TURING 图灵程序设计丛书

MANNING

The Well-Grounded Java Developer

Vital Techniques of Java 7 and Polyglot Programming

Java 程序员修炼之道

[英] Benjamin J. Evans
[荷兰] Martijn Verburg 著
吴海星 译

涵盖Java 7最新特性

全面解读Groovy、Scala和Clojure在JVM上的应用

透视函数式编程的优势

人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书

The Well-Grounded Java Developer

Vital Techniques of Java 7 and Polyglot Programming

Java 程序员修炼之道

[英] Benjamin J. Evans
[荷兰] Martijn Verburg 著

吴海星 译

人民邮电出版社
北 京

图书在版编目（CIP）数据

Java程序员修炼之道 / （英）埃文斯（Evans, B. J.），（荷）费尔堡（Verburg, M.）著；吴海星译. -- 北京：人民邮电出版社，2013.8
（图灵程序设计丛书）
书名原文：The well-grounded Java developer:vital techniques of Java 7 and polyglot programming
ISBN 978-7-115-32195-4

Ⅰ. ①J… Ⅱ. ①埃… ②费… ③吴… Ⅲ. ①JAVA语言－程序设计 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2013)第138623号

内 容 提 要

本书分为四部分，第一部分全面介绍 Java 7 的新特性，第二部分探讨 Java 关键编程知识和技术，第三部分讨论 JVM 上的新语言和多语言编程，第四部分将平台和多语言编程知识付诸实践。从介绍 Java 7 的新特性入手，本书涵盖了 Java 开发中最重要的技术，比如依赖注入、测试驱动的开发和持续集成，探索了 JVM 上的非 Java 语言，并详细讲解了多语言项目，特别是涉及 Groovy、Scala 和 Clojure 语言的项目。此外，书中含有大量代码示例，帮助读者从实践中理解 Java 语言和平台。

本书适合 Java 开发人员以及对 Java7 和 JVM 新语言感兴趣的各领域人士阅读。

◆ 著　[英] Benjamin J. Evans　[荷兰] Martijn Verburg
译　吴海星
责任编辑　刘美英
执行编辑　李　洁
责任印制　焦志炜

◆ 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：26
字数：658千字　2013年8月第1版
印数：1－3 000册　2013年8月北京第1次印刷
著作权合同登记号　图字：01-2012-8794号

定价：89.00元

读者服务热线：(010)51095186转604　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

序

“Kirk说加油站也卖啤酒。”这是Ben Evans跟我说的第一句话。他来克里特岛参加一个开放型Java会议。我说我通常到加油站就是加油，但那边拐角确实有个店卖啤酒。Ben看起来对我的回答有点儿失望。我在这个希腊小岛上生活了5年，还从来没在加油站买过啤酒。

当我在看这本书时，那种似曾相识的感觉又来了。我自认为是一名Java专家：用Java写了15年程序，发表了几百篇文章，在各种会议中演讲，还执教Java高级课程。可阅读Ben和Martijn的这本大作，经常能给我一些意料之外的启发。他们一开始先介绍了为改变Java生态系统所做的开发工作。类库的内部实现修改起来相对容易，一般也能见到成效。例如，`Arrays.sort()`的内部实现在Java 7中不再用MergeSort算法，而是改用了TimSort。由于这个变化，你不用修改自己对偏序数组进行排序的代码就可能看到性能的提升。然而，修改类文件格式或添加新的VM特性则需要大量工作。Ben了解这些情况，因为他在JCP执行委员会任职。这本书也是关于Java 7的，所以你能接触到Java 7中的所有新特性，比如语法糖的完善、`String`上的`switch`、分支/合并，还有Java NIO.2。

并发就是线程和同步，对吗？如果这就是你对多线程的认识，那么你需要学习新知识了。就像作者在书中指出的，“并发领域的研究工作正开展得热火朝天”。与并发相关的邮件列表上每天都有讨论，新点子层出不穷。本书会告诉你如何看待分而治之策略以及如何规避某些安全陷阱。

在我看到类加载那一章时，我觉得他们说得有点儿过了。那都是我和朋友们过去用来炫耀的技巧，居然也给摆出来供大家研习了！他们讲解了`javap`（这个小工具用于透视Java编译器生成的字节码）的工作方式，还谈到了新的`invokedynamic`指令，并解释了它跟普通反射的区别。

我特别喜欢讲性能调优的第6章。除了Jack Shirazi的*Java Performance Tuning*，这还是第一本能够抓住“如何使系统运行更快”这个本质问题的书。我可以用四个字来总结这一章的内容：“测量，别猜。”这是做好性能调优的本质，因为人们不可能猜到运行慢的是哪段代码。这一章从硬件的角度解读了性能方面的问题，而不是只提供编码技巧。作者还向你展示了如何测量性能。有一个挺有意思的基础测试小工具——`CacheTester`类，可用于查看缓存未命中时的开销。

本书第三部分介绍了JVM上的多语言编程。Java不仅仅是Java编程语言，它还是一个可以运行其他语言的平台。我们已经见过不同类型语言的爆炸式增长了。有些是函数式的，有些是声明式的，还有一些是平台的接口（Jython和JRuby），让其他语言可以在JVM上运行。语言分为动态的（如Groovy）和静态的（如Java和Scala）。在JVM上我们可能因为多种原因而使用非Java的语言。如果正好要开始一个新项目，在做决定之前先看看都有什么可用吧。你可能不用再写那么多套路化的代码了。

Ben和Martijn向我们介绍了三种备选语言：Groovy、Scala和Clojure。在我看来，它们是当下最切实可行的选择。作者描述了这些语言之间的差异、与Java的差异以及它们的特性。不需要太多的技术细节，介绍每种语言的各章足以帮你弄清楚应该用哪一种。别指望能在书中看到Groovy的参考手册，但你会了解哪种语言更适合你。

之后，你将深入了解如何进行测试驱动开发以及如何持续集成系统。我发现一件很有意思的事，忠实的“老管家”Hudson这么快就被Jenkins取代了。无论如何，这些工具跟Checkstyle和FindBugs一样都是项目管理的基本工具。

你有望通过研读本书成为一名优秀的Java开发人员。不仅如此，你还能了解如何保持优秀。Java一直在变。下一版中我们将见到lambda表达式和模块化。[①]人们也在不断设计新语言，不断更新并发结构。你现在了解的很多真相将来可能不再是真相。因此，我们必须活到老学到老！

一天，我又开车路过Ben想买啤酒的那个加油站。在经济状况如此低迷的希腊，它也像很多公司一样关张了。我再也不可能知道他们卖不卖啤酒了。

Heinz Kabutz博士

知名Java技术教育家、The Java Specialists'Newsletter创始人

① 实现Java模块化的Jigsaw项目被延后到Java 9了，至少要到2015年。——译者注

前　言

本书最开始是给德意志银行外汇IT部的新人准备的培训笔记。Ben觉得市面上没有面向经验匮乏的Java开发人员的书，所以决定写一本来填补这个空白。

在德意志银行IT管理团队的支持下，Ben去了比利时的Devoxx会议寻找灵感。在那里他见到了IBM的三位工程师（Rob Nicholson、Zoe Slattery和Holly Cummins），他们把他引荐给了伦敦Java社区（LJC，伦敦Java用户组）。

接下来的周六正好是LJC组织的年度开放会议，就在那次会议上，Ben遇到了LJC的一位领导者——Martijn Verburg。两人一见如故，把酒言欢，惺惺相惜，大有相见恨晚之意。也正是两人对技术和教学的共同热爱促成了本书。

软件开发是一项社会活动，我们希望能借助本书唱响这一主题。我们认为，虽然在这项活动中技术占有很重要的地位，但人与人之间微妙的沟通和交互关系也不容忽视。要在书里轻松解释这些东西并不容易，但这一主题自始至终贯穿本书。

凭借着对技术的执着和对学习的热爱，开发人员孜孜不倦地工作着。我们希望本书讨论的一些话题能够激发他们的学习热情。这是一次观光之旅，而不是百科全书式的灌输，这就是我们的初衷：帮助你入门，然后让你自己去探索那些激发你想象力的东西。

本书不仅为大学毕业生准备了接引指南，更为所有心有困惑的Java开发人员提供了指导。因为他们都很想知道："接下来我该学什么？未来要向什么方向发展？我要再好好考虑考虑！"

从Java 7的新特性到现代软件开发的最佳实践，再到平台的未来发展，本书一路向前，向你展示在成长为资深Java开发人员的过程中我们认为至关重要的那些知识。并发、性能、字节码和类加载是最让我们着迷的核心技术。我们还会谈到JVM上那些新的非Java语言（即多语言编程），因为在接下来的几年里，对于很多开发人员来说它们将变得越来越重要。

归根结底，这是一次以你和你的兴趣为核心的、具有前瞻性的旅程。我们认为成为一名优秀的Java开发人员有助于你彻底投入到工作中去并顺利驾驭开发，也有助于你对不断变化的Java世界及它的周边生态系统有更多了解。

我们希望这本"经验的结晶"对你来说既实用又有趣，希望它能让你深思，同时还能带给你快乐。无论如何，写这本书的体验确实如此！

致　谢

老话说得好，“众人拾柴火焰高”。对于本书来说，这句话非常贴切。如果没有朋友、亲人、同事、同行，甚至偶尔跟我们对立的那些人，我们就不可能完成本书。我们一直都非常幸运，因为那些对我们批评最强烈的人也可以算是我们的朋友。

帮助过我们的人太多了，很难全部列举出来。本书快付印的时候，我们在http://www.java7developer.com上发过一个帖子，其中也列出了一批名单，那些人值得我们感谢。

如果名单里遗漏了谁，都怪我们没有牢记您的大名，请接受我们的歉意！下面这些人对本书出版都有贡献，在此一并感谢（排名不分先后）。

伦敦 Java 社区

伦敦Java社区（LJC，www.meetup.com/londonjavacommunity）是我们两位作者相遇相知的地方。我们要感谢下面这些帮忙审校本书的人：Peter Budo、Nick Harkin、Jodev Devassy、Craig Silk、N. Vanderwildt、Adam J. Markham、“Rozallin”、Daniel Lemon、Frank Appiah、P. Franc、“Sebkom” Praveen、Dinuk Weerasinghe、Tim Murray Brown、Luis Murbina、Richard Doherty、Rashul Hussain、John Stevenson、Gemma Silvers、Kevin Wright、Amanda Waite、Joel Gluth、Richard Paul、Colin Vipurs、Antony Stubbs、Michael Joyce、Mark Hindess、Nuno、Jon Poulton、Adrian Smith、Ioannis Mavroukakis、Chris Reed、Martin Skurla、Sandro Mancuso和Arul Dhesiaseelan。

在Java语言之外，我们得到了James Cook、Alex Anderson、Leonard Axelsson、Colin Howe、Bruce Durling和Russel Winder博士非常认真的指导。在这里要特别感谢他们。

我们还要特别感谢LJC JCP委员会成员：Mike Barker、Trisha Gee、Jim Gough、Richard Warburton、Simon Maple、Somay Nakhal和David Illsley。

最后，感谢LJC的发起人Barry Cranford。四年前，他带领几个勇敢的人，怀抱一个梦想创建了LJC。现在，LJC已经有约2500名成员，并且很多其他技术社区也发源于它。LJC已经成为伦敦技术界的中流砥柱。

www.coderanch.com

我们要感谢Maneesh Godbole、Ulf Ditmer、David O’Meara、Devaka Cooray、Greg Charles、Deepak Balu、Fred Rosenberger、Jesper De Jong、Wouter Oet、David O’Meara、Mark Spritzler和

Roel De Nijs，因为他们提供了详细的评论和宝贵的反馈。

Manning 出版社

感谢Manning的Marjan Bace接受我们这两个有着疯狂想法的作者。在制作本书出版的整个过程中，我们跟许多人打过交道，非常感谢他们：Renae Gregoire、Karen G. Miller、Andy Carroll、Elizabeth Martin、Mary Piergies、Dennis Dalinnik和Janet Vail。毫无疑问，还要感谢那些我们从未见过面的幕后工作者。没有他们，就没有这本书！

感谢Candace Gillhoolley在营销上的努力，还有Christina Rudloff和Maureen Spencer一直以来的支持。

感谢John Ryan III在本书付印前对书稿的全面技术审校。

感谢那些在本书编写的不同阶段阅读原稿并给编辑和我们提供宝贵反馈的审校者：Aziz Rahman、Bert Bates、Chad Davis、Cheryl Jerozal、Christopher Haupt、David Strong、Deepak Vohra、Federico Tomassetti、Franco Lombardo、Jeff Schmidt、Jeremy Anderson、John Griffin、Maciej Kreft、Patrick Steger、Paul Benedict、Rick Wagner、Robert Wenner、Rodney Bollinger、Santosh Shanbhag、Antti Koivisto和Stephen Harrison。

特别致谢

感谢Andy Burgess为本书开发的网站www.java7developer.com，非常棒。感谢实习生Dragos Dogaru测试了所有代码示例。

感谢Matt Raible非常友善地允许我们在第13章引用他关于如何选择Web框架的一些内容。

感谢Alan Bateman，他领导了Java 7的NIO.2。他的反馈对于向所有Java开发人员普及这个伟大“新API”至关重要。

Jeanne Boyarsky友好地充当了我们最优秀的技术把关者。她果然名不虚传，什么都躲不过她鹰一般犀利的眼睛。感谢Jeanne！

感谢Martin Ling非常细致地讲解了计时硬件，那是第4章介绍相关内容的主要原因。

感谢Jason Van Zyl，他非常友善地允许我们在第12章引用Sonatype Company出版的*Maven: The Complete Reference*中的一些内容。

感谢Kirk Pepperdine对第6章的反馈和意见，还有他的热情及对这个行业独到的见解。

感谢Heinz M. Kabutz博士为我们写的推荐序和他在克里特岛的盛情款待，还有非常赞的Java专家简报（www.javaspecialists.eu/）。

Ben Evans 的致谢

许多人都以不同的方式为本书作出了贡献，但篇幅有限，就不一一致谢了。在此，特别感谢下面这些人。

感谢Bert Bates和其他Manning的员工，他们让我了解了稿件和书之间的区别。

感谢Martijn，当然，为了友谊，为了能帮我度过难关坚持写作，为了很多东西，总之一言难尽。

感谢我的家人，特别是我的外祖父John Hinton和祖父John Evans，我之所以成为我，是因为从他们那里继承了很多东西。

最后，感谢E-J（水獭在书中出现如此频繁的原因）和Liz，太多夜晚因为写作不能陪他们，感谢他们的理解。我的爱，献给他们。

Martijn Verburg 的致谢

感谢我的妈妈Janneke和爸爸Nico，感谢你们在我和姐姐小时候把Commodore 64电脑带回了家。尽管玩《跳跳人》（这真的是一个非常非常酷的平台游戏。老妈拿着操纵杆跑来跑去真是太好笑了:-））几乎成了我们在家里用电脑做得最多的事，但正是它的编程手册激发了我对技术的热情。爸爸还教会了我一个道理：如果你能把小事情做对，那么由小事情组成的大事也就不在话下了。我至今依然在编码和日常工作中将它奉为真理。

感谢我的姐姐Kim，感谢你小时候跟我一起写代码！我永远也忘不了第一个星域（缓慢地——那时，我可不太擅长做性能调优）出现在屏幕上时的场景，魔法真的显灵了！姐夫Jos给了我们俩很多灵感（不仅仅因为他是一位空间科学家，尽管那确实很酷！）。我那超级可爱的外甥女Gweneth也出现在了本书里，看你能不能找到她！

Ben是我认识的业内最棒的技术专家之一。有时他的技术水平可以用“吓人”来形容！我很荣幸能跟他合写本书，没想到关于JVM还有那么多知识我不知道。Ben一直都是LJC的优秀领导者，我俩在技术大会上一唱一和的演讲之后，甚至有人认为我们该一块去演喜剧了。能跟朋友合写一本书，真好。

最后，感谢我可爱的妻子Kerry，为了保证本书每一章的插图和屏幕截图都恰到好处，我们一次次地取消了活动计划，而你毫无怨言——你总是那么迷人。愿每个人都能拥有这样的爱和支持。

关于本书

欢迎阅读本书。通过阅读本书，你将成为紧跟时代潮流的Java程序员，重燃对这一语言和平台的热情。学习过程中，你会发现Java 7的新特性，熟悉重要的现代软件技术（比如依赖注入、测试驱动开发和持续集成），并开始探索JVM上的非Java语言这个美丽新世界。

首先，我们来看看James Iry在他精彩的博文“A Brief, Incomplete, and Mostly Wrong History of Programming Languages”（简明、不完整并且漏洞百出的编程语言历史）中对Java语言的描述：

> 1996年，James Gosling发明了Java。Java相对繁琐、基于类，是支持垃圾收集、静态类型、单派发的面向对象语言，继承方式为实现单继承和接口多继承。Sun大肆宣扬Java的新颖性。

他对Java的描述基本上是在插科打诨，C#在文中也受到了同等待遇。但作为对一种语言的描述，这种方式也不赖。博文还有很多精彩之处，参见James的博客（http://james-iry.blogspot.com/）。没事的时候看看还是挺有收获的。

James的描述的确提出了一个很实际的问题。为什么我们还要讨论一种有将近16年历史的语言呢？它真的已经稳定，没有多少新东西或有意思的事情值得探讨了吗？

如果真是那样，本书就会很薄。事实是，我们依然在谈论Java，因为它的一大优点就是其在以下几个核心设计决策之上的构造能力，这些都已经在市场中获得了成功：

- 运行时环境的自动管理（比如垃圾收集、即时编译）；
- 语法简单，核心概念相对较少；
- 保守的语言进化方式；
- 在类库中增加功能和复杂性；
- 广泛、开放的生态系统。

这些设计决策一直在推动着Java世界的创新，简单的核心使得开发门槛很低，而广阔的生态系统使得后来者很容易找到适合自己需要的现成组件。

尽管从历史趋势上来看语言的变化很缓慢，但这些特质使得Java平台和语言既强大又充满活力。Java 7仍然延续了这一趋势。语言的改变是演进式，而不是革命式的。然而，Java 7跟之前版本相比有一个主要区别：它是第一个明确着眼于下一次发布的新版本。根据Oracle有关发布的“B计划”，Java 7为Java 8的主要变化打下了基础。

近年来，JVM上非Java语言的崛起也一个重大变化。这引发了Java和其他JVM语言之间的相

互融合。现在有大量的项目完全运行在JVM之上（这个数量还在增加），而Java只是它们所用的编程语言之一。

多语言特别是涉及Groovy、Scala和Clojure语言的项目，是当前Java生态系统的一个重要因素，也是本书最后一部分的主题。

阅读须知

本书内容大体上适合顺序阅读，但我们也能理解某些读者想直奔主题的心情，因此也在一定程度上迎合了这种阅读需求。

我们非常认同自己动手的学习方法，所以建议读者在阅读的同时尝试示例代码。接下来介绍本书主要内容，希望习惯跳跃阅读的读者能从这里找到线索。

本书分四部分：

- 用Java 7做开发；
- 关键技术；
- JVM上的多语言编程；
- 多语种项目开发。

第一部分共两章，都是关于Java 7的内容。本书通篇使用Java 7的语法和语义，所以第1章“初识Java 7”是*必读*的。那些要处理文件、文件系统和网络I/O的开发人员应该会对第2章“新I/O”特别感兴趣。

第二部分共四章（第3~6章），涉及的主题包括依赖注入、现代并发、类文件/字节码以及性能调优。

第三部分共四章（第7~10章）介绍了JVM上的多语言编程。第7章是*必读*的，因为这一章介绍的JVM上可用语言的类型和使用是阅读后面章节的基础。接下来的三章分别介绍与Java类似的语言Groovy、兼具OO和函数式特色的混合语言Scala和纯函数式语言Clojure。刚接触函数式编程的开发人员可能需要按顺序阅读，但这几章本身是相互独立的，可以跳着读。

第四部分（最后四章）在之前内容基础上介绍了新内容。虽然各章可以独立阅读，但是在某些部分我们会假定你已经读过之前的内容，或者已经熟悉那些主题。

简言之，如果*整本书你必看一章*，那就看第1章。如果你会看第三部分，那*一定要看第7章*。其他各章既可以顺序阅读，也可以独立阅读，但后面的某些章节会假定你已经看过前面的内容。

读者对象

本书主要是为那些希望掌握Java语言和平台现代化知识的开发人员写的。如果你想跟上Java 7的步伐，就请阅读本书吧。

如果你想提升一下自己的技能，想搞清楚依赖注入、并发、测试驱动开发之类的主题，本书能为你打下良好的基础。

本书也是为已经认识到多语言编程趋势并想深入下去的开发人员准备的。具体来说，如果你想学习函数式编程，那么本书介绍Scala和Clojure的两章会很有帮助。

路线图

第一部分只有两章。第1章介绍了Java 7及其Coin项目，该项目包含很多小巧高效的特性。第2章全面介绍了新I/O API，包括对文件系统API的全面梳理，还介绍了新的异步I/O能力。

第二部分分四章介绍了Java 7的关键技术。第3章告诉你依赖注入技术的源流，接着展示了Java中的标准解决方案Guice 3。第4章阐述在Java中如何正确进行现代并发开发。因为硬件行业坚定地朝着多核处理器方向发展，这个话题再次成为焦点。第5章介绍了JVM的类文件和字节码，揭示了它们的秘密，让你明白Java的工作原理。第6章讲解Java应用程序调优的基础知识，并讨论垃圾收集器等内容。

第三部分介绍JVM上的多语言编程，由四章组成。多语言编程的内容从第7章开始，这里讲述了多语言编程背景知识，以及使用另一种语言的恰当时机。第8章介绍了Groovy——Java动态编程的朋友。Groovy突显了语法相似的动态语言如何大幅提升Java开发人员的生产率。第9章将你带入函数式/OO混合的Scala世界。Scala是一种强大精炼的语言。第10章是为Lisp粉丝们准备的。Clojure被广泛誉为“使用得当的Lisp”，它全面展示了JVM上函数式语言的力量。

第四部分以前三部分的内容为基础，讨论多语言编程技术在几个编程领域涉及的问题。第11章谈到了测试驱动开发，还提供了一个围绕处理模拟对象的方法，给出了一些实战建议。第12章介绍了两种得到广泛应用的工具，用于构建流程中的Maven 3和用于持续集成的Jenkins/Hudson。第13章涵盖了与快速Web开发相关的主题，解释了Java在这一领域的传统缺陷，并提供了一些原型化的新技术（Grails和Compojure）。第14章是对全书的总结和对未来的展望，其中包括Java 8可能支持的新功能。

代码约定及下载

首先需要下载和安装的是Java 7。只要找到适合你的OS的发布包，按照下载和安装说明来做就行了。Oracle网站上Java SE部分有安装包的在线下载和说明，网址是www.oracle.com/technetwork/ java/javase/downloads/index.html。

另请参见附录A中的安装说明以及源码运行的指南。

书中所有源码都是等宽字体，以区别于周围的文字。很多代码清单中都有注解，指出其中的关键概念，有时候文中使用带圆圈的数字，对代码进行更详细的注解。我们尽量按照版心的宽度设置代码格式，也在必要时换行，并谨慎使用缩进。但有时候代码行太长，因此会有续行标记。

书中所有示例的源码都可在www.manning.com/TheWell-GroundedJavaDeveloper找到。[①]书中

① 本书源码也可在图灵社区http://www.ituring.com.cn/book/1027下载。——编者注

自始至终都有示例代码。比较长的代码清单有清楚的清单标题，短一些的代码清单就在文字的行与行之间。

软件需求

如今，Java 7几乎可运行在任何现代平台上。只要你使用以下某个操作系统，就可以运行源码示例：

- MS Windows XP及以上版本；
- 较新版的*nix；
- Mac OS X 10.6及以上版本。

多数人都想在IDE中试验代码示例。以下这些主流IDE都对Java 7和最新版的Groovy、Scala、Clojure提供良好支持：

- Eclipse 3.7.1及以上版本；
- NetBeans 7.0.1及以上版本；
- IntelliJ 10.5.2 及以上版本。

我们使用了NetBeans 7.1和Eclipse 3.7.1来创建和运行代码示例。

作者在线

购买本书英文版的读者可以免费访问由Manning出版社运营的专用Web论坛，并在论坛中对该书进行评论、提出技术问题、从作者和其他用户那里得到帮助。要访问并订阅该论坛，请访问www.manning.com/TheWell-GroundedJavaDeveloper，这个页面介绍了注册后如何访问论坛、可以得到什么帮助以及论坛上的行为规则。

Manning向读者承诺为读者之间、读者与作者之间提供一个可以对话的场所，但不会强制作者参与，作者在论坛上的贡献都是自愿而且不收费的。为使作者感兴趣，提高其参与度，我们建议读者向作者提一些具有挑战性的问题。

只要本书英文版仍然在售，读者就可以从出版社的网站上访问作者在线论坛和之前讨论话题的归档。

关于作者

Ben Evans是伦敦Java用户组的发起人、协助定义Java生态系统标准的Java社区过程执行委员会成员。他在技术圈已经度过了很多年“有趣的时光”，现为一家面向金融业的Java技术公司的CEO。Ben经常在公开场合发表关于Java平台、性能和并发的演讲。

Martijn Verburg（是jClarity的CTO）作为一名技术专家和众多初创企业的OSS导师，拥有十多年的经验。他也是伦敦Java用户组的领导者，带领全球的Java用户组成员为JSR（采用JSR计划）和OpenJDK（采用OpenJDK计划）作出了贡献。作为一位公认的技术团队优化专家，他经常应邀出席Java界的大型会议（JavaOne、Devoxx、OSCON、FOSDEM等）并发表演讲，人送雅号“开发魔头”，赞颂他敢于向行业现状挑战的精神。

关于封面图片

本书封面上的画像标题为“卖花人”，摘自19世纪法国出版的沙利文·马雷夏尔（Sylvain Maréchal）四卷本的地域服饰风俗纲要。其中每幅插图都是手工精心绘制并上色的。马雷夏尔这套书展示的丰富服饰，令我们强烈感受到200年前乡村与城镇的巨大文化差异。不同地域的人山水阻隔，言语不通。无论奔走于街巷，还是驻足于乡间，通过他们的服饰，一眼就能看出他们的生活场所、职业，以及生活境况。

时过境迁，书中描绘的那些区域性服饰差异到如今已经不复存在。即使是不同国家，都很难再看出人们着装的区别，再不必说城镇和乡村了。或许，我们今天多姿多彩的人生，正是从前那些文化差异的体现。只不过，如今的生活更加多元，而且技术环境下的生活节奏也更快了。

今时今日，计算机图书层出不穷，Manning就以马雷夏尔这套书中多样性的图片，来表达对IT行业日新月异的发明与创造的赞美。

目　录

第一部分　用 Java 7 做开发

第二部分　关键技术