时运行的 C++ 程序，例如 KNX/EIB 家庭自动化监控软件。除了 C/C++ 之外，Marc 还喜欢 C#，并且会用 PHP 创建网页。

2007 年 4 月，他凭借 Visual C++ 方面的专业技能，获得了微软年度 MVP 称号。

Marc 还是比利时 C++ 用户组 (www.becpp.org) 的创始人，是 C++ *Standard Library Quick Reference* (Apress) 一书的作者，以及多家出版社出版的多本书籍的技术编辑，是 CodeGuru 论坛上的活跃分子 (id 为 Marc G)。Marc 还在 www.nuonsoft.com/blog/ 上维护了一个博客，他热爱旅游和烹饪。

技术编辑简介

Peter Van Weert 是一名软件工程师。他的主要爱好和特长是 C++、编程语言、算法和数据结构。他毕业于比利时勒芬大学，勒芬大学的考试团认定 **Peter** 在计算机科学方面获得了全优成绩，因此授予他硕士学位。2010 年，他在勒芬大学获得博士学位，他的博士论文围绕基于规则的编程语言(主要是 Java)的有效编译展开。在攻读博士学位期间，他还担任面向对象的分析与设计、Java 编程和声明性编程语言等课程的助教。

完成学业后，**Peter** 供职于 Nikon Metrology，负责利用 C++ 开发 3D 激光扫描软件和点云检测软件。在 2017 年，**Peter** 加盟 Nobel Biocare 的数字化牙科软件研发部门。在他的职业生涯中，**Peter** 精通 C++ 开发以及超大型软件库的管理、重构和调试。**Peter** 也熟悉软件开发过程的所有方面，包括功能和技术需求分析，以及基于敏捷和 Scrum 的项目管理和团队管理。

Peter 是比利时 C++ 用户组的成员，并且经常在该用户组发表技术演讲。他曾参与撰写以下两本由 Apress 出版的技术书籍：*C++ Standard Library Quick Reference* 和 *Beginning C++ (5th edition)*。

致 谢

在此感谢 John Wiley & Sons 和 Wrox 出版社的编辑和产品团队，感谢他们的支持。特别感谢 Wiley 的执行编辑 Jim Minatel 给我撰写本书的机会，感谢 Wiley 的高级项目编辑 Adaobi Obi Tulton 对这个项目的支持。感谢文稿编辑 Marylouise Wiack 对文字的梳理，保证术语前后一致，以及确保语法正确无误。

特别感谢技术编辑 Peter Van Weert，感谢他一丝不苟地对本书进行审阅。他提出了很多具有建设性的评论和意见，将本书的质量推进到更高水平。

当然，还要感谢我的父母、兄长和妻子给予的支持和关爱，你们的支持对本书的完成至关重要。也衷心感谢 Nikon Metrology 在我完成本项目期间给予的支持。

最后，我要感谢各位亲爱的读者，希望你们通过本书能学习到高级 C++ 软件开发知识。

多年来，C++都是编写性能卓越、功能强大的企业级面向对象程序的事实标准语言。尽管 C++语言已经风靡全球，但这种语言却非常难完全掌握。专业 C++程序员使用一些简单但高效的技术，这些技术并未出现在传统教材中；即使是经验丰富的 C++程序员，也未必完全了解 C++中某些很有用的特性。

编程书籍往往重点描述语言的语法，而不是语言在真实世界中的应用。典型的 C++教材在每一章中介绍语言中的大部分知识，讲解语法并列举示例。本书不遵循这种模式。本书并不讲解语言的大量细节并给出少量真实世界的场景，而是教你如何在真实世界中使用 C++。本书还会讲解一些鲜为人知的让编程更简单的特性，以及区分编程新手和专业程序员的编程技术。

读者对象

就算使用 C++已经多年，也仍可能不熟悉 C++的一些高级特性，或仍不具有使用这门语言的完整能力。也许你编写过实用的 C++代码，但还想学习更多有关 C++中设计和良好编程风格的内容。也许你是 C++新手，想在入门时就掌握“正确”的编程方式。本书能满足上述需求，将你的 C++技能提升到专业水准。

因为本书专注于从对 C++具有基本或中等了解水平蜕变为一名专业 C++程序员的过程，所以本书假设你对该语言具有一定程度的认识。第 1 章涵盖 C++的一些基础知识，可以当成复习材料，但是不能替代实际的语言培训和语言使用手册。如果刚开始接触 C++，但有很丰富的 C、Java 或 C#语言经验，那么应该能从第 1 章获得所需的大部分知识。

不管属于哪种情况，都应该具有很好的编程基础。应该知道循环、函数和变量。应该知道如何组织一个程序，而且应该熟悉基本技术，例如递归。应该了解一些常见的数据结构(例如队列)以及有用的算法(例如排序和搜索)。不需要预先了解有关面向对象编程的知识——这是第 5 章讲解的内容。

还应该熟悉开发代码时使用的编译器。稍后将简要介绍 Microsoft Visual C++和 GCC 这两种编译器。要了解其他编译器，请参阅编译器自带的指南。

本书主要内容

阅读本书是学习 C++语言的一种方法，通过阅读本书既能提升编码质量，又能提升编程效率。本书贯穿对 C++17 新特性的讨论。这些新的 C++17 特性并不是分散在各章中，而是穿插于全书，在有必要的情况下，所有例子都已更新为使用这些新特性。

本书不仅讲解 C++语法和语言特性，还强调编程方法论、可重用的设计模式以及良好的编程风格。本书讲解的方法论覆盖整个软件开发过程——从设计和编码，到调试以及团队协作。这种方法可让你掌握 C++语言及其独有特性，还能在大型软件开发中充分利用 C++语言的强大功能。

想象一下有人学习了 C++的所有语法但没有见过一个 C++例子的情形。他所了解的知识会让他处于非常危险的境地。如果没有示例的引导，他可能会认为所有源代码都要放在程序的 `main()`函数中，还有可能认为所有变量都应该为全局变量——这些都不是良好的编程实践。

专业的 C++程序员除了理解语法外，还要正确理解语言的使用方式。他们知道良好设计的重要性、面向对

象编程的理论以及使用现有库的最佳方式。他们还开发了大量有用的代码并了解可重用的思想。

通过阅读和理解本书的内容,你也能成为一名专业的C++程序员。你在C++方面的知识会得到扩充,将接触到鲜为人知和常被误解的语言特性。你还将领略面向对象设计,掌握卓越的调试技能。最重要的或许是,通过阅读本书,你的头脑中有了大量“可重用”思想,可将这种思想贯彻到日常工作中。

有很多好的理由让你努力成为一名专业的C++程序员,而非只是泛泛了解C++。了解语言的真正工作原理有助于提升代码质量。了解不同的编程方法论和过程可让你更好地和团队协作。探索可重用的库和常用的设计模式可提升日常工作效率,并帮助避免白费力气地重复工作。所有这些学习课程都在帮助你成为更优秀的程序员,同时成为更有价值的雇员。

本书结构

本书包括5部分。

第I部分“专业的C++简介”是C++基础速成教程,能确保读者掌握C++的基础知识。在速成教程后,第I部分深入讨论字符串和字符串视图的使用;因为字符串在示例中应用广泛。第I部分的最后一章介绍如何编写清晰易读的C++代码。

第II部分“专业的C++软件设计”介绍C++设计方法论。你会了解到设计的重要性、面向对象方法论和代码重用的重要性。

第III部分“专业的C++编码方法”从专业角度概述C++技术。你将学习在C++中管理内存的最佳方式,如何创建可重用的类,以及如何利用重要的语言特性,例如继承。你还会学习这门语言的一些不同寻常之处、输入输出技术、错误处理、字符串本地化和正则表达式的使用,讨论如何实现运算符重载,如何编写模板。这一部分还讲解C++标准库,包括容器、迭代器和算法。你还会学习C++标准中的其他一些库,例如处理时间、随机数和文件系统的库。

第IV部分“掌握C++的高级特性”讲解如何最大限度地使用C++。本书这一部分揭示C++中神秘的部分,并描述如何使用这些更高级的特性。你将学习如何定制和扩充标准库以满足自己的需求、高级模板编程的细节(包括模板元编程),以及如何通过多线程编程来充分利用多处理器和多核系统。

第V部分“C++软件工程”重点介绍如何编写企业级质量的软件。相关的主题如下:当今编程组织使用的工程实践,如何编写高效的C++代码,软件测试概念(如单元测试和回归测试),C++程序的调试技术,如何在自己的代码中融入设计技术、框架和概念性的面向对象设计模式,跨语言和跨平台代码的解决方案,等等。

本书最后是4个附录。附录A列出在C++技术面试中取得成功的指南,附录B是带注解的参考文献列表,附录C总结C++标准中的头文件,附录D简要介绍UML(Unified Modeling Language,统一建模语言)。

本书没有列出C++中每个类、方法和函数的参考。Peter Van Weert和Marc Gregoire撰写的*C++ Standard Library Quick Reference*是C++标准库提供的所有重要数据结构、算法和函数的浓缩版。附录B列出了更多参考资料。下面是两个很好的在线参考。

www.cppreference.com

可使用这个在线参考,也可下载其离线版本,在没有连接到互联网时使用。

www.cplusplus.com/reference/

本书正文中提到“标准库参考资料”时,就是指上述C++参考资料。

使用本书的条件

要使用本书,只需要一台带有C++编译器的计算机。本书只关注C++中的标准部分,而没有任何编译器厂商相关的扩展。

本书包含C++17标准引入的新特性。在撰写本书时,有些编译器还不能完全支持C++17的所有新特性。

可使用任意 C++ 编译器。如果还没有 C++ 编译器，可下载一个免费的。这有许多选择。例如，对于 Windows，可下载 Microsoft Visual Studio 2017 Community Edition，这个版本免费且包含 Visual C++；对于 Linux，可使用 GCC 或 Clang，它们也是免费的。

下面将简要介绍如何使用 Visual C++ 和 GCC。可参阅相关的编译器文档来了解更多信息。

Microsoft Visual C++

首先需要创建一个项目。启动 VC++，单击 File | New | Project，在左边的项目模板树中选择 Visual C++ | Win32，再在窗口中间的列表中选择 Win32 Console Application(或 Windows Console Application)模板。在底部指定项目的名称、保存位置，单击 OK。

这会打开一个向导，单击 Next 按钮，选择 Console Application 和 Empty Project，再单击 Finish 按钮。注意，你可能看不到向导，具体取决于使用的 VC++ 2017 版本。相反，将自动创建一个新的项目，其中包含 4 个文件：stdafx.h、stdafx.cpp、targetver.h 和 <projectname>.cpp。如果遇到这种情况，而你想要编译源代码文件(取自从配套网站下载的本书源代码压缩文件)，则必须在 Solution Explorer(选择 View | Solution Explorer)中选择这些文件，然后删除它们。

加载新项目后，就会在 Solution Explorer 中看到项目文件列表。如果这个停靠窗口不可见，可选择 View | Solution Explorer。在 Solution Explorer 中右击项目名，再选择 Add | New Item 或 Add | Existing Item，就可以给项目添加新文件或已有文件。

使用 Build | Build Solution 编译代码。没有编译错误后，就可以使用 Debug | Start Debugging 运行了。

如果程序在查看输出之前就退出了，可使用 Debug | Start without Debugging。这会在程序末尾暂停，以便查看输出。

在撰写本书期间，Visual C++ 2017 尚未自动启用 C++17 功能。要启用 C++17 功能，可在 Solution Explorer 窗口中右击项目，然后单击 Properties。在 Properties 窗口中，选择 Configuration Properties | C/C++ | Language，根据使用的 Visual C++ 版本，将 C++ Language Standard 选项设置为 ISO C++17 Standard 或 ISO C++ Latest Draft Standard。仅当项目至少包含一个.cpp 文件时，才能访问这些选项。

Visual C++ 支持“预编译的头文件”，这个话题超出了本书的讨论范围。通常而言，如果编译器支持的话，建议使用预编译的头文件。但是，从本书网站下载的源代码文件不使用预编译的头文件，因此，只有禁用这项功能才能使这些代码正确编译。在 Solution Explorer 窗口中右击项目，选择 Properties。在 Properties 窗口中，找到 Configuration Properties | C/C++ | Precompiled Headers，将 Precompiled Header 选项设置为 Not Using Precompiled Headers。

GCC

用自己喜欢的任意文本编辑器创建源代码，保存到一个目录下。要编译代码，可打开一个终端，运行如下命令，指定要编译的所有.cpp 文件：

```
gcc -lstdc++ -std=c++17 -o <executable_name> <source1.cpp> [source2.cpp ...]
```

-std=c++17 用于告诉 GCC 启用 C++17 支持。

例如，可切换到包含代码的目录，运行如下命令来编译第 1 章的 AirlineTicket 示例：

```
gcc -lstdc++ -std=c++17 -o AirlineTicket AirlineTicket.cpp AirlineTicketTest.cpp
```

没有编译错误后，就可以使用如下命令运行了：

```
./AirlineTicket
```

勘误表

尽管我们已经尽了各种努力来保证文章或代码中不出现错误，但错误总是难免的。如果在本书中找到错误，例如拼写错误或代码错误，请告诉我们，我们将非常感激。通过勘误表，可以让其他读者避免受挫，当然，这还有助于提供更高质量的信息。

· 请给 wkservice@vip.163.com 发电子邮件，我们就会检查你提供的信息，如果是正确的，我们将在本书的后续版本中采用。

要在网站上找到本书的勘误表，可登录 <http://www.wrox.com>，通过 Search 工具或书名列表查找本书，然后在本书的细目页面上，单击 Book Errata 链接。在这个页面上可查看 Wrox 编辑已提交和粘贴的所有勘误项。完整的图书列表还包括每本书的勘误表，网址是 www.wrox.com/misc-pages/booklist.shtml。

源代码

读者在学习本书中的示例时，可以手动输入所有代码，也可使用本书附带的源代码文件。本书使用的所有源代码都可以从本书合作站点 www.wiley.com/go/proc++4e 下载。

另外，也可进入 <http://www.wrox.com/dynamic/books/download.aspx> 上的 Wrox 代码下载主页，查看本书和其他 Wrox 图书的所有代码。

还可通过扫描本书封底的二维码来下载源代码。

提示：

由于许多图书的书名都十分类似，因此按 ISBN 搜索是最简单的，本书英文版的 ISBN 是 978-1-119-42130-6。

下载代码后，只需要用自己喜欢的解压缩软件进行解压缩即可。

目 录

第 I 部分 专业的 C++ 简介

第 1 章 C++ 和标准库速成	2
1.1 C++ 基础知识	2
1.1.1 小程序 “hello world”	3
1.1.2 名称空间	5
1.1.3 字面量	6
1.1.4 变量	7
1.1.5 运算符	8
1.1.6 类型	10
1.1.7 条件语句	12
1.1.8 逻辑比较运算符	14
1.1.9 函数	15
1.1.10 C 风格的数组	16
1.1.11 std::array	17
1.1.12 std::vector	17
1.1.13 结构化绑定	18
1.1.14 循环	18
1.1.15 初始化列表	19
1.1.16 这些都是基础	19
1.2 深入研究 C++	20
1.2.1 C++ 中的字符串	20
1.2.2 指针和动态内存	20
1.2.3 const 的多种用法	24
1.2.4 引用	24
1.2.5 异常	25
1.2.6 类型推断	26
1.3 作为面向对象语言的 C++	27
1.3.1 定义类	27
1.3.2 使用类	29
1.4 统一初始化	29
1.5 标准库	31
1.6 第一个有用的 C++ 程序	31
1.6.1 雇员记录系统	32
1.6.2 Employee 类	32
1.6.3 Database 类	34

1.6.4 用户界面	36
1.6.5 评估程序	38
1.7 本章小结	38
第 2 章 使用 string 和 string_view	39
2.1 动态字符串	39
2.1.1 C 风格的字符串	39
2.1.2 字符串字面量	41
2.1.3 C++ std::string 类	42
2.1.4 std::string_view 类	46
2.1.5 非标准字符串	47
2.2 本章小结	47
第 3 章 编码风格	48
3.1 良好外观的重要性	48
3.1.1 事先考虑	48
3.1.2 良好风格的元素	49
3.2 为代码编写文档	49
3.2.1 使用注释的原因	49
3.2.2 注释的风格	52
3.3 分解	55
3.3.1 通过重构分解	56
3.3.2 通过设计来分解	56
3.3.3 本书中的分解	56
3.4 命名	56
3.4.1 选择恰当的名称	57
3.4.2 命名约定	57
3.5 使用具有风格的语言特性	59
3.5.1 使用常量	59
3.5.2 使用引用代替指针	59
3.5.3 使用自定义异常	59
3.6 格式	60
3.6.1 关于大括号对齐的争论	60
3.6.2 关于空格和圆括号的争论	61
3.6.3 空格和制表符	61
3.7 风格的挑战	61

3.8 本章小结	62
----------	----

第 II 部分 专业的 C++ 软件设计

第 4 章 设计专业的 C++ 程序	64
4.1 程序设计概述	64
4.2 程序设计的重要性	65
4.3 C++ 设计的特点	66
4.4 C++ 设计的两个原则	67
4.4.1 抽象	67
4.4.2 重用	68
4.5 重用代码	69
4.5.1 关于术语的说明	69
4.5.2 决定是否重用代码	70
4.5.3 重用代码的策略	71
4.5.4 绑定第三方应用程序	74
4.5.5 开放源代码库	75
4.5.6 C++ 标准库	76
4.6 设计一个国际象棋程序	76
4.6.1 需求	76
4.6.2 设计步骤	77
4.7 本章小结	80
第 5 章 面向对象设计	82
5.1 过程化的思考方式	82
5.2 面向对象思想	83
5.2.1 类	83
5.2.2 组件	83
5.2.3 属性	83
5.2.4 行为	84
5.2.5 综合考虑	84
5.3 生活在对象世界里	85
5.3.1 过度使用对象	85
5.3.2 过于通用的对象	85
5.4 对象之间的关系	86
5.4.1 “有一个”关系	86
5.4.2 “是一个”关系(继承)	87
5.4.3 “有一个”与“是一个”的区别	88
5.4.4 not-a 关系	90
5.4.5 层次结构	91
5.4.6 多重继承	91
5.4.7 混入类	92
5.5 抽象	93
5.5.1 接口与实现	93
5.5.2 决定公开的接口	93

5.5.3 设计成功的抽象	94
5.6 本章小结	95

第 6 章 设计可重用代码	96
6.1 重用哲学	96
6.2 如何设计可重用代码	97
6.2.1 使用抽象	97
6.2.2 构建理想的重用代码	98
6.2.3 设计有用的接口	102
6.2.4 SOLID 原则	106
6.3 本章小结	106

第 III 部分 专业的 C++ 编码方法

第 7 章 内存管理	108
7.1 使用动态内存	108
7.1.1 如何描绘内存	109
7.1.2 分配和释放	110
7.1.3 数组	111
7.1.4 使用指针	116
7.2 数组-指针的对偶性	117
7.2.1 数组就是指针	117
7.2.2 并非所有指针都是数组	119
7.3 低级内存操作	119
7.3.1 指针运算	119
7.3.2 自定义内存管理	120
7.3.3 垃圾回收	120
7.3.4 对象池	121
7.4 智能指针	121
7.4.1 unique_ptr	122
7.4.2 shared_ptr	124
7.4.3 weak_ptr	125
7.4.4 移动语义	126
7.4.5 enable_shared_from_this	127
7.4.6 旧的、过时的/取消的 auto_ptr	127
7.5 常见的内存陷阱	127
7.5.1 分配不足的字符串	127
7.5.2 访问内存越界	128
7.5.3 内存泄漏	128
7.5.4 双重删除和无效指针	131
7.6 本章小结	131
第 8 章 熟悉类和对象	132
8.1 电子表格示例介绍	132
8.2 编写类	133

8.2.1	类定义	133	10.2	使用继承重用代码	194
8.2.2	定义方法	135	10.2.1	WeatherPrediction类	194
8.2.3	使用对象	137	10.2.2	在派生类中添加功能	195
8.3	对象的生命周期	138	10.2.3	在派生类中替换功能	196
8.3.1	创建对象	138	10.3	利用父类	196
8.3.2	销毁对象	149	10.3.1	父类构造函数	196
8.3.3	对象赋值	149	10.3.2	父类的析构函数	197
8.3.4	编译器生成的复制构造函数和复制赋值运算符	151	10.3.3	使用父类方法	198
8.3.5	复制和赋值的区别	151	10.3.4	向上转型和向下转型	200
8.4	本章小结	153	10.4	继承与多态性	201
第9章	精通类与对象	154	10.4.1	回到电子表格	201
9.1	友元	154	10.4.2	设计多态性的电子表格单元格	201
9.2	对象的动态内存分配	155	10.4.3	SpreadsheetCell基类	202
9.2.1	Spreadsheet类	155	10.4.4	独立的派生类	203
9.2.2	使用析构函数释放内存	157	10.4.5	利用多态性	204
9.2.3	处理复制和赋值	158	10.4.6	考虑将来	205
9.2.4	使用移动语义处理移动	162	10.5	多重继承	206
9.2.5	零规则	167	10.5.1	从多个类继承	206
9.3	与方法有关的更多内容	167	10.5.2	名称冲突和歧义基类	207
9.3.1	静态方法	167	10.6	有趣而晦涩的继承问题	209
9.3.2	const方法	168	10.6.1	修改重写方法的特征	209
9.3.3	方法重载	169	10.6.2	继承的构造函数	211
9.3.4	内联方法	170	10.6.3	重写方法时的特殊情况	214
9.3.5	默认参数	171	10.6.4	派生类中的复制构造函数和赋值运算符	219
9.4	不同的数据成员类型	172	10.6.5	运行时类型工具	220
9.4.1	静态数据成员	172	10.6.6	非public继承	221
9.4.2	静态常量数据成员	173	10.6.7	虚基类	221
9.4.3	引用数据成员	174	10.7	本章小结	222
9.4.4	常量引用数据成员	175	第11章	理解灵活而奇特的C++	223
9.5	嵌套类	175	11.1	引用	223
9.6	类内的枚举类型	176	11.1.1	引用变量	224
9.7	运算符重载	177	11.1.2	引用数据成员	225
9.7.1	示例: 为SpreadsheetCell实现加法	177	11.1.3	引用参数	225
9.7.2	重载算术运算符	179	11.1.4	将引用作为返回值	226
9.7.3	重载比较运算符	181	11.1.5	右值引用	226
9.7.4	创建具有运算符重载的类型	181	11.1.6	使用引用还是指针	227
9.8	创建稳定的接口	182	11.2	关键字的疑问	229
9.9	本章小结	184	11.2.1	const关键字	229
第10章	揭秘继承技术	185	11.2.2	static关键字	232
10.1	使用继承构建类	185	11.2.3	非局部变量的初始化顺序	235
10.1.1	扩展类	186	11.2.4	非局部变量的销毁顺序	235
10.1.2	重写方法	188	11.3	类型和类型转换	235

11.3.1	类型别名	235	13.1.2	流的来源和目的地	276
11.3.2	函数指针的类型别名	236	13.1.3	流式输出	277
11.3.3	方法和数据成员的指针的类型别名	238	13.1.4	流式输入	280
11.3.4	typedef	238	13.1.5	对象的输入输出	285
11.3.5	类型转换	239	13.2	字符串流	286
11.4	作用域解析	242	13.3	文件流	287
11.5	特性	243	13.3.1	文本模式与二进制模式	287
11.5.1	[[noreturn]]特性	243	13.3.2	通过seek()和tell()在文件中转移	288
11.5.2	[[deprecated]]特性	244	13.3.3	将流链接在一起	289
11.5.3	[[fallthrough]]特性	244	13.4	双向I/O	290
11.5.4	[[nodiscard]]特性	244	13.5	本章小结	291
11.5.5	[[maybe_unused]]特性	244	第14章	错误处理	292
11.5.6	供应商专用特性	245	14.1	错误与异常	292
11.6	用户定义的字面量	245	14.1.1	异常的含义	292
11.7	头文件	246	14.1.2	C++中异常的优点	293
11.8	C的实用工具	247	14.1.3	我们的建议	294
11.8.1	变长参数列表	247	14.2	异常机制	294
11.8.2	预处理器宏	249	14.2.1	抛出和捕获异常	295
11.9	本章小结	250	14.2.2	异常类型	296
第12章	利用模板编写泛型代码	251	14.2.3	按const和引用捕获异常对象	297
12.1	模板概述	252	14.2.4	抛出并捕获多个异常	297
12.2	类模板	252	14.2.5	未捕获的异常	299
12.2.1	编写类模板	252	14.2.6	noexcept	300
12.2.2	尖括号	258	14.2.7	抛出列表(已不赞成使用/已删除)	300
12.2.3	编译器处理模板的原理	258	14.3	异常与多态性	301
12.2.4	将模板代码分布在多个文件中	258	14.3.1	标准异常体系	301
12.2.5	模板参数	260	14.3.2	在类层次结构中捕获异常	302
12.2.6	方法模板	263	14.3.3	编写自己的异常类	303
12.2.7	类模板的特例化	266	14.3.4	嵌套异常	305
12.2.8	从类模板派生	267	14.4	重新抛出异常	306
12.2.9	继承还是特例化	268	14.5	堆栈的释放与清理	307
12.2.10	模板别名	268	14.5.1	使用智能指针	308
12.3	函数模板	269	14.5.2	捕获、清理并重新抛出	309
12.3.1	函数模板的特例化	270	14.6	常见的错误处理问题	309
12.3.2	函数模板的重载	271	14.6.1	内存分配错误	309
12.3.3	类模板的友元函数模板	271	14.6.2	构造函数中的错误	311
12.3.4	对模板参数推导的更多介绍	272	14.6.3	构造函数的function-try-blocks	312
12.3.5	函数模板的返回类型	272	14.6.4	析构函数中的错误	314
12.4	可变模板	274	14.7	综合应用	315
12.5	本章小结	274	14.8	本章小结	318
第13章	C++ I/O揭秘	275	第15章	C++运算符重载	319
13.1	使用流	275	15.1	运算符重载概述	319
13.1.1	流的含义	276	15.1.1	重载运算符的原因	320

15.1.2	运算符重载的限制	320
15.1.3	运算符重载的选择	320
15.1.4	不应重载的运算符	322
15.1.5	可重载运算符小结	322
15.1.6	右值引用	324
15.1.7	关系运算符	325
15.2	重载算术运算符	325
15.2.1	重载一元负号和一元正号运算符	325
15.2.2	重载递增和递减运算符	326
15.3	重载按位运算符和二元逻辑运算符	327
15.4	重载插入运算符和提取运算符	327
15.5	重载下标运算符	328
15.5.1	通过operator[]提供只读访问	330
15.5.2	非整数数组索引	330
15.6	重载函数调用运算符	331
15.7	重载解除引用运算符	332
15.7.1	实现operator*	333
15.7.2	实现operator->	333
15.7.3	operator.*和operator->*的含义	334
15.8	编写转换运算符	334
15.8.1	使用显式转换运算符解决 多义性问题	335
15.8.2	用于布尔表达式的转换	335
15.9	重载内存分配和内存释放运算符	337
15.9.1	new和delete的工作原理	337
15.9.2	重载operator new和operator delete	338
15.9.3	显式地删除/默认化operator new 和operator delete	340
15.9.4	重载带有额外参数的operator new和 operator delete	340
15.9.5	重载带有内存大小参数的 operator delete	340
15.10	本章小结	341
第 16 章	C++标准库概述	342
16.1	编码原则	343
16.1.1	使用模板	343
16.1.2	使用运算符重载	343
16.2	C++标准库概述	343
16.2.1	字符串	343
16.2.2	正则表达式	344
16.2.3	I/O流	344
16.2.4	智能指针	344
16.2.5	异常	344
16.2.6	数学工具	344
16.2.7	时间工具	345
16.2.8	随机数	345
16.2.9	初始化列表	345
16.2.10	pair和tuple	345
16.2.11	optional、variant和any	345
16.2.12	函数对象	346
16.2.13	文件系统	346
16.2.14	多线程	346
16.2.15	类型特质	346
16.2.16	标准整数类型	346
16.2.17	容器	346
16.2.18	算法	351
16.2.19	标准库中还缺什么	358
16.3	本章小结	358
第 17 章	理解容器与迭代器	359
17.1	容器概述	359
17.1.1	对元素的要求	360
17.1.2	异常和错误检查	361
17.1.3	迭代器	361
17.2	顺序容器	363
17.2.1	vector	363
17.2.2	vector<bool>特化	376
17.2.3	deque	377
17.2.4	list	377
17.2.5	forward_list	380
17.2.6	array	381
17.3	容器适配器	382
17.3.1	queue	382
17.3.2	priority_queue	384
17.3.3	stack	386
17.4	有序关联容器	387
17.4.1	pair工具类	387
17.4.2	map	388
17.4.3	multimap	393
17.4.4	set	396
17.4.5	multiset	397
17.5	无序关联容器/哈希表	397
17.5.1	哈希函数	397
17.5.2	unordered_map	399
17.5.3	unordered_multimap	401
17.5.4	unordered_set/unordered_multiset	402
17.6	其他容器	402