THE EXPERT'S VOICE® IN C++

Using the C++ Standard Template Libraries

C++标准模板库编程实战

[美] Ivor Horton 著
郭小虎 程聪 译

清华大学出版社

C++标准模板库编程实战

[美] Ivor Horton 著
郭小虎 程 聪 译

清华大学出版社
北 京

Ivor Horton
Using the C++ Standard Template Libraries
EISBN: 978-1-4842-0005-6

北京市版权局著作权合同登记号　图字：01-2016-8565

图书在版编目(CIP)数据

C++标准模板库编程实战 / (美)爱弗·霍顿(Ivor Horton) 著；郭小虎，程聪　译. — 北京：清华大学出版社，2017

书名原文：Using the C++ Standard Template Libraries
ISBN 978-7-302-45580-6

Ⅰ. ①C…　Ⅱ. ①爱…　②郭…　③程…　Ⅲ. ①C 语言－程序设计　Ⅳ. ①TP312.8

中国版本图书馆 CIP 数据核字(2016)第 281893 号

责任编辑：王　军　李维杰
封面设计：牛艳敏
版式设计：思创景点
责任校对：牛艳敏
责任印制：李红英

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
课 件 下 载：http://www.tup.com.cn，010-62794504
印 装 者：清华大学印刷厂
经　　销：全国新华书店
开　　本：185mm×260mm　　印　　张：30　　字　　数：711 千字
版　　次：2017 年 1 月第 1 版　　印　　次：2017 年 1 月第 1 次印刷
印　　数：1～3000
定　　价：69.80 元

产品编号：069421-01

译者序

C++语言是C语言的继承者，它的经典性和地位是其他高级语言不可比拟的。虽然现在随着互联网时代的到来，计算机技术的发展日新月异，各种新语言层出不穷，令人眼花缭乱，但是强大、灵活且高效的C++语言一直在TIBOE榜单位居前列，这说明了它强大的生命力。C++语言可以进行底层驱动的开发，可以开发操作系统、传统桌面程序、大型游戏(3A大作基本都是在C++开发的游戏引擎的基础上开发的)，等等。甚至在新兴的移动开发领域也有其一席之地，它的身影遍及各个领域。C++语言是一门优秀的面向对象入门语言，对于初学者，在学习完C语言之后，就可以通过学习C++来学习面向对象思想。现在主流的高级语言基本都是面向对象语言，在掌握C++之后，学习其他高级语言，譬如C#、Java、Python和Go，会有事半功倍的效果，笔者结合自己的学习经历，深有感触。

学习C++就跳不过C++的STL，STL可以说是C++的精华所在。STL提供的算法和容器等工具大大方便了我们的日常开发，我们不用再重复造轮子，减少了出错的可能性，减少了代码量，增强了代码的复用性。STL也是C++的一大难点，市场上关于STL的书籍浩如烟海，但罕有像本书这样从浅入深、循序渐进介绍STL的书，这一点本书的作者功不可没。本书作者Ivor Horton是世界著名的计算机图书大师，他在IBM公司工作多年，有着丰富的实践经验，其著作帮助很多程序员走进编程殿堂。本书是他的新作，无疑又是一本学习C++语言的经典之作。

学习语言是极其枯燥的，但是学成之后，用它写出一个程序的感觉却是令人无比喜悦，很有成就感。就笔者个人的经历而言，对语言的学习，注重实践，所以对于每章后面的练习最好都先自己独立完成一遍，这样能够帮助你回顾在这一章学到的知识，中间遇到问题，也能引发思考，从而学到知识。

就本书的结构而言，是按照由浅入深、从易到难的顺序讲解，所以建议顺序阅读。一开始介绍序列容器，而后是容器适配器，再是map容器，到set容器等，这种结构很合理。此外，本书还引入了很多C++新标准的知识，比如lambda表达式、右值参数、移动语义、完美转发、auto等，让我们能够学习C++最新的技术标准，在帮我们夯实基础的同时，开拓了我们的眼界。在后面的章节中，也介绍了STL提供的很多高级算法，它们可以按照概

率论中的各种分布生成随机数，能处理复数以及与时间相关的一些操作。当然，STL 提供的库不止于此，但这已经能够展现 STL 强大丰富的一面。

最后，由衷地感激清华大学出版社的编辑们，本书的顺利出版凝聚了他们很多的心血和汗水。

对于这本书，译者本人在翻译的过程中力求忠于原文，尽量让译文再现原书风貌，尽量简洁明了到所翻译图书的原意。但由于译者本人水平有限，难免会出现错误和失误，如果发现本书的任何错误或有任何意见和建议，请不吝指正。本书全部章节由郭小虎、程聪翻译，参与翻译的还有徐浩、黄飞、胡泽靖、徐汉，在此一并表示感谢！

最后，希望读者能够通过本书学会 STL，领略 STL 的优雅与内涵。

译　者

作者简介

Ivor Horton 毕业于数学系，却因向往 IT 工作轻松而收入丰厚，因而涉足 IT 领域。尽管现实情况常常是工作辛苦而收入却相对不高，但他仍坚持从事计算机工作至今。在不同的时期，他从事过的工作包括程序设计、系统设计、顾问以及管理和实现相当复杂的项目。

Ivor 在计算机系统的设计和实现方面，拥有多年的工作经验，这些系统应用于多种行业的工程设计和制造运营。他不仅能运用多种编程语言开发特殊用途的应用程序，而且还为科研人员和工程人员提供教学，以帮助他们完成这类工作，在这些方面他都拥有相当丰富的经验。多年来他一直从事程序设计方面书籍的撰写工作，目前出版的著作有 C、C++和 Java 等方面的教程。目前，他不忙于写书或给他人提供咨询服务时，会钓鱼、旅游和尽情地享受生活。

技术编辑简介

Marc Gregoire 是来自于比利时的一位软件工程师，他以“在计算机科学中的土木工程师”(相当于在工程中的计算机科学理学硕士)学位毕业于比利时的天主教鲁汶大学。次年，他在母校获得了优等学位的人工智能专业硕士。在学习生涯结束后，Marc 为 Ordina Belgium 软件咨询公司工作。作为一名顾问，他服务于西门子和诺基亚西门子网络关键的 2G 和 3G 软件，这些软件运行在电信运营商的 Solaris 系统上。这要求工作在从南美和美国，跨越到欧洲、中东、非洲和亚洲的国际团队中。现在，Marc 为 Nikon Metrology 研发 3D 激光扫描软件。

Marc 主要擅长 C/C++，尤其是微软的 VC++和 MFC 框架。除了 C/C++，Marc 也喜欢 C#，用 PHP 生成网页程序。他的主要兴趣在于 Windows 开发，此外，他还有在 Linux 平台(比如 EIB 的家用自动化软件)上开发全天候运行的 C++程序的经验。

从 2007 年 4 月开始，他每年都因为 Visual C++专长被授予微软 MVP 称号。

Marc 是比利时 C++用户组的创始人，同时也是 CodeGuru 论坛(类似于 Marc G)的活跃成员。他创建了很多免费和共享软件，这些软件通过他的 www.nuonsoft.com 网站发布，并且他在 www.nuonsoft.com/blog/上维护了一个博客。

致　谢

感谢 Mark Powers 和 Apress 出版社的其他编辑以及产品团队，感谢他们自始至终提供的帮助和支持。特别感谢马克所做的出色的技术审查工作，他的很多评论和建议毫无疑问使这本书变得更加优秀。

前　言

欢迎学习《C++标准模板库编程实战》一书。本教程介绍了由 C++标准库组成的头文件子集中所包含的一些类和函数的模板。这些模板是功能强大、易于使用的泛型编程工具，并使很多不容易实现的任务变得易于实现。它们生成的代码通常比我们自己编写的更加高效和可靠。

通常，笔者不喜欢只解释它们做了些什么，而不详细论述这么做的原因。从前者是很难猜出后者的。因此笔者的目标不仅仅是解释类和函数模板的功能，还会尽可能地展示如何在实际场景中应用它们。这会导致在某些知识点的介绍中包含相当大的代码块，但相信你会觉得它们是值得的。

之前提到的作为本书主题的来自于 C++标准库的头文件的集合，被称作 C++标准库或 STL。在本书中，会用 STL 作为一种方便的缩写来表示包含本书所讨论模板的头文件的集合。当然，并没有 STL 这种东西——C++语言标准并没有提到它，因此正规而言，它并不存在。尽管它并没有被定义，但很多 C++程序员都大致知道 STL 是什么意思。这种叫法由来已久。

贯穿 STL 的泛型编程思想早在 1979 年起源于 Alexander Stepanov——很久之前并没有 C++语言标准。C++的 STL 的第一个实现起源于 Stepanov 和其他在 1989 年前后工作于惠普公司的职员，而且在那时，STL 的实现和 C++编译器所提供的库是互补的。在 20 世纪 90 年代，STL 提供的功能开始被考虑纳入第一个 C++语言标准的提议中，而且 STL 的精髓使它成为公布于 1998 年的第一个 C++语言标准。从那时起，STL 所代表的泛型编程开始被改进和扩展，并且很多不属于 STL 的头文件中开始出现了模板。本书中的所有材料都和编写本书时最新通过的语言标准相关，也就是 C++14。

STL 不是一个准确的概念，并且本书中并没有包含 C++标准库的全部模板。本书只描述和展示了笔者认为 C++程序员应该首先选择理解的标准库中的模板，尤其是那些初次接触 C++的开发者。书中将被深度讨论的主要标准库头文件包括：

用于数据容器：<array>、<vector>、<deque>、<stack>、<queue>、<list>、<forward_list>、<set>、<unordered_set>、<map>、<unordered_map>

用于迭代器：<iterator>
用于算法：<algorithm>
用于随机数和统计：<random>
用于数值处理：<valarray>、<numeric>
用于时间和定时：<ratio>、<chrono>
用于复数：<complex>

来自于其他头文件的模板，比如<pair>、<tuple>、<functional>和<memory>也被加入到本书的不同章节中。数据容器的模板是基础，在很多程序中都会用到它们。迭代器是使用容器时的基本工具，因此它们也被包含了进来。算法是操作保存在容器中的数据的函数模板，也可以将这些强大的工具应用到数组上，在示例中会对此进行描述和展示。书中有一章将解释随机数生成和统计相关的模板，但是它们中有一些是相当专业的。在模拟、建模和游戏程序中，很多都得到了广泛应用。本书还讨论了计算扩展数值数据的模板，以及和时间、定时相关的模板。最后，简短介绍了一个关于用于处理复数的类模板。

使用本书的先决条件

为了理解本书的内容，需要具备一些C++语言的基本知识。本书是对《C++入门经典(第4版)》一书的补充，所以如果成功读完了那本书，就可以开始阅读这本了。需要知道的基本知识包括：类和函数模板是什么，它们工作的本质是怎样的。笔者在第1章包含了这些基本知识的概述，如果之前不习惯使用模板，它们的语法会让人觉得它们比它们本身要复杂得多。一旦开始习惯这些记号，就会发现它们的用法相对容易。STL 中也频繁使用了lambda 表达式，所以也必须习惯使用它。

我们需要一个兼容C++14 的编译器。当然，为了编写程序代码，也需要一个合适的文本编辑器。在最近的几年中，C++编译器发展得相当好，尽管它是最近才通过的标准，但已经有几个很不错的编译器在很大程度上遵从了C++14。至少可以选择3 个可用的免费编译器：

- GCC 是支持C++、C、Fortan 和其他语言的GNU 编译器集合，它支持在本书中用到的所有C++14 特性，可以从gcc.gnu.org 下载GCC。GCC 编译器集合适用于GNU 和Linux，但也可以从www.mingw.org 下载Microsoft Windows 版。
- ideaone 在线编译器支持C++14，可以通过ideaone.com 访问。在编写代码时，对于C++14，它使用的是GCC 5.1。ideaone.com 也支持很多其他语言，包括C、Fortran 和Java。
- Microsoft Visual Studio 2015 Community Edition 在Microsoft Windows 操作系统下运行，它支持C++，也支持几种其他的语言，并配置了一个完整的开发环境。

如何使用本书

对于大部分内容，为了可以顺序阅读，已经对本书的材料进行了组织，所以使用本书的最佳方式是从头阅读到尾。一般情况下，在没解释功能之前，不会使用它们。一旦解释了，无论在什么时候，只要有意义，都会把它插到后续的材料中，这也是为什么推荐按顺序阅读章节的原因。很少有需要理解背后的数学知识的主题，并且在这些实例中，也会介绍数学知识。如果不熟悉数学，可以跳过这些，因为这不会限制对后面内容的理解。

没有人可以只通过看书就学会编程。只有通过写代码才能学会如何使用 STL。强烈建议自己敲一遍所有的代码——而不仅仅只是从下载的文件中复制代码——并编译和执行你所敲入的代码。这有时可能变得很乏味，但令人惊讶的是，只需要敲一些程序语句就可以帮助我们理解代码，尤其是在我们有些琢磨不定时。它也能帮助我们记住东西。如果例子无法运行，请抵制直接回到书中查找原因的诱惑，尝试从自己的代码中查找错误。

在本书的所有章节中，对于很多部分来说，如果包含了适当的头文件，代码段都是可以执行的。如果把它们放到 main()函数中，一般都可以执行它们，并得到输出结果。因此建议为此创建一个程序项目。可以将代码复制到定义为空的 main()中，并为需要的头文件加上#include 指令。为了防止名称冲突，大多数时候需要删除之前的代码。

要让自己的错误成为学习过程的一部分，并且书中的练习为我们提供了这样的机会。我们犯的错误越多，找到和解决的问题就越多，就越能更好地了解使用模板时会犯哪些错误。确保自己可以完成所有能完成的练习，并且不去查看答案，除非确信自己不能独立地解决。许多练习都只涉及所涵盖章节的直接应用——换句话说，它们只是练习——但有些还需要有点想法，甚至需要一些灵感。

希望每个人都能学会 STL。总之，尽情享受吧！

要获取本书的源代码，读者可以访问网址 http://www.apress.com 或 http://www.tupwk.com.cn/downpage/，搜索本书的英文或中文书名，找到相应的链接下载即可，读者也可用手机扫描本书封底的二维码，直接下载。

—Ivor Horton

目　录

第 1 章

STL 介绍

本章将会说明标准模板库(Standard Template Library，STL)背后的基本思想，让你对STL 的各种对象如何相互支持有一个全局的把握。你将看到关于本章内容的更深入示例和讨论。本章将介绍以下内容：

- 什么是 STL
- 如何定义和使用模板
- 什么是容器
- 什么是迭代器以及如何使用它们
- 智能指针的重要性以及它们在容器中的用法
- 什么是算法，如何运用它们
- 数值库提供了哪些支持
- 什么是函数对象
- 如何定义和使用 lambda 表达式

除了介绍 STL 背后的一些基本思想，本章将会对一些你需要熟悉的 C++语言特性做一个简短的回顾，因为在后续章节中会频繁地使用它们。如果已经熟悉某些章节的主题，可以略过它们。

1.1 基本思想

STL 是一个功能强大且可扩展的工具集，用来组织和处理数据。为了最大限度地满足各种类型数据的需求，这个工具集全部由模板定义。虽然断定你肯定比较熟悉如何定义和使用类模板、函数模板，但本书还是会在下一节为你介绍一些它们的要领。STL 可以被划分为 4 个概念库：

- **容器库(Containers Library)**定义了管理和存储数据的容器。这个库的模板被定义在以下几个头文件中：array、vector、stack、queue、deque、list、forward_list、set、unordered_set、map 以及 unorderd_map。
- **迭代器库(Iterators Library)**定义了迭代器，迭代器是类似于指针的对象，通常被用于引用容器类的对象序列。这个库被定义在单个的头文件 iterator 中。
- **算法库(Algorithms Library)**定义了一些使用比较广泛的通用算法，可以运用到容器中的一组元素上。这个库的模板被定义在 algorithm 头文件中。
- **数值库(Numerics Library)**定义了一些通用的数学函数和一些对容器元素进行数值处理的操作。这个库也包含一些用于随机数生成的高级函数。这个库的模板被定义在 complex、cmath、valarray、numeric、random、ratio 以及 cfenv 这些头文件中。其中 cmath 已经有一些年头了，由于它包含了很多的数学函数，因此在 C++ 11 中得到了扩展。

只需要几行简洁明了的 STL 代码，很多复杂困难的任务都可以轻松地完成。例如，以下代码可以从标准输入流中读取任意数目的浮点数，然后计算并输出它们的平均值：

```
std::vector<double> values;
std::cout << "Enter values separated by one or more spaces. Enter Ctrl+Z
   to end:\n ";
values.insert(std::begin(values), std::istream_iterator<double>(std::cin),
   std::istream_iterator<double>());
std::cout << "The average is "
          << (std::accumulate(std::begin(values), std::end(values),
             0.0)/values.size())
          << std::endl;
```

只有 4 行代码，虽然每行都很长，但是我们没有使用循环去输入，因为这些工作 STL 已帮我们完成。可以很容易地将这段代码修改为从文件中获取数据，然后实现上面同样的任务。STL 强大、广泛的适用性，使它已经成为任何 C++程序员必备的工具箱。所有的 STL 名称都被定义在 std 命名空间中，所以不会在代码中显式地使用 std 来限定 STL 名称。当然，本书会在任何必要的地方使用 std 来限定。

1.2 模板

模板是一组函数或类的参数实现。编译器能够在需要使用函数或类模板时，用模板生