HZ BOOKS
华章科技

全面阐释Java 7在语法、JVM、API类库等方面的所有重要新功能和新特性，可帮助开发者大幅度提升编码效率和代码质量

对JVM、源代码和字节代码操作、类加载器、对象生命周期、多线程、并发编程、泛型、安全等Java平台的核心技术进行深入解析，包含大量最佳实践

华章原创精品

深入理解 Java 7

核心技术与最佳实践

Understanding the Java 7
the Core Techniques and Best Practice

成富 著

机械工业出版社
China Machine Press

华章原创精品

深入理解 Java 7

核心技术与最佳实践

Understanding the Java 7
the Core Techniques and Best Practice

成富 著

机械工业出版社
China Machine Press

本书是学习 Java 7 新功能和新特性以及深入理解 Java 核心技术的最佳选择之一。经过近 6 年的等待，Java 迎来了它的又一个历史性的版本——Java 7。Java 7 在提高开发人员的生产效率、平台性能和模块方向上又迈进了一步，变得更加强大和灵活。本书不仅对 Java 7 的所有重要更新进行了全面的解读，而且还对 Java 平台的核心技术的底层实现进行了深入探讨，包含大量最佳实践。

全书的主要内容可分为三大部分：第一部分是 1 ～ 6 章，全面阐释 Java 7 在语法、JVM、类库和 API 等方面的所有重要新功能和新特性，掌握这部分内容有助于大幅度提升编码效率和提高代码质量；第二部分是 7 ～ 13 章，对 JVM、Java 源代码和字节代码操作、类加载器、对象生命周期、多线程、并发编程、泛型、安全等 Java 平台的核心技术进行了深入解析，掌握这部分内容有助于深入理解 Java 的底层原理；第三部分为第 14 章，是对 Java 8 的展望，简要介绍了 Java 8 中将要增加的新特性。

图书在版编目（CIP）数据

深入理解 Java 7：核心技术与最佳实践 / 成富著 . —北京：机械工业出版社，2012.5

ISBN 978-7-111-38039-9

I. 深…　II. 成…　III. JAVA 语言 – 程序设计　IV. TP312

中国版本图书馆 CIP 数据核字（2012）第 068801 号

机械工业出版社（北京市西城区百万庄大街 22 号　邮政编码　100037）
责任编辑：朱秀英
北京京师印务有限公司印刷
2012 年 5 月第 1 版第 1 次印刷
186mm×240mm • 29.25 印张
标准书号：ISBN 978-7-111-38039-9
定价：79.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88378991；88361066
购书热线：（010）68326294；88379649；68995259
投稿热线：（010）88379604
读者信箱：hzjsj@hzbook.com

前　言

为什么要写这本书

我最早开始接触 Java 语言是在大学的时候。当时除了用 Java 开发一些小程序之外，就是用 Struts 框架开发 Web 应用。在后来的实习和工作中，我对 Java 的使用和理解更加深入，逐渐涉及 Java 相关的各种不同技术。使用 Java 语言的一个深刻体会是：Java 语言虽然上手容易，但是要真正掌握并不容易。

Java 语言对开发人员屏蔽了一些与底层实现相关的细节，但是仍然有很多内容对开发人员来说是很复杂的，这些内容恰好是容易出现错误的地方。我在工作中就经常遇到与类加载器和垃圾回收相关的问题。在解决这些问题的过程中，我积累了一些经验，遇到类似的问题可以很快地找到问题的根源。同时，在解决这些实际问题的过程中，我意识到虽然可以解决某些具体的问题，但是并没有真正理解这些问题的解决办法背后所蕴含的基本原理，仍然还只是处于一个“知其然，不知其所以然”的状态。于是我开始阅读 Java 相关的基础资料，包括 Java 语言规范、Java 虚拟机规范、Java 类库的源代码和其他在线资料等。在阅读的基础上，编写小程序进行测试和试验。通过阅读和实践，我对 Java 平台中的一些基本概念有了更加深入的理解。从 2010 年开始，我对积累的相关知识进行了整理，在 InfoQ 中文站的“Java 深度历险”专栏上发表出来，受到了一定的关注。

2011 年 7 月，在时隔数年之后，Java 的一个重大版本 Java SE 7 发布了。在这个新的版本中，Java 平台增加了很多新的特性。在 Java 虚拟机方面，invokedynamic 指令的加入使虚拟机上的动态语言的性能得到很大的提升。这使得开发人员可以享受到动态语言带来的在提高生产效率方面的好处。在 Java 语言方面，语言本身的进一步简化，使开

发人员编写代码的效率更高。在 Java 类库方面，新的 IO 库和同步实用工具类为开发人员提供了更多实用的功能。从另外一个角度来说，Java SE 7 是 Oracle 公司收购 Sun 公司之后发布的第一个 Java 版本，从侧面反映出了 Oracle 公司对 Java 社区的领导力，可以继续推动 Java 平台向前发展。这可以打消企业和社区对于 Oracle 公司领导力的顾虑。Java SE 7 的发布也证明了基于 JCP 和 OpenJDK 的社区驱动模式可以很好地推动 Java 向前发展。

随着新版本的发布，肯定会有越来越多的开发人员想尝试使用 Java SE 7 中的新特性，毕竟开发者社区对这个新版本期待了太长的时间。在 Java 程序中使用这些新特性，可以提高代码质量，提升工作效率。Java 平台的每个版本都致力于提高 Java 程序的运行性能。随着新版本的发布，企业都应该考虑把 Java 程序的运行平台升级到最新的 Java SE 7，这样可以享受到性能提升所带来的好处。对于新的 Java 程序开发，推荐使用 Java SE 7 作为标准的运行平台。本书将 Java SE 7 中的新特性介绍和对 Java 平台的深入探讨结合起来，让读者既可以了解最新版本的 Java 平台的新特性，又可以对 Java 平台的底层细节有更加深入的理解。

读者对象及如何阅读本书

本书面向的主要读者是具备一定 Java 基础的开发人员和在校学生。本书中不涉及 Java 的基本语法，因此不适合 Java 初学者阅读。如果只对 Java SE 7 中的新特性感兴趣，可以阅读第 1 章到第 6 章；如果对 Java 中的特定主题感兴趣，可以根据目录有选择地阅读。另外，第 1 章到第 6 章虽然以 Java SE 7 的新特性介绍为主，但是其中也穿插了对相关内容的深入探讨。

本书可分为三大部分：

第一部分为 Java SE 7 新特性介绍，从第 1 章到第 6 章。这部分详细地介绍了 Java SE 7 中新增的重要特性。在对新特性的介绍中，也包含了对 Java 平台相关内容的详细介绍。

第二部分为 Java SE 7 的深入探讨，从第 7 章到第 13 章。这部分着重讲解了 Java 平台上的底层实现，并对一些重要的特性进行了深入探讨。这个部分所涉及的内容包括 Java 虚拟机、Java 源代码和字节代码操作、Java 类加载器、对象生命周期、多线程与并发编程实践、Java 泛型和 Java 安全。

第三部分为 Java SE 8 的内容展望，即第 14 章。这部分简要介绍了 Java SE 8 中将

要增加的新特性。

本书还通过两个附录对 OpenJDK（附录 A）和 Java 语言的历史（附录 B）进行了简要的介绍。

勘误和支持

由于作者的水平有限，编写时间仓促，书中难免会出现一些错误或者不准确的地方，恳请读者批评指正。如果您有更多的宝贵意见，欢迎发送邮件至邮箱 alexcheng1982@gmail.com，也可以通过微博（http://weibo.com/alexcheng1982）与我取得联系。期待能够得到您的真挚反馈。

本书官方微群：http://q.weibo.com/943166。

本书中的源代码请登录华章公司的网站（http://www.hzbook.com）本书页面进行下载。

致谢

感谢 InfoQ 中文站和 InfoQ 编辑张凯峰先生。这本书能够面世，得益于我在 InfoQ 中文站的“Java 深度历险”专栏上发表的文章。

感谢机械工业出版社华章公司的编辑杨福川和姜影的辛勤工作，使得这本书能够最终顺利完成。

感谢家人和朋友对我的支持与帮助！

Java 的挑战与展望

从 Java 语言出现到现在的 16 年间，在 Java 语言本身发展演化的同时，整个软件开发行业也在发生着巨大的变化。新的软件开发思想和程序设计语言层出不穷。虽然 Java 语言一直是最流行的程序设计语言之一，但它也面临着来自其他编程语言的冲击。这其中主要是互联网应用发展所带来的动态语言的影响。

Java 是静态强类型语言。这种特性使 Java 编译器在编译时就可以发现非常多的类型错误，而不会让这些错误在运行时才暴露出来。对于构建一个稳定而安全的应用来说，这是一个很大的优势，但是这种静态的类型检查也限制了开发人员编写代码时的创造性和灵活性。

Web 2.0 概念的出现和互联网应用的发展，为新语言的流行创造了契机。Ruby 语言凭借着杀手级应用 Ruby on Rails 一举蹿红，而 Google 的 Web 应用开发平台 Google App Engine 最初也只支持 Python 一种语言，甚至流行的 JavaScript 语言也借助于 node.js 和 Aptana Jaxer 等平台在服务器端开发中占据了一席之地。这些语言的共同特征是动态类型与灵活自由的语法。开发人员一旦掌握了这些语言，开发效率会非常高。在这一点上，Java 语言繁琐的语法就显得缺乏吸引力。Java 语言也受到来自同样运行在 Java 虚拟机上的其他语言的挑战。这些语言包括 Groovy、Scala、JRuby 和 Jython 等。任何语言，只要它生成的字节代码符合 Java 字节代码规范，就可以在 Java 虚拟机上运行。前面提到的这些 Java 虚拟机上的语言既具有简洁优雅的语法，又能充分利用已有的 Java 虚拟机资源，相对于 Java 语言本身来说，非常具有竞争力。

基于前面的这些现状，在社区中有人悲观地预言：Java 已死，COBOL 式的死亡。

COBOL 这门诞生于 20 世纪 50 年代末的编程语言，已经被诸多机构和个人论证为已经死亡的语言。实际上，COBOL 语言仍然在银行、金融和会计等商业应用领域占据着主导地位。只要这些应用存在，COBOL 语言就不会消亡。Java 语言也是如此。只要运行在 Java 平台上的应用还存在，Java 语言就能一直生存下去。事实上，现在仍然有许多公司和个人在向 Java 平台投资。这些投资既包括投入资金和人力来开发基于 Java 平台的应用，也包括投入时间来学习 Java 平台的相关技术。

当然，Java 平台也有不足之处，其中最明显的是整个 Java 平台的复杂性。最早在 JDK 1.0 发布的时候，只有几百个 Java 类，而现在的 Java 6 已经包括 Java SE、Java EE 和 Java ME 等多个版本，所包含的 Java 类多达数千个。对于普通开发者来说，完全理解和熟悉如此庞大的类库的难度非常大。在日常的开发过程中，经常可以看到开发者在重复实现某些功能，而这些功能在 Java 类库中已经存在，只是不被人知道而已。除了庞大的类库之外，Java 语言的语法本身也缺乏足够的灵活性，实现某些功能所需的代码量可能是其他语言的几倍。另外一个复杂性体现在 Web 应用开发方面。一个完整的 Java EE 应用程序要求程序员掌握和理解的概念太多，要使用的库也非常多。这点从市面上到处可见的以 Java Web 应用开发和 Struts、Spring 及 Hibernate 等框架为内容的图书上就可以看出来。虽然新出现的 Grails 和 Play 框架等都试图降低这个复杂度，但是这些新的框架的流行仍然需要足够长的时间。

对于 Java 语言的未来，我们有理由相信 Java 平台会一直发展下去。其中很重要的依据是 Java 平台的开放性。依托 JCP 和 OpenJDK 项目，Java 平台不仅在语言规范这个层次上有健康的开放管理流程，也有与之对应的参考实现。Java 语言有着人数众多的开发者社区，每年有非常多新的开发者学习和使用 Java。大量的开发者使用 Java 语言开发各种不同类型的应用。在社区中可以看到很多提供不同功能的类库和框架。Java 虚拟机已经被安装到数以十亿计的不同类型的设备上，包括服务器、个人计算机、移动设备和智能卡等。依托庞大的社区和数量众多的运行平台，Java 语言的发展前景是非常乐观的。

对于 Java 平台来说，未来的发展将侧重于以下几个重要的方面。**第一个方面是提高开发人员的生产效率**。由于 Java 语言的静态强类型特性，使用 Java 语言编写的程序代码一般比较繁琐，包含了过多不必要的语法元素，这在一定程度上降低了开发人员的生产效率。大量的时间被浪费在语言本身上，而不是真正需要的业务逻辑上。从另外一个角度来说，Java 语言的这种严谨性，对于复杂应用的团队开发是大有好处的，有利于构建健壮的应用。Java 语言需要在这两者之间达到一个平衡。Java 语言的一个发展趋势是在可能的范围内降低语言本身的语法复杂度。从 J2SE 5.0 中增强的 for 循环，到 Java

SE 7 中的 try-with-resources 语句和 <> 操作符，再到 Java SE 8 中引入的 lambda 表达式，Java 正在不断地简化自身的语法。

第二个方面是提高性能。Java 平台的性能一直为开发人员所诟病，这主要是因为 Java 虚拟机这个中间层次的存在。随着硬件技术的发展，越来越多的硬件平台采用了多核 CPU 和多 CPU 的架构。应用程序应该充分利用这些资源来提高程序的运行性能。Java 平台需要帮助开发人员更好地实现这个目标。Java SE 7 中的 fork/join 框架是一个高效的任务执行框架。Java SE 8 对集合类框架和相关 API 做了增强，以支持对批量数据进行自动的并行处理。

第三个方面是模块化。一直以来，Java 平台所包含的各种功能不同的类库是一个统一的整体。在一个程序的运行过程中，很多类库其实是不需要的。比如对于一个服务器端运行的程序来说，Swing 用户界面组件库通常是不需要的。模块化的含义是把 Java 平台提供的类库划分成不同的相互依赖的模块，程序可以根据需要选择运行时所依赖的模块，只有被选择的模块才会在运行时被加载。模块化的实现不仅可以应用到 Java 平台本身，也可以应用到 Java 应用程序的开发中，OpenJDK 中的 Jigsaw 项目提供了这种模块化的支持。

目　录