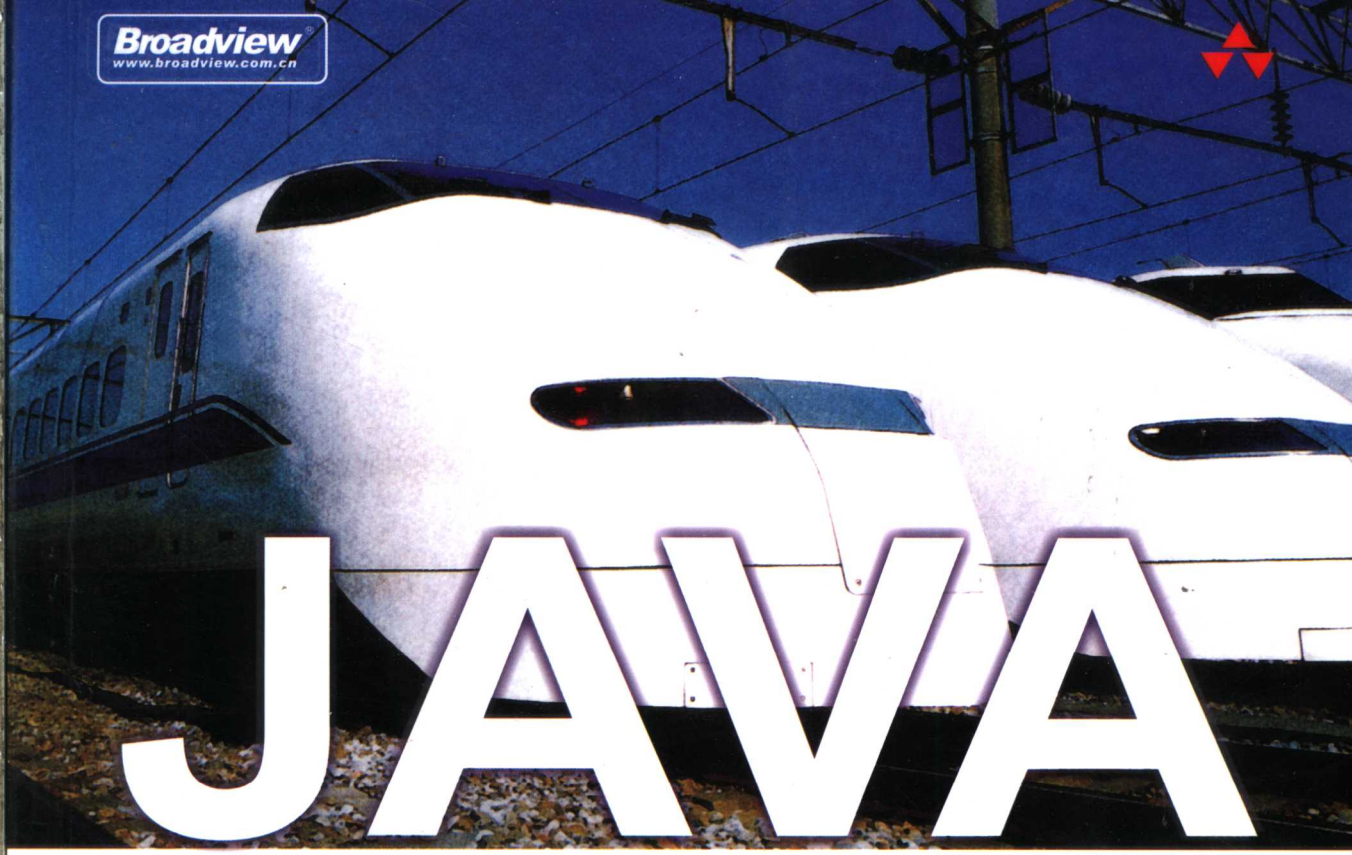
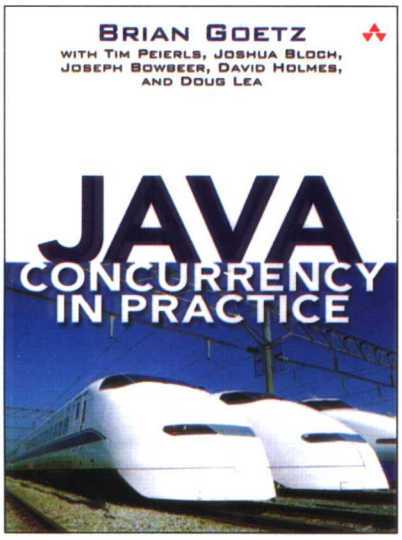
Broadview
www.broadview.com.cn



JAVA 并发编程实践

JAVA CONCURRENCY IN PRACTICE



[美] Brian Goetz, Tim Peierls, Joshua Bloch, Joseph Bowbeer, David Holmes, Doug Lea 著

韩锴 方妙 译

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

JAVA 并发编程实践

JAVA CONCURRENCY IN PRACTICE

[美] Brian Goetz, Tim Peierls, Joshua Bloch, Joseph Bowbeer, David Holmes, Doug Lea 著

韩 锴 方 妙 译

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

随着多核处理器的普及，使用并发成为构建高性能应用程序的关键。Java 5 以及 6 在开发并发程序中取得了显著的进步，提高了 Java 虚拟机的性能以及并发类的可伸缩性，并加入了丰富的新并发构建块。在本书中，这些便利工具的创造者不仅解释了它们究竟如何工作、如何使用，还阐释了创造它们的原因，及其背后的设计模式。

本书既能够成为读者的理论支持，又可以作为构建可靠的、可伸缩的、可维护的并发程序的技术支持。本书并不仅仅提供并发 API 的清单及其机制，还提供了设计原则、模式和思想模型，使我们能够更好地构建正确的、性能良好的并发程序。本书适合于具有一定 Java 编程经验的程序员、希望了解 Java SE 5 以及 6 在线程技术上的改进和新特性的程序员，以及 Java 和并发编程的爱好者。

图书在版编目（CIP）数据

JAVA 并发编程实践 /（美）戈茨（Goetz,B.）等著；韩锴，方妙译. —北京：电子工业出版社，2007.6

书名原文：JAVA CONCURRENCY IN PRACTICE

ISBN 978-7-121-04316-1

I. J… II. ①戈…②韩…③方… III. JAVA 语言－程序设计 IV. TP312

中国版本图书馆 CIP 数据核字（2007）第 063572 号

策划编辑：方 舟
责任编辑：周 筠 王继花
印 刷：北京智力达印刷有限公司
装 订：北京中新伟业印刷有限公司
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：27.5 字数：585 千字
印 次：2007 年 6 月第 1 次印刷
定 价：58.00 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 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。服务热线：（010）88258888。

对《JAVA 并发编程实践》的赞誉

Advance praise for JAVA CONCURRENCY IN PRACTICE

我曾经和一个梦幻般的团队一起工作，设计并实现了在 Java 5.0 和 Java 6 中为 Java 平台引入并发特性的功能，为此我感到无上的荣幸。如今，还是同一个团队，提供了目前为止对这些特性以及更一般的并发性的最佳阐释。并发不再仅仅是高级用户谈论的主题。每一名 Java 开发者都应该阅读本书。

—Martin Buchholz
JDK Concurrency Czar, Sun Microsystems

过去的近 30 年间，计算机性能一直由摩尔定律来推动；从今天起，它将由 Amdahl 定律推动。编写能够高效利用多处理器的代码，将会成为很大的挑战。在为今天以及未来的系统编写安全、可伸缩的 Java 程序时，你一定会发现《JAVA 并发编程实践》提供的概念和技术很有用。

—Doron Rajwan
Research Scientist, Intel Corp

如果你正在编写——或者设计、调试、维护、研究——多线程 Java 程序，这本书正是为你所写。如果曾经写过同步化的方法，但是并不确定其中的缘由，那么你应该给自己和你的用户都买一本来阅读，从头读到尾。

—Ted Neward
《Effective Enterprise Java》的作者

Brain 以不凡的清晰性，回答并解决了并发中的基本问题和复杂问题。对于使用线程和关心性能的人，这是一本必读书。

—Kirk Pepperdine
CTO, JavaPerformanceTuning.com

这本书言简意赅地覆盖了相当艰深和微妙的主题，使它成为一部理想的 Java 并发参考手册。每一页都充斥着程序员每日与之斗争的问题（和解决它的办法！）。今天，维系摩尔定律的不是更快的内核，而是更多的内核。有效地利用并发性在当今变得越来越重要，而这本书将向你展示如何做到这一切。

—Dr. Cliff Click
Senior Software Engineer, Azul Systems

我对并发有强烈的兴趣，与大多数程序员相比，我大概已经写了更多的线程死锁，也犯了更多的同步错误。Brian 的书在 Java 线程和并发主题中，是可读性最好的一本。通过精彩的循序渐进的方式，这本书讲解了这些困难的主题。我把这本书推荐给 Java Specialists' Newsletter 上的所有读者，因为这本书很有趣，也很有用，而且关系到今天 Java 开发者面对的问题。

—Dr. Heinz Kabutz
The Java Specialists' Newsletter

我所关注的职业是让简单的工作更简单，不过这本书却雄心勃勃并且富有成效地致力于简化一个复杂而关键的主题：并发。《JAVA 并发编程实践》以其革命性的讲述方式，明快而简洁的风格，以及及时地得以面世——它注定会成为一部极为重要的书。

—Bruce Tate
《Beyond Java》的作者

《JAVA 并发编程实践》是献给 Java 开发者的关于线程"所以然"的一份无价的作品。我发现阅读本书让人感到思维上的愉悦，部分原因是它对 Java 并发 API 的介绍相当精彩，但是更重要的原因在于，这本书以透彻的、可理解的方式，揭示了专家关于线程方面的知识，这在其他地方是很难找到的。

—Bill Venners
《Inside the Java Virtual Machine》的作者

To Jessica

作者为中文版作的序

能够为《JAVA 并发编程实践》中文版书写序言，我深感荣幸。

一年前，当这本书首次出版发行的时候，我们曾经写道“多核处理器正变得越来越便宜，足可以用于中端桌面系统了”。现在看来，多处理器系统正变得越来越便宜这一趋势还在继续，如果有什么不同，只是愈演愈烈了。几乎所有主要的计算机生产厂商都在初级笔记本电脑和台式机中加入了多核处理器特性，而在服务器级别的机器上，每个处理器的内核数量也多于去年的内核数量。事实上，寻找出单处理器的系统正变得越来越难了。

这些硬件的发展趋势使软件的开发者面临着严峻的挑战。我们不能满足于仅仅在新的CPU 上运行现有的程序，使它们跑得更快；如果我们想要发挥全新的处理器的能力，就必须编写程序来支持并发环境。在做这件事的过程中，会在架构、编程和测试方面遇到重大的挑战。而本书的目的就在于：为了让你能够应付这些挑战，书中提供了技术、模式和工具，可以用来分析并发编程并降低封装并发交互的复杂性。

现在，与以往任何时候相比，“并发”与每一位 Java 开发者都更加息息相关了。我们很高兴地看到，在中国有几十万的 Java 程序员将会成为本书的读者。

Brian Goetz

2007 年 2 月于 Williston, VT

Preface to the Chinese Edition

It is a great honor to be able to introduce the Chinese edition of Java Concurrency in Practice.

When this book was first published, one year ago, we wrote that "multicore processors are just becoming inexpensive enough to appear in midrange desktop systems." Now we see that this trend towards cheap multiprocessor systems has continued, and if anything, accelerated. Entry-level laptops and desktops from nearly every major computer manufacturer feature multi-core processors, and server-class machines are featuring even more cores per processor than they were last year. Indeed, it is getting hard to find single-processor systems.

These trends in hardware pose significant challenges for software developers. No longer can we just run our existing programs on new CPUs and have them run faster; our programs must be written to support concurrent environments if we want to take advantage of the power of newer processors. And doing so presents significant architectural, programming, and testing challenges. It is the goal of this book to respond to these challenges by offering techniques, patterns, and tools for analyzing concurrent programs and for encapsulating the complexity of concurrent interactions.

Concurrency is now more relevant than ever for Java developers. We are pleased that the audience for this book now includes the hundreds of thousands of Java programmers in China.

Brian Goetz

Williston, VT

February 2007

推荐序

在汗牛充栋的 Java 图书堆中，关于并发性的书籍却相当稀少，然而这本书的出现，将极大地弥补这一方面的空缺。即使并发性编程还没进入到您的 JAVA 日常开发当中来，您也应当花些时间来阅读这本重要的图书。该书是由 developerWorks 《JAVA 理论与实践》 *http://www.ibm.com/developerworks/cn/java/j-jtp/* 专刊的作者 Brian Goetz (*http://www.briangoetz.com/*) 执笔，他曾是 Quiotix 软件开发和咨询公司的首席顾问，现在是 Sun Microsystems 的高级工程师，并效力于多个 JCP 专家组。他作为专业的软件开发人员已经有 20 年的经验了，其在 Java 并发性领域的研究与贡献是有目共睹的。

这是一本目前在 Java 并发性领域研究的编程图书中最值得一读的力作。随着计算机技术的不断迅速发展，各种各样的编程模型也越来越多，越来越复杂化与多样化。虽然当前 CPU 主频在不断升高，但是 X86 架构的硬件已经成为瓶颈，这种架构的 CPU 主频最高为 4GHz，事实上目前 3.6GHz 主频的 CPU 已经接近顶峰，多线程编程模型不仅是目前提高应用性能的手段，更是下一代编程模型的核心思想。它的目的就是“最大限度地利用 CPU 资源”，当某一线程的处理不需要占用 CPU 而只需要 I/O 等其他资源时，就可以让需要占用 CPU 资源的其他线程有机会获得 CPU 资源。因此，就目前来说，多线程编程模型仍是计算机系统架构的最有效的编程模型。

Java 提供了语言级的多线程支持，所以在 Java 中使用多线程相对于在 C/C++ 当中使用多线程来说更加简单与快捷。除了 Brian Goetz 自己的研究、经验和热心读者的贡献之外，本书还吸取了一些并发性前沿人员的真知灼见，包括 Tim Peierls、Joshua Bloch、Joseph Bowbeer、David Holmes 和 Doug Lea。在该书中， Brian Goetz 从最基本的知识开始介绍，首先集中描述了在 Java 平台上创建线程应用程序以及同步对共享资源的访问时的细微之处；然后分析了 Java SE 5 提供的更高层次的线程执行构造，以及如何最好地把它们应用到现实世界中的不同场景，并整合了一些最佳实践和最新的研究主张；再就现实中的生存保证、性能、可伸缩性和可测试性的困难问题进行了分析，并把当前的最佳实践调查与相关的研究结果相结合，提供了一些可行的替代方案；最后介绍了一些在开发中可能适用的高级并发性技术，包括显式锁、定制同步器、原子变量与非阻塞同步，还介绍了低级的 Java 存储模型。同时，在全书贯穿了许多简洁的代码示例，用来演示问题和可行

的解决方案。

当我们从今天以应用程序为核心的开发平台转移到不远的未来支持多核处理器的操作系统和平台机制时，《JAVA 并发编程实践》代表了这个容易出错的领域当前最新的并发性实践和研究。相信这一本优秀的图书将是您案头的必备书籍，强烈建议您阅读并实践之。

俞黎敏

2007 年 3 月于上海

译者序

随着译稿最后一个字符的键入，我长舒一口气，目光不知不觉中落到了我的新笔记本电脑上，上面赫然贴着某处理器厂商新推出的多核处理器的 Logo。几个月来不断地与并发打交道，让我深深地感受到，我们已经进入了一个全新的时代——多核时代。

回想十几年前，提起多处理器或者多核系统，人们总会想到被摆放在实验室中发着轰鸣响声的机器，它们通常体型庞大，而且价格不菲，很多程序员对它们都是顶礼膜拜，又有多少人能为它们编写程序呢？在很多程序员心中，“并发”也已经成为了“高深”的代名词。

造成这种现象的原因有两点。其一，在很多人心中，编写“并发”程序相对于编写“串行”程序更加困难。“并发专家”也因此被笼罩了一层神秘的面纱。其二，没有“需求”。多余的并发性在单处理器系统上显得有些力不从心。程序员的懒惰让他们很少有时间去编写没有人需要的代码。几年前，大量客户端代码都是运行在单处理器环境中的。在这种情况下，如果产生多个线程完成同一件事（比如压缩文件），效率只能随着线程生命周期管理与上下文切换而降低。

然而今天，多核已经成为不可阻挡的历史潮流了。计算机在人们的生产生活中发挥着越来越重要的作用。无论是科学计算、商业系统还是个人桌面应用，对性能的要求始终在增长。另一方面，流行 40 年之久的“摩尔定律”已渐渐显露疲态，电子电路的物理极限是无法突破的。在这样的环境下，“多核”应运而生。在一块芯片上嵌入更多的处理单元，相对于复杂昂贵的多核系统，是提升系统并行性与性能的简单、经济的途径，它更容易被普通大众所接受与使用。如果你正在准备或者刚刚购买新的个人电脑，你注意到它的处理器中包含了多少个内核了吗？

多核的影响是广泛且深入的。企业需要改变以往基于 CPU 数的计价方式。系统设计者需要重新审视多核产生的并行性带来的影响。一线的开发人员要学习新的开发思路、技巧和工具。并行的理念需要融入到系统的设计与实现的过程中。对程序员来说，多核带来的变化并不像时钟频率增加那样透明。以前我们已经对软件加入了很多的期望，比如可扩展性、健壮性、可伸缩性、可测试性，等等，今天我们还要考虑软件是否充分发挥了微处理器的全部性能。只有充分挖掘程序的并行性，才能让多核处理器物尽其用，才能让你的软件在今后内核不断增加的日子里，得以保持升级。

面对这些挑战，不禁令人有些担心。好在现代先进的运行时环境和编译器都对并行做好了大量的优化准备。更难得的是，像 Java 这样的高级语言提供了对并发的语言级支持。这个特性正是 Java 的伟大创新之一。尤其是 Java 5 的发布，更是 Java 并发开发的质的飞跃。若干年前必须依靠本机代码才能实现的功能，如今可以使用纯 Java 代码实现；若干年前只有并发专家才能开发的程序，如今你也可以编写出来；若干年前极易产生 bug 的功能模块，如今可以使用 Java 核心类轻松实现。而 Brian 的这本书就是帮你做到这一点的最佳指南。Brian 以其特有的师者风范，以精准而不失诙谐的文字，将 Java 并发开发的奥秘抽丝剥茧，让读者在享受阅读乐趣的同时，获得广博而深入的并发知识和技能。这些足以帮助开发者消除（减轻）多核时代的“危机”。

如果是孤军奋战，很难想象可以完成这部著作的翻译，期间我们得到了太多的帮助，即使挂一漏万，也要尽力列在这里：

首先要感谢本书的原作者 Brian Goetz 先生。如果没有他的付出，就谈不上这个译本的出现。还要感谢他不厌其烦地回答了我们大量的问题。Brian 每次平易近人和极为认真的回复，都让我们在提高译本质量的过程中受益匪浅。

法律专业出身的原莹小姐，感谢你以极大的勇气，踏入这个对你来说并不熟悉的领域。得益于你的辛劳，这本书的文字才会摆脱一些计算机科学的冰冷。

感谢博文视点的各位老师和编辑，尤其是晓菲。你们的敬业与认真，深深地感动了我。

感谢中国人民银行软件开发中心的各位同事，是你们的宽容，让我有更多的时间投入到这本书的翻译工作中。

最后，也是最重要的，感谢这本书的另一位合译者——方妙小姐。你的专业知识和文字功底让我惊叹。与你的合作是一段令人愉快且难忘的经历。

……

计算机领域从来都不乏变革，每一次都带来新的机遇与挑战。“多核”无疑是一场令人激动的变革。迎接技术变革与挑战，是软件开发者的空气和水。很高兴你已经读到了这里，那么请不要停下来，继续享受挑战的乐趣，继续面对这场变革，前进吧！

韩锴

2007 年 3 月于北京

译者简介

韩锴，毕业于北京工业大学软件工程专业；目前就职于中国人民银行软件开发部，负责战略发展部的核心框架开发，对软件开发有其独到的理解和认识。目前主要研究方向为：敏捷软件过程方法论、Eclipse 平台及其相关技术、Java 下的并发编程，等等。联系方式为：*isaachanstar@gmail.com*。

方妙，毕业于北京工业大学软件工程专业；目前就职于 Ivar Jacobson 软件咨询公司中国分公司，担任软件设计工程师，从事新一代过程构建与 Web 2.0 开发，对 Java 技术、轻量级过程方法论与架构设计有浓厚的兴趣。联系方式为：*miaofang@gmail.com*。

联系博文视点

您可以通过如下方式与本书的出版方取得联系。

读者信箱：*sheguang@broadview.com.cn*

投稿信箱：*broadvieweditor@gmail.com*

北京博文视点资讯有限公司（武汉分部）

湖北省 武汉市 洪山区 吴家湾 邮科院路特 1 号 湖北信息产业科技大厦 1402 室

邮政编码：430074

电话： (027)87690813　　传真：(027)87690813 转 817

若您希望参加博文视点的有奖读者调查，或对写作和翻译感兴趣，欢迎您访问：

http://bv.csdn.net

关于本书的勘误、资源下载及博文视点的最新书讯，欢迎您访问博文视点官方博客：

http://blog.csdn.net/bvbook

序

Preface

写作本书时，出于桌面系统的迫切需求，多核处理器正在变得越来越便宜。与此不协调的是，很多开发团队还没有注意到，在他们的项目中，出现了越来越多的关于线程的错误报告。在 NetBeans 开发者站点上最近的一次通告中，一位核心维护者注意到，为了修复某个类的一个线程相关的问题，已经被打了 14 次补丁。Dion Almaer，前 TheServerSide 的编辑（经过一次痛苦的调试过程，最终发现是一个线程的 bug 之后），最近在 Blog 上写道，大多数 Java 程序都充斥着并发 bug，它们仅仅是“碰巧”可以工作。

的确，由于并发性的 bug 不会以可预见的方式自己“蹦”出来，因此多线程程序的开发、测试和调试都会变得极端困难。bug 浮出水面的时刻，通常可能是最坏的时候——对应于生产环境，就是指在高负载的时候。

使用 Java 开发并发程序所要面对的挑战之一，是要面对平台提供的各种并发特性之间的不匹配，还有就是程序员在他们的程序中应该如何思考并发性。语言提供了一些低层机制，比如同步和条件等待，但是这些机制在实现应用级的协议与策略时才是必须的。不顾这些策略的约束，很容易创建出一个程序，它在编译和运行时看上去一切正常，不过这其中却存在隐患。很多并发方面相当不错的书都没能达到预期的目标，它们过分地关注于低层的机制和 API，而不是设计层面的策略和模式。

Java 5.0 是在使用 Java 开发并发应用程序的进程中，迈出的巨大一步。它提供了新的高层组件以及更多的低层机制，这些将使得一名新手更容易像专家那样去构建并发应用程序。本书的作者都是 JCP 专家组的主要成员，正是这个专家组创建了这些新工具；除了去描述新工具的行为和特性，我们还向您展示了它们低层的设计模式，预期的使用场景以及将它们纳入平台核心库的动机。

我们的目标是给读者一些设计法则和理念模型，让读者在使用 Java 构建正确、高效的并发类和应用程序时，变得更容易、更有趣。

我们希望你在阅读《JAVA 并发编程实践》的过程中能够获得愉悦感。

Brian Goetz

2006 年 3 月于 Williston, VT

如何使用本书

Java 低层机制与设计层必要的策略之间存在着抽象的不匹配。为了解决这个问题，我们呈现了一个经过简化的规则集，用来编写并发程序。专家们看到这些规则会说："嗨，这可不是完整的规则，类 C 即使破坏了规则 R，它仍然可以是线程安全的。"尽管打破我们的规则，也可能写出正确的并发程序，不过这样做需要对 Java 存储模型的低层细节有着深入的理解。我们的愿望是：开发者**不用**熟悉这些细节，就有能力编写正确的并发程序。只要坚持遵守我们简单的规则，就能编写出正确的、可维护的并发程序。

我们假设读者已经具备了对 Java 基本并发机制的一些了解。《JAVA 并发编程实践》不是并发的入门指南——关于这个，可以参看任何正统的介绍性的大部头书籍，比如《Java 编程语言》（Arnold et al., 2005）。本书也不是关于并发的百科全书似的参考手册——关于这个，可以参看《JAVA 并发编程（Lea, 2000）》。对本书更恰当的描述是，它提供了实际的设计规则，可以协助开发者，他们正在处于创建安全的、高效的并发类这一艰难的过程中。在适当的地方，我们穿插引用了下列图书的章节：《The Java Programming Language》、《Concurrent Programming in Java》、《The Java Language Specification（Gosling et al., 2005）》以及《Effective Java（Bloch, 2001）》，并使用[JPL n.m]、[CPJ n.m]、[JLS n.m]和[EJ Item n]来表示它们。

结束"介绍"（第 1 章）之后，本书分为 4 部分：

基础。第 1 部分（第 2～5 章）关注于同步和线程安全的基本概念，以及如何使用类库提供的构建块组合线程安全类。第 110 页上，有一个"并发诀窍清单"总结了出现在第 1 部分中最重要的规则。

第 2 章（线程安全性）与第 3 章（共享对象）构成了全书的基础。几乎所有用来避免并发危险、创建线程安全类以及验证线程安全的规则都包括在这里。轻"理论"，重"实践"的读者可能会禁不住诱惑跳过这部分，而直接进入第 2 部分，但是在开始编写任何并发代码之前，一定要确保回过头来阅读这两章！第 4 章所涵盖的技术，用于把线程安全类组合到更大的线程安全类中。第 5 章（构建块）涵盖了平台核心库提供的并发构建块——线程安全的容器和同步工具（synchronizer）。

构建并发应用程序。第 2 部分（第 6～9 章）描述了如何利用线程提高并发应用程序的吞吐量或响应性。第 6 章（任务执行）讲述如何识别可并行执行的任务，并在任务执行框架内部执行它们。第 7 章讲到的技术可以让任务和线程在正常终止之前妥善地终止；区分并发应用程序是健壮的，还是仅仅可以将就工作的，有众多的因素，"程序如何处理取消与关闭"就是其中之一。第 8 章（应用线程池）关注了一些任务执行框架中更为高级的特性。

第 9 章（GUI 应用程序）关注了用来提高单线程化子系统响应性的技术。

活跃度、性能和测试。第 3 部分（第 10～12 章）涉及并发程序自身。要确保并发程序执行了你所希望它做的事情，而且性能是可以接受的。第 10 章（避免活跃度危险）描述了如何避免活跃度失败，活跃度失败会阻止程序继续向前执行。第 11 章（性能和可伸缩性）涵盖的技术用来提高并发代码的性能和可伸缩性。第 12 章（测试并发程序）涵盖的技术用来测试并发代码的正确性和性能。

高级主题。第 4 部分（第 13～16 章）涵盖的主题可能只会引起资深程序员的兴趣：它们是显式锁、原子变量、非阻塞算法和开发自定义的 synchronizer。

代码示例

尽管书中的很多通用概念适用于 Java 5.0 之前版本，甚至是非 Java 环境，不过大多数示例代码（以及关于 Java 存储模型的每一句话）都假定是以 Java 5.0 或更新的 JDK 为基础的。有些代码示例还会用到 Java 6 中添加到类库中的特性。

代码示例已经被裁减，以降低它们的尺寸和突出相关的部分。完整版本的代码示例、辅助示例和勘误表，可以从本书的网站 *http://www.javaconcurrencyinpractice.com* 上获得。

代码示例分为三类："好"示例，"一般"示例和"坏"示例。"好"示例阐释的技术应该被效仿。"坏"示例阐释的技术绝对**不**应该被效仿，而且还会用一个"Mr. Yuk"[1]的图标清楚地表明这是"有害"的代码（参见清单 1）。"一般"示例阐释的技术不一定是错的，但却是是脆弱的、有风险的或者执行效果差的，而且会用一个"Mr. CouldBeHappier"图标标识出来，如同清单 2。

清单 1　糟糕的列表排序方法（不要这样做）

```
public <T extends Comparable<? super T>> void sort(List<T> list) {
// 永远不要返回错误的答案
System.exit(0);
}
```

有些读者会质疑"坏"示例在本书中的角色；毕竟，一本书应该展现如何做正确的事，而不是错误的事。"坏"示例有两个目的。它们揭示了常见的缺陷，更重要的是它们示范了如何分析程序的线程安全性——完成此事的最佳办法就是观察威胁线程安全的各种方式。

[1] Mr. Yuk是the Children's Hospital of Pittsburgh的注册商标，以授权在本书中使用。