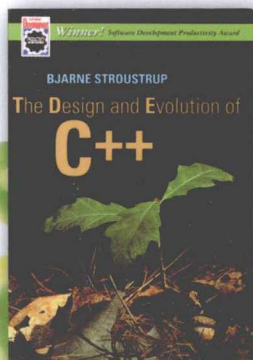


(美) **Bjarne Stroustrup** / 编著
裘宗燕 / 译

C++语言的设计与演化

The Design and Evolution of C++



荣获美国Software Development Productivity Award大奖，著译双馨



科学出版社

C++语言的设计与演化

The Design and Evolution of C++

[美] Bjarne Stroustrup 编著
裘宗燕 译

科学出版社

图字: 01-2011-7229 号

内 容 简 介

本书是C++的设计者关于C++语言的最主要著作之一。作者综合性地论述了C++的发展历史,C++中各种重要机制的本质意义和设计背景,这些机制的基本用途和使用方法,讨论了C++所适合的应用领域及其未来的发展前景。全书共18章,分两大部分,其中第1部分(第1~9章)分别介绍了C++的史前时代、C with Classes、C++的诞生、C++语言设计规则、1985—1993年表、标准化、关注和使用、库、展望;第2部分(第10~18章)分别介绍了存储管理、重载、多重继承、类概念的精练、强制转换、模板、异常处理、名称空间、C语言预处理器。为便于查找与学习,全书最后附有专业术语索引。本书在帮助人们理解C++语言方面的作用是不可替代的。

本书面向每个学习和关注C++语言的专业工作者、科研人员、教师及学生。

著作权声明

Authorized translation from the English language edition, entitled the Design and Evolution of C++, 1E, 978-0-201-54330-8 by Bjarne Stroustrup, published by Pearson Education, Inc, publishing as Addison-Wesley Professional, Copyright © 2004 by 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 China Science Publishing and Media Ltd.(Science Press). Copyright © 2010.

本书中文简体字版由培生教育出版公司授权中国科技出版传媒股份有限公司(科学出版社)合作出版,未经出版者书面许可,不得以任何形式复制或抄袭本书的任何部分。

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。

图书在版编目(CIP)数据

C++语言的设计与演化/(美)斯特劳斯特鲁普
(Stroustrup, B.) 编著; 裘宗燕译. —北京: 科学
出版社, 2012. 2
ISBN 978-7-03-033346-9

I. ①C… II. ①斯… ②裘… III. ①C 语
言—程序设计 IV. ①TP312

中国版本图书馆CIP数据核字(2012)第005949号

责任编辑: 郑楠 何武 吴俊华 / 责任校对: 杨慧芳
责任印刷: 新世纪书局 / 封面设计: 张世杰

科学出版社 出版

北京东黄城根北街16号

邮政编码: 100717

<http://www.sciencep.com>

中国科技出版传媒集团新世纪书局策划

三河市李旗庄少明装订厂印刷

中国科技出版传媒集团新世纪书局发行 各地新华书店经销

*

2012年3月 第一版 开本: 16开

2012年3月第一次印刷 印张: 26

字数: 569 000

定价: 86.00元

(如有印装质量问题, 我社负责调换)

前言

一个人，如果不耕作，
就必须写作。

——Martin A. Hansen

ACM关于程序设计语言历史的HOLP-2会议邀请我写一篇有关C++历史的文章，这看起来是一个很不错的想法，还带着点荣誉性质，于是我就开始写了。为了对C++的演化有一个更全面、更公正的陈述，我向一些朋友询问了他们对C++早期的记忆。这就使关于这个工作的小道消息不胫而走，甚至有关的故事逐渐变了味——有一天，我忽然接到一个朋友的来函，问我在哪里可以买到我关于C++设计的新书，这封电子邮件就是编写本书的真正起由。

在传统上，关于程序设计和程序设计语言的书都是在解释某种语言究竟是什么，还有就是如何去使用它。但无论如何，有许多人也很想知道某个语言为什么会具有它现在的这个样子，以及它是怎样成为这个样子的。本书就是想针对C++语言，给出对后面这两个问题的解释。在这里要解释C++怎样从它的初始设计演化为今天的这种语言，要描述造就了C++的各种关键性问题、设计目标、语言思想和各种约束条件，以及这些东西又是如何随着时间的推移而变化的。

当然，C++语言和造就它的设计思想、编程思想本身并没有演化，真正演化的是C++用户对于实际问题的理解，以及他们对于能够帮助解决这些问题的工具的理解。因此，在本书中也将追溯人们用C++去处理的各种关键性问题，以及实际处理那些问题的人们的认识，这些都对C++的发展产生了重要影响。

C++仍然是一种年轻的语言，许多用户对这里将要讨论的一些问题还不知晓。这里所描述的各种决策的进一步推论，可能还需要一些年才能变得更清晰起来。本书要展示的是我个人关于C++如何出现、它是什么，以及它应该是什么的观点。希望这些东西能帮助人们理解怎样才能更好地使用C++，理解C++正在继续进行的演化进程。

特别要强调的是书中整体的设计目标、现实的约束及造就出C++的那些人们。有关各种语言特征的关键性设计决策的讨论被放到了相应的历史环境里，在这里追溯了C++的演化过程，从C with Classes开始，经过Release 1.0和2.0，直到当前ANSI/ISO的标准化工作，讨论了使用、关注、商业行为、编译系统、工具、环境和库的爆炸性增长，还讨论了C++与C、Simula之间关系的许多细节，对C++与其他语言的关系只做了简短讨论。对主要语言功能的设计，例如类、继承、抽象类、重载、存储管理、模板、异常处理、运行时类型信息和名称空间等，都在一定细节程度上进行了讨论。

本书的根本目的，就是想帮助C++程序员更好地认识他们所用的语言、该语言的背景和基本概念，希望能激励他们去试验那些对他们而言完全是新的C++使用方式。本书也可供有经验的程序员和程序设计专业的学生阅读，可以帮助他们确定使用C++是不是一件值得做的事情。

前言

鸣谢

我非常感谢Steve Clamage、Tony Hansen、Lorraine Juhl、Peter Juhl、Brian Kernighan、Lee Knight、Doug Lea、Doug McIlroy、Barbara Moo、Jens Palsberg、Steve Rumsby和Christopher Skelly，感谢他们完整地阅读了本书的手稿，他们建设性的意见使本书的内容和组织都发生了重要变化。Steve Buroff、Martin Carroll、Sean Corfield、Tom Hagelskjaer、Rick Hollinbeck、Dennis Mancl和Stan Lippman通过对一些章节的评论提供了帮助，还要感谢Archie Lachner在我还没有想到本书之前就提出了对本书的要求。

自然，我还应该感谢那些帮助创造出C++语言的人们。从某种意义上说，本书就是献给他们的礼物，他们中一部分人的名字可以在各个章节中找到。如果要我点出一些人来，那就必然是Brian Kernighan、Andrew Koenig、Doug McIlroy和Jonathan Shapiro。他们每一个在过去十多年间都一直支持和鼓励我，都是提供各种想法的源泉。还有，要感谢Kristen Nygaard和Dennis Ritchie作为Simula和C的设计师，C++从它们那里借用了一些关键性的成分。经过这些年，我已经逐渐了解到他们不仅是才华横溢、实事求是的语言设计师，还是真正的绅士和绝对亲切的人。

Bjarne Stroustrup
Murray Hill, New Jersey

译者序

这是一本独特的书，是由C++语言的设计师本人写就，描述C++语言的发展历史、设计理念及技术细节的著作。在计算机发展的历史上，这种从多方面、多角度描述一种主流语言各个方面的综合性著作，至今我还只看到这一本。阅读本书，不仅可以了解有关C++语言的许多重要技术问题和细节，还可以进一步理解各种C++特征的设计出发点，设计过程中所考虑的方方面面问题，以及语言成形过程中的各种权衡和选择。每个学习和使用C++语言的人，一定能由此加深对自己所用工具的认识，进一步理解应该如何用好这种语言，还能看到作者对于复杂的系统程序设计的许多观点和想法。如果一个人想深入了解C++语言，想使C++成为自己得心应手的工具，想在复杂的系统程序设计领域中做出一些有价值的工作，想了解面向对象程序设计语言的一般性问题，想了解程序设计语言的发展现状、问题和前景，本书是最值得阅读的书籍之一。

C++语言的设计目标是提供一种新的系统开发工具，希望能在一些方面比当时的各种工具语言有实质性的进步。今天看，C++最重要的作用就是使那时的阳春白雪（数据抽象、面向对象的理论和技术等）变成了普通系统开发人员可以触及、可以接受并使用、可以从中获益的东西。这件事在计算科学技术发展的历史记录上必定会留下明显的痕迹。本书从一个最直接参与者的角度，记录了C++语言的起源和发展，记录了它怎样成长为今天的这种语言，怎样使语言研究的成果变成了程序员手中的现实武器。

从来都没有一种完美的程序设计语言。C++语言由于其出身（出自C语言），其发展过程中各种历史和现实因素的影响，也带着许多瑕疵和不和谐，尤其是在作为C++基础的C语言的低级成分与面向数据抽象的高级机制之间。对于一个目标是支持范围广泛的复杂系统实现的语言来说，这类问题也很难避免。为了系统的效率和资源的有效利用，人们希望有更直接的控制手段（低级机制）；而为能将复杂的功能组织成人易于理解和把握的系统，又需要有高级的机制和结构。在使用同时提供这两方面机制的语言时，应该如何合理而有效地利用它们，使其能互为补充而不是互相冲突？本书中许多地方讨论到这些问题，也提出了许多建议。这些，对于正确合理地使用C++语言都是极其重要的。

C++并不是每个人都喜欢的语言（没有任何语言可能做到这一点），但不抱偏见的人都会承认，C++语言取得了极大成功。C++语言的工作开始于一个人（本书作者）的某种很合理、很直观的简单想法（为复杂的系统程序设计提供一种更好的工具），由于一个人始终不渝的努力，一小批人的积极参与，在一大批人（遍及世界的系统开发人员）的热心关注、评论和监督下，最终造就出了一项重要的工作。这个工作过程本身就非常耐人寻味，它也是在现代信息环境（主要是互连网络）下，开展全球范围科学技术研究的一个最早的演练。在这个成功中，商业的考虑、宣传和炒作从来没有起过任何实质性的作用，起作用的仍然是理性的思维、严肃的科学态度、踏踏实实的实际工作。这些，与今天在信息科学技术领域中常见的浮躁情绪和过分的利益追求形成了鲜明对比。许多事实给了我们一种警示：时尚转眼就可能变成无人理睬的烂泥，仅仅被眼球注意的东西很快就会被忽视，炒作最凶的东西往往也消失得最快，而真正有价值的成果则往往起源于人们最基本的需求和向往。

作为C++语言的创造者，作者对于自己的作品自然是珍爱有加。对某些针对C++语言的批评，本书中也有一些针锋相对的、比较尖锐的观点。但通观全书，作者的论点和意

译者序

见还是比较客观的，并没有什么过于情绪化的东西。在前瞻性讨论中，作者提出了许多预见。经过这些年，其中一些已经变成了现实，也有些（例如特别有效的开发环境）还在发展之中。这些可能也说明了语言本身的一些性质：C++是个比较复杂的语言，做好支持它的工具绝不是一件容易的事情，在这些方面还有许多发展余地。

作者在讨论C++的设计和发展的过程中，还提出了许多人文科学领域的问题，提出了他在从事科学技术工作中的人文思考，这些认识和观点也是C++成长为今天这样一种语言的基础。当然，作者的这些想法也可以供我们参考。

今天，作为一种通用的系统程序设计语言，C++已经得到了广泛的认可。许多个人和企业将C++作为软件系统的开发工具，许多计算机专业课程用它作为工具语言。近十年来，国外的一些计算机教育工作者也一直在探索将C++作为CS1（计算机科学的第一门课程）工作语言的可行性，国内学习和使用C++的人也越来越多。在这种情况下，由C++语言设计师Bjarne Stroustrup本人撰写的有关C++的两部重要著作——本书和《C++程序设计语言》都已在中国出版，这当然是非常有意义的事情。为此我非常感谢相关出版社的管理和编辑人员（相信许多计算机工作者也会如此），感谢他们在国内出版界更多关注时髦计算机图书的浪潮中，愿意付诸努力，出版一些深刻的、影响长远的重要著作。我祝愿这种工作能获得丰厚的回报，对于整个社会，也包括出版社。

作为译者，我希望本书能给学习C++语言、用这个语言从事教学、从事程序设计工作和复杂系统程序设计的人们提供一点帮助，使这本有关C++语言的最重要著作中阐释的事实和思想能够被更多人所了解。虽然我始终将这些铭记在心，但译文中仍难免出现差错和疏漏，在此也恳请有见识的读者不吝赐教。

裘宗燕

2001年6月于北大

新版注记

本书的中文初版于2002年由华章公司出版。承蒙那里朋友的理解和支持，以及中国科技出版传媒股份有限公司科海新世纪书局的帮助，本书得以再版。再版图书保持了初版的基本文字，但也根据情况和读者意见做了全面的修改和润色，并保留了2002年中文版的译者序，特此说明。

关于译者

裘宗燕，北京大学数学学院信息科学系教授。关注的主要学术领域包括计算机软件理论、形式化方法、程序设计方法学、程序设计语言等。已出版多部著作和译著，包括《程序设计语言基础》（译）、《Mathematica数学软件系统的应用与程序设计》、《从问题到程序——程序设计与C语言引论》、《程序设计实践》（译）、《C++程序设计语言》（译）、《从规范出发的程序设计》（译）、《B方法》（译）、《C++基本程序设计》、《程序设计语言——实践之路》（译）等。

E-mail: qzy@math.pku.edu.cn

目 录

第0章 致读者	1
---------------	---

第1部分

第1章 C++的史前时代	17
1.1 Simula和分布式系统	17
1.2 C与系统程序设计	19
1.3 一般性的背景	20
第2章 C with Classes	23
2.1 C with Classes的诞生	23
2.2 特征概览	25
2.3 类	26
2.4 运行时的效率	28
2.5 连接模型	30
2.5.1 纯朴的实现	32
2.5.2 对象连接模型	33
2.6 静态类型检查	35
2.6.1 窄转换	36
2.6.2 警告的使用	36
2.7 为什么是C	37
2.8 语法问题	39
2.8.1 C声明的语法	39
2.8.2 结构标志与类型名	41
2.8.3 语法的重要性	42
2.9 派生类	42
2.9.1 没有虚函数时的多态性	43
2.9.2 没有模板时的容器类	44
2.9.3 对象的布局模型	45
2.9.4 回顾	46
2.10 保护模型	46
2.11 运行时的保证	48



2.11.1 构造函数与析构函数	49
2.11.2 存储分配和构造函数	49
2.11.3 调用函数和返回函数	50
2.12 次要特征	50
2.12.1 赋值的重载	50
2.12.2 默认实参	51
2.13 考虑过，但是没有提供的特征	52
2.14 工作环境	52

第3章 C++的诞生 55

3.1 从C with Classes到C++	55
3.2 目标	57
3.3 Cfront	58
3.3.1 生成C	59
3.3.2 分析C++	60
3.3.3 连接问题	61
3.3.4 Cfront发布	61
3.4 语言特征	63
3.5 虚函数	63
3.5.1 对象布局模型	66
3.5.2 覆盖和虚函数匹配	67
3.5.3 基成员的遮蔽	68
3.6 重载	68
3.6.1 基本重载	70
3.6.2 成员和友元	71
3.6.3 运算符函数	73
3.6.4 效率和重载	73
3.6.5 变化和新运算符	75
3.7 引用	75
3.8 常量	78
3.9 存储管理	80
3.10 类型检查	81

3.11 次要特征	82
3.11.1 注释	82
3.11.2 构造函数的记法	82
3.11.3 限定	83
3.11.4 全局变量的初始化	84
3.11.5 声明语句	87
3.12 与经典C的关系	88
3.13 语言设计工具	91
3.14 《C++程序设计语言》（第一版）	92
3.15 有关“什么是”的论文	93
第4章 C++语言设计规则	96
4.1 规则和原理	96
4.2 一般性规则	97
4.3 设计支持规则	100
4.4 语言的技术性规则	103
4.5 低级程序设计支持规则	106
4.6 最后的话	108
第5章 1985—1993年表	109
5.1 引言	109
5.2 Release 2.0	110
5.3 带标注的参考手册（ARM）	112
5.4 ANSI和ISO标准化	114
第6章 标准化	118
6.1 什么是标准	118
6.1.1 实现细节	119
6.1.2 现实的检查	120
6.2 委员会如何运作	121
6.3 净化	123
6.3.1 查找问题	123
6.3.2 临时量的生存期	127



6.4 扩充	130
6.4.1 评价准则	133
6.4.2 状况	134
6.4.3 好扩充的问题	136
6.4.4 一致性	137
6.5 扩充建议实例	137
6.5.1 关键词实参	137
6.5.2 受限指针	141
6.5.3 字符集	142
第7章 关注和使用	146
7.1 关注和使用的爆炸性增长	146
7.1.1 C++市场的缺位	147
7.1.2 会议	148
7.1.3 杂志和书籍	148
7.1.4 编译器	149
7.1.5 工具和环境	149
7.2 C++的教与学	150
7.3 用户和应用	156
7.3.1 早期用户	156
7.3.2 后来的用户	156
7.4 商业竞争	157
7.4.1 传统语言	158
7.4.2 更新一些的语言	158
7.4.3 期望和看法	159
第8章 库	162
8.1 引言	162
8.2 C++库设计	162
8.2.1 库设计的折中	163
8.2.2 语言特征和库的构造	164
8.2.3 处理库的多样性	164

8.3 早期的库	165
8.3.1 I/O流库	166
8.3.2 并行支持	168
8.4 其他库	171
8.4.1 基础库	171
8.4.2 持续性和数据库	172
8.4.3 数值库	172
8.4.4 专用库	172
8.5 一个标准库	173
第9章 展望	175
9.1 引言	175
9.2 回顾	176
9.2.1 C++在其预期领域取得了成功吗?	176
9.2.2 C++是不是一种统一的语言?	177
9.2.3 什么是最大失误?	180
9.3 仅仅是一座桥梁吗?	180
9.3.1 在一个很长的时期我们还需要这座桥梁	180
9.3.2 如果C++是答案,那么问题是什么?	181
9.4 什么能使C++更有效	184
9.4.1 稳定性和标准	184
9.4.2 教育和技术	185
9.4.3 系统方面的问题	185
9.4.4 在文件和语法之外	186
9.4.5 总结	187

第2部分

第10章 存储管理	189
10.1 引言	189
10.2 将存储分配和初始化分离	190
10.3 数组分配	191

10.4 放置	192
10.5 存储释放问题	193
10.6 存储器耗尽	195
10.7 自动废料收集	196
10.7.1 可选的废料收集	196
10.7.2 可选择的废料收集应该是什么样子的?	198
第11章 重载	200
11.1 引言	200
11.2 重载的解析	201
11.2.1 细粒度解析	201
11.2.2 歧义控制	203
11.2.3 空指针	205
11.2.4 overload关键字	207
11.3 类型安全的连接	208
11.3.1 重载和连接	208
11.3.2 C++连接的一种实现	209
11.3.3 回顾	210
11.4 对象的建立和复制	212
11.4.1 对复制的控制	212
11.4.2 对分配的控制	213
11.4.3 对派生的控制	213
11.4.4 按成员复制	214
11.5 记法约定	215
11.5.1 灵巧指针	216
11.5.2 灵巧引用	217
11.5.3 增量和减量的重载	219
11.5.4 重载 ->*	221
11.5.5 重载逗号运算符	221
11.6 给C++增加运算符	221
11.6.1 指数运算符	221
11.6.2 用户定义运算符	223

11.6.3 复合运算符	225
11.7 枚举	226
11.7.1 基于枚举的重载	227
11.7.2 布尔类型	227
第12章 多重继承	229
12.1 引言	229
12.2 普通基类	230
12.3 虚基类	231
12.4 对象布局模型	235
12.4.1 虚基布局	237
12.4.2 虚基类和强制	237
12.5 方法组合	238
12.6 有关多重继承的论战	239
12.7 委托	242
12.8 重命名	243
12.9 基类和成员初始化	245
第13章 类概念的精练	247
13.1 引言	247
13.2 抽象类	247
13.2.1 为处理错误而用的抽象类	248
13.2.2 抽象类型	249
13.2.3 语法	250
13.2.4 虚函数和构造函数	251
13.3 const成员函数	253
13.3.1 强制去掉const	254
13.3.2 const定义的精练	254
13.3.3 可变性与强制	255
13.4 静态成员函数	256
13.5 嵌套的类	257
13.6 Inherited::	258



13.7 放松覆盖规则	260
13.8 多重方法	264
13.9 保护成员	267
13.10 改进代码生成	268
13.11 指向成员的指针	269
第14章 强制转换	271
14.1 主要扩充	271
14.2 运行时类型信息	272
14.2.1 问题	273
14.2.2 dynamic_cast运算符	273
14.2.3 RTTI的使用和误用	278
14.2.4 为什么提供一个“危险特征”	280
14.2.5 typeid()运算符	281
14.2.6 对象布局模型	284
14.2.7 一个例子：简单的对象I/O	285
14.2.8 考虑过的其他选择	287
14.3 强制的一种新记法	290
14.3.1 问题	290
14.3.2 static_cast运算符	292
14.3.3 reinterpret_cast运算符	293
14.3.4 const_cast运算符	295
14.3.5 新风格强制的影响	296
第15章 模板	298
15.1 引言	298
15.2 模板	299
15.3 类模板	301
15.4 对模板参数的限制	303
15.4.1 通过派生加以限制	304
15.4.2 通过使用加以限制	305

15.5 避免代码重复	306
15.6 函数模板	307
15.6.1 函数模板参数的推断	308
15.6.2 描述函数模板的参数	309
15.6.3 函数模板的重载	311
15.7 语法	313
15.8 组合技术	315
15.8.1 表述实现策略	316
15.8.2 描述顺序关系	316
15.9 模板类之间的关系	318
15.9.1 继承关系	318
15.9.2 转换	320
15.9.3 成员模板	321
15.10 模板的实例化	322
15.10.1 显式的实例化	323
15.10.2 实例化点	325
15.10.3 专门化	329
15.10.4 查找模板定义	331
15.11 模板的作用	333
15.11.1 实现与界面的分离	334
15.11.2 灵活性和效率	334
15.11.3 对C++其他部分的影响	335
第16章 异常处理	337
16.1 引言	337
16.2 目标和假设	338
16.3 语法	339
16.4 结组	340
16.5 资源管理	341
16.6 唤醒与终止	343
16.7 非同步事件	346

16.8 多层传播	347
16.9 静态检查	347
16.10 不变式	350
第17章 名称空间	351
17.1 引言	351
17.2 问题	352
17.3 解决方案的思想	354
17.4 一个解决方案: 名称空间	355
17.4.1 有关使用名称空间的观点	357
17.4.2 使名称空间投入使用	358
17.4.3 名称空间的别名	359
17.4.4 利用名称空间管理版本问题	360
17.4.5 细节	361
17.5 对于类的影响	366
17.5.1 派生类	366
17.5.2 使用基类	368
17.5.3 清除全局的static	368
17.6 与C语言的兼容性	369
第18章 C语言预处理器	371
索引	375