

PEARSON

C和C++实务精选

品味岁月积淀，读享技术菁华

C++ Templates

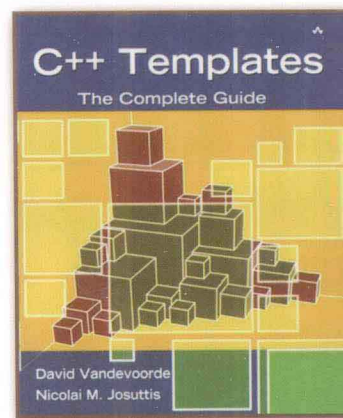
中文版

[美] David Vandevoorde [德] Nicolai M. Josuttis 著

陈伟柱 译

C++ Templates: The Complete Guide

- 详细讲解C++模板的概念与使用技巧
- 透彻剖析C++模板特性的强大功能
- *C/C++ Users Journal* 前任总编 Marc Briand 倾力推荐



人民邮电出版社
POSTS & TELECOM PRESS

PEARSON

C++ Templates

中文版

[美] David Vandevoorde [德] Nicolai M. Josuttis 著
陈伟柱 译



人民邮电出版社
北京

图书在版编目 (C I P) 数据

C++ Templates中文版 / (美) 范德沃德
(Vandevor, D.), (德) 约祖蒂斯 (Josuttis, N.M.)
著; 陈伟柱译. -- 北京: 人民邮电出版社, 2013.4
ISBN 978-7-115-31281-5

I. ①C… II. ①范… ②约… ③陈… III. ①
C语言—程序设计 IV. ①TP312

中国版本图书馆CIP数据核字(2013)第045000号

版权声明

David Vandevor, Nicolai M. Josuttis: C++ Templates: The Complete Guide

Copyright © 2003 Pearson Education, Inc.

ISBN: 0201734842

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise without the prior consent of Addison Wesley.

版权所有。未经出版者书面许可, 对本书任何部分不得以任何方式或任何手段复制和传播。
本书中文简体版由人民邮电出版社经 Pearson Education, Inc.授权出版。版权所有, 侵权必究。
本书封面贴有 Pearson Education (培生教育出版集团) 激光防伪标签。无标签者不得销售。

C++ Templates 中文版

-
- ◆ 著 [美]David Vandevor [德]Nicolai M. Josuttis
译 陈伟柱
责任编辑 傅道坤
 - ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子邮件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京艺辉印刷有限公司印刷
 - ◆ 开本: 800×1000 1/16
印张: 32.25
字数: 716千字 2013年4月第1版
印数: 1-3000册 2013年4月北京第1次印刷
著作权合同登记号 图字: 01-2013-1020号
ISBN 978-7-115-31281-5
-

定价: 89.00 元

读者服务热线: (010)67132692 印装质量热线: (010)67129223

反盗版热线: (010)67171154

广告经营许可证: 京崇工商广字第0021号

译者序

C++真可谓是包罗万象、博大精深。每个在 C++中沉迷多年的爱好者都难免有这样的感慨：使用 C++多年过后，我们往往只能算是一个熟练的使用者，却从来不敢给自己冠上“精通 C++”的头衔。难道“精通 C++”永远都是不断的大言？然而，在学习、使用和研究 C++的过程中，我们总是期望能够向“精通”不断迈进，并领悟 C++语言的精髓。我想，要做到这一点起码要注意三个方面：一要把握语言发展的脉搏；二要多应用标准技术；三要洞悉标准技术背后的实现细节。做到这些往往能够事半功倍。

近年来，C++的新发展主要是在 GP（泛型程序设计）方面大放异彩：标准库、boost 库、容器、迭代子、仿函数等都是围绕着 GP 不断呈现出来的，它们代表了现今 C++程序设计的特性。而在这种技术的背后，隐含着一种根深蒂固的共性：模板技术，处处都是模板代码。我们可以说：泛型程序设计本身就是基于模板的程序设计。也正是模板的这种编译期机制，进一步地展现了 GP 的优越，体现 C++高效率的特点，更有助于 GP 达到与 OO 并驾齐驱的地位。

使用了多年标准库等技术之后，每个人都曾经编写过许许多多模板代码，但在每天的重复劳动之余，很多人却未能真正洞悉隐藏在模板背后的实现细节。诸如特化、局部特化、实例化、重载解析等编译器实现机理，相信真正了解的人并不多。这使得我们始终未能真正摆脱我们所使用的特性的束缚，也就无法实现更加符合具体应用的技术与特性。在这种情况下，用起这些特性来总会觉得心里不踏实。这未免是程序员的一种悲哀。

从前面列出的 3 个方面来看，本书都能够解决读者的疑惑。本书前半部分内容为读者释疑解惑，后半部分内容则更加贴近开发者，使所探讨的技术真正发挥其效能。因而，也总能带给人豁然开朗的感觉，并使你深深体会到作者选材的独到之处。关于本书内容的全面介绍，请参考第 1 章，我在此就不再赘述了。

C++编程的书籍，现如今已是琳琅满目、硕果累累。但是对于 C++和模板这个至关重要的领域，即使在未来很长一段时间里，本书也必定有着不可替代的地位，这一点从亚马逊的 5 星级公评和一直位于前列的销售排名可见一斑。

对于本书的翻译，我力求做到语言平实无华，期望能以流畅的语句带给读者一个轻松的阅读过程。在近一年的翻译过程中，我一次又一次地拖延了出版社的计划，正是为了真正尽到一个译者的职责，对技术和文字把好关。但“丑媳妇总要见公婆”，这本书也终究还是要和读者见面，所以我的修润也只能告一段落。在阅读的过程中，如果你有中肯的批评意见，我

本书是 C++ 模板编程的完全指南，旨在通过基本概念、常用技巧和应用实例三方面的有用资料，为读者打下 C++ 模板知识的坚实基础。

全书共 22 章。第 1 章全面介绍了本书的内容结构和相关情况。第 1 部分（第 2～7 章）以教程的风格介绍了模板的基本概念，第 2 部分（第 8～13 章）阐述了模板的语言细节，第 3 部分（第 14～18 章）介绍了 C++ 模板所支持的基本设计技术，第 4 部分（第 19～22 章）深入探讨了各种使用模板的普通应用程序。附录 A 和附录 B 分别为一处定义原则和重载解析的相关资料。

本书适合 C++ 模板技术的初学者阅读，也可供有一定编程经验的 C++ 程序员参考。

一定虚心地接受。我也希望能够就此书的内容与读者有更多的交流。

致谢

首先，我要感谢人民邮电出版社的编辑。对我每次提交的电子稿件，他们都仔细研读，并与我细细讨论书中的每个细节。与他们合作是一次令人愉快的经历。

我要感谢我的恩师北京理工大学的陈英老师，感谢陈老师的宽容与培养。袁卫东是本书前半部分的第 1 位读者，他花了很多时间，为我指出了许多不足之处。王曦（虫虫）是第 16 章的初稿译者，他的译文准确生动，给我带来很多宝贵的启发。另外，孟岩和熊节两位好友对 C++ 有着多年的学习经历和丰富的知识背景，他们在我不断学习探索的过程中，给予我极大的帮助。

再一次，我要感谢在深圳的许多好友的支持。最后，感谢我的亲人和我的女友：在我工作的时候，每次都是你们在我身边；在我收获的时候，我最先想到的人总是你。

陈伟柱

2003 年 12 月

序

在 C++ 中，模板 (Template) 这个概念已经存在十几年了，1990 年出版的 *Annotated C++ Reference Manual* (即 “ARM”，见 [EllisStroustrupARM]) 就已经介绍了模板的一些内容。实际上，在这之前的许多专业文档也已经对模板进行了一些描述。然而，即使过了十几年之后，对于模板这一吸引人的、复杂的、强有力的 C++ 特性，仍然没有一本著作能够集中阐述它的基础概念和高级技术。我们觉得有必要阐述这些令人费解的地方，于是就决定编写这本关于模板的书籍（这些说法或许会显得有点不够谦虚）。

然而，我们两人有着不同的背景；对于这项任务，也有着不同的目的。David 是一个很有经验的编译器实现者，同时也是 C++ 标准委员会核心语言工作组的成员。他的目的在于详细而且准确地描述模板的功能（和问题）。Nico 是一个“普通”的（应用程序）程序员，同时也是 C++ 标准委员会程序库工作小组的成员。他的目的在于让读者理解他所使用的各种模板技术和使用过程中的收获。另外，我们期望可以与你（读者）和整个 (C++) 社团共享这些知识，让我们都避免那些对模板的误解、疑惑和忧虑。

于是，你在书中既会看到带有例子的概念性介绍，也会看到模板具体行为的详细描述。我们将从模板的基本概念开始介绍，逐步过渡到“模板程序设计的艺术”，其中你将会发现（或者再次发现）诸如静态多态、policy 类、metaprogramming、表达式模板等技术。另外，基于标准库中几乎到处都涉及到模板，在此你还可以加深对标准库的理解。

在本书的编写过程中，我们学会了很多知识，也获得了不少的乐趣。我们希望你阅读的过程中也能有这样的感受，享受这本书和这份乐趣！

致 谢

这本书引用了许多别人的思想、概念、解决方案和例子，在此我们感谢在过去几年里所有帮助和支持我们的个人和公司。

首先，我们感谢所有的审阅者和对我们的早期草稿提意见的人，这本书的质量很大程度上要归功于他们（她们）的付出。本书的审阅者有：Kyle Blaney、Thomas Gschwind、Dennis Mancl、Patrick Mc Killen 和 Jan Christiaan Van Winkel；特别要感谢 Dietmar Kühl，他细致入微地审阅和校对了整本书，他的反馈大大提高了这本书的质量。

我们还要感谢所有帮助我们在不同的编译器平台测试书中例子程序的个人和公司。特别要感谢 Edison Design Group 公司，他们为我们提供了一个很优秀的编译器，还给予了我们大力的支持；这对本书的编写和 C++ 的标准化过程都带来了很大的帮助。另外，我们还要感谢免费的 GNU 和 egcs 编译器的开发者（Jason Merrill 是特别要感谢的人）和 Microsoft，他们为我们提供了一份评估版本的 Visual C++（Jonathan Caves、Herb Sutter、Jason Shirk 是我们在那边的朋友）。

总的说来，现存的许多“C++ Wisdom”得益于在线 C++ 团体。其中的大多数内容都来自新闻组 `comp.lang.c++.moderated` 和 `comp.std.c++`；因此我们要特别感谢这些新闻组活跃的管理者，正是他们的努力才使所讨论的内容更加有用，更具建设性。我们还要感谢那些多年来不遗余力地为我们解释并和我们分享他们的想法的人。

Addison-Wesley 出版公司的团队做了一份很出色的工作。我们特别要感谢 Debbie Lafferty（我们的编辑），感谢他那温和的督促、良好的建议、不倦的工作和对这本书的支持。另外还

要感谢 Tyrrell Albaugh、Bunny Ames、Melanie Buck、Jacquelyn Doucette、Chanda Leary-Coutu、Catherine Ohala 和 Marty Rabinowitz。我们还要衷心感谢 Marina Lang，正是他首先在 Addison-Wesley 内部提出这个选题的。最后，Susan Winer 的早期编辑工作，也大大有助于我们后面的工作。

Nico 的致谢

我首先要感谢我的家人：Ulli、Lucas、Anica 和 Frederic，感谢他们对我 and 这本书的耐心、关怀和鼓励。

另外，我还要感谢 David，他是一个非常优秀的专家，而且他的耐心显得尤为难得（有时，我甚至会问一些比较幼稚的问题）。和他一起工作让我感到极大的乐趣。

David 的致谢

首先要感谢我的妻子 Karina，这本书能够完成要归功于她的帮助和在生活中给我带来的一切。当写书时间和其他活动安排发生冲突的时候，“利用空闲时间”写书显然是不现实的。Karina 帮我安排了整个时间计划，为了挤出更多的时间来写作，她教我如何对一些活动说“不”；所有的这一切都对这本书的完成给予了极大的帮助。

能够和 Nico 一起工作让我感到非常荣幸。除了承担部分书稿的写作，另外，正是由于 Nico 的经验和专业精神，才令这本原先显得凌乱的草稿变成一本结构合理的书籍。

John “Mr. Template” Spicer 和 Steve “Mr. Overload” Adamczyk 是很好的朋友和同事；在我看来，他们都是核心 C++ 语言的权威。他们澄清了书中一些令人疑惑的问题；如果你在书中找到关于 C++ 语言的特性（element）的一些错误，那么是我的疏忽，怪我没有向他们请教。

最后，我要对下面这些支持我这份工作的许多人表达我的谢意；尽管他们的支持是间接的，但他们给我带来的一切同样是不可低估的。首先，我要感谢我的父母，正是他们的关爱和鼓励，才使我的一切发生了改变。下面是一些给我关怀（譬如询问“书进行得怎么样了？”）的朋友，他们的鼓励同样给予了我很大的动力：Michael Beckmann、Brett and Julie Beene、Jarran Carr、Simon Chang、Ho and Sarah Cho、Christophe De Dinechin、Ewa Deelman、Neil Eberle、Sassan Hazeghi、Vikram Kumar、Jim 和 Lindsay Long、R.J. Morgan、Mike Puritano、Ragu Raghavendra、Jim 和 Phuong Sharp、Gregg Vaughn 和 John Wiegley。

第 1 章 关于本章	1
1.1 阅读本书所需具备的知识	2
1.2 本书的整体结构	2
1.3 如何阅读本书	2
1.4 关于编程风格的一些说明	3
1.5 标准和现实	5
1.6 代码例子和更多信息	5
1.7 反馈	5
第 1 部分 基础	7
第 2 章 函数模板	9
2.1 初探函数模板	9
2.1.1 定义模板	9
2.1.2 使用模板	10
2.2 实参的演绎 (deduction)	12
2.3 模板参数	13
2.4 重载函数模板	15
2.5 小结	19
第 3 章 类模板	21
3.1 类模板 Stack 的实现	21

3.1.1	类模板的声明	22
3.1.2	成员函数的实现	23
3.2	类模板 Stack 的使用	25
3.3	类模板的特化	27
3.4	局部特化	29
3.5	缺省模板实参	30
3.6	小结	32
第 4 章	非类型模板参数	33
4.1	非类型的类模板参数	33
4.2	非类型的函数模板参数	36
4.3	非类型模板参数的限制	37
4.4	小结	38
第 5 章	技巧性基础知识	39
5.1	关键字 <code>typename</code>	39
5.2	使用 <code>this-></code>	41
5.3	成员模板	42
5.4	模板的模板参数	45
5.5	零初始化	51
5.6	使用字符串作为函数模板的实参	52
5.7	小结	55
第 6 章	模板实战	57
6.1	包含模型	57
6.1.1	链接器错误	57
6.1.2	头文件中的模板	59
6.2	显式实例化	60
6.2.1	显式实例化的例子	61
6.2.2	整合包含模型和显式实例化	62
6.3	分离模型	63
6.3.1	关键字 <code>export</code>	63
6.3.2	分离模型的限制	65
6.3.3	为分离模型做好准备	66
6.4	模板和内联	67
6.5	预编译头文件	68

6.6	调试模板	70
6.6.1	理解长段的错误信息	71
6.6.2	浅式实例化	72
6.6.3	长符号串	75
6.6.4	跟踪程序	75
6.6.5	oracles	79
6.6.6	archetypes	80
6.7	本章后记	80
6.8	小结	81
第 7 章	模板术语	83
7.1	“类模板”还是“模板类”	83
7.2	实例化和特化	84
7.3	声明和定义	85
7.4	一处定义原则	86
7.5	模板实参和模板参数	86
第 2 部分	深入模板	89
第 8 章	深入模板基础	91
8.1	参数化声明	91
8.1.1	虚成员函数	94
8.1.2	模板的链接	95
8.1.3	基本模板	96
8.2	模板参数	96
8.2.1	类型参数	97
8.2.2	非类型参数	97
8.2.3	模板的模板参数	98
8.2.4	缺省模板实参	99
8.3	模板实参	100
8.3.1	函数模板实参	101
8.3.2	类型实参	103
8.3.3	非类型实参	105
8.3.4	模板的模板实参	107
8.3.5	实参的等价性	109
8.4	友元	109

8.4.1	友元函数	110
8.4.2	友元模板	113
8.5	本章后记	113
第 9 章	模板中的名称	115
9.1	名称的分类	115
9.2	名称查找	117
9.2.1	Argument-Dependent Lookup (ADL)	119
9.2.2	友元名称插入	121
9.2.3	插入式类名称	121
9.3	解析模板	123
9.3.1	非模板中的上下文相关性	123
9.3.2	依赖型类型名称	125
9.3.3	依赖型模板名称	127
9.3.4	using-declaration 中的依赖型名称	129
9.3.5	ADL 和显式模板实参	130
9.4	派生和类模板	131
9.4.1	非依赖型基类	131
9.4.2	依赖型基类	132
9.5	本章后记	134
第 10 章	实例化	137
10.1	On-Demand 实例化	137
10.2	延迟实例化	139
10.3	C++的实例化模型	142
10.3.1	两阶段查找	142
10.3.2	POI	142
10.3.3	包含模型与分离模型	145
10.3.4	跨翻译单元查找	146
10.3.5	例子	147
10.4	几种实现方案	149
10.4.1	贪婪实例化	151
10.4.2	询问实例化	152
10.4.3	迭代实例化	153
10.5	显式实例化	155
10.6	本章后记	159

第 11 章 模板实参演绎	163
11.1 演绎的过程	163
11.2 演绎的上下文	165
11.3 特殊的演绎情况	167
11.4 可接受的实参转型	168
11.5 类模板参数	169
11.6 缺省调用实参	169
11.7 Barton-Nackman 方法	170
11.8 本章后记	172
第 12 章 特化与重载	175
12.1 当泛型代码不再适用时	175
12.1.1 透明自定义	176
12.1.2 语义的透明性	177
12.2 重载函数模板	178
12.2.1 签名	179
12.2.2 重载的函数模板的局部排序	182
12.2.3 正式的排序原则	183
12.2.4 模板和非模板	185
12.3 显式特化	185
12.3.1 全局的类模板特化	186
12.3.2 全局的函数模板特化	189
12.3.3 全局成员特化	191
12.4 局部的类模板特化	194
12.5 本章后记	197
第 13 章 未来的方向	199
13.1 尖括号 Hack	199
13.2 放松 typename 的原则	200
13.3 缺省函数模板实参	201
13.4 字符串文字和浮点型模板实参	202
13.5 放松模板的模板参数的匹配	204
13.6 typedef 模板	206
13.7 函数模板的局部特化	207
13.8 typeof 运算符	208
13.9 命名模板实参	210

13.10	静态属性	211
13.11	客户端的实例化诊断信息	212
13.12	重载类模板	214
13.13	List 参数	215
13.14	布局控制	217
13.15	初始化器的演绎	218
13.16	函数表达式	219
13.17	本章后记	221
第 3 部分	模板与设计	223
第 14 章	模板的多态威力	225
14.1	动多态	225
14.2	静多态	228
14.3	动多态和静多态	231
14.3.1	术语	231
14.3.2	优点和缺点	232
14.3.3	组合这两种多态	232
14.4	新形式的设计模板	233
14.5	泛型程序设计	234
14.6	本章后记	236
第 15 章	trait 与 policy 类	239
15.1	一个实例：累加一个序列	239
15.1.1	fixed traits	240
15.1.2	value trait	243
15.1.3	参数化 trait	247
15.1.4	policy 和 policy 类	249
15.1.5	trait 和 policy：区别在何处	251
15.1.6	成员模板和模板的模板参数	252
15.1.7	组合多个 policie 和/或 trait	254
15.1.8	运用普通的迭代器进行累积	255
15.2	类型函数	256
15.2.1	确定元素的类型	257
15.2.2	确定 class 类型	259
15.2.3	引用和限定符	261

15.2.4	promotion trait	264
15.3	policy trait	267
15.3.1	只读的参数类型	268
15.3.2	拷贝、交换和移动	271
15.4	本章后记	275
第 16 章	模板与继承	277
16.1	命名模板参数	277
16.2	空基类优化	281
16.2.1	布局原则	281
16.2.2	成员作基类	284
16.3	奇特的递归模板模式	286
16.4	参数化虚拟性	289
16.5	本章后记	290
第 17 章	metaprogram	293
17.1	metaprogram 的第一个实例	293
17.2	枚举值和静态常量	295
17.3	第 2 个例子：计算平方根	297
17.4	使用归纳变量	301
17.5	计算完整性	304
17.6	递归实例化和递归模板实参	304
17.7	使用 metaprogram 来展开循环	306
17.8	本章后记	309
第 18 章	表示式模板	313
18.1	临时变量和分割循环	314
18.2	在模板实参中编码表达式	319
18.2.1	表达式模板的操作数	320
18.2.2	Array 类型	323
18.2.3	运算符	325
18.2.4	回顾	327
18.2.5	表达式模板赋值	329
18.3	表达式模板的性能与约束	330
18.4	本章后记	331

第 4 部分 高级应用程序	335
第 19 章 类型区分	337
19.1 辨别基本类型	337
19.2 辨别组合类型	340
19.3 辨别函数类型	342
19.4 运用重载解析辨别枚举类型	346
19.5 辨别 class 类型	348
19.6 辨别所有类型的函数模板	349
19.7 本章后记	352
第 20 章 智能指针	355
20.1 holder 和 trule	355
20.1.1 安全处理异常	356
20.1.2 holder	358
20.1.3 作为成员的 holder	360
20.1.4 资源获取于初始化	362
20.1.5 holder 的局限	363
20.1.6 复制 holder	364
20.1.7 跨函数调用来复制 holder	365
20.1.8 trule	366
20.2 引用记数	368
20.2.1 计数器在什么地方	370
20.2.2 并发访问计数器	370
20.2.3 析构和释放	371
20.2.4 CountingPtr 模板	372
20.2.5 一个简单的非侵入式计数器	375
20.2.6 一个简单的侵入式计数器模板	377
20.2.7 常数性	378
20.2.8 隐式转型	379
20.2.9 比较	381
20.3 本章后记	383
第 21 章 tuple	385
21.1 duo	385
21.2 可递归 duo	390