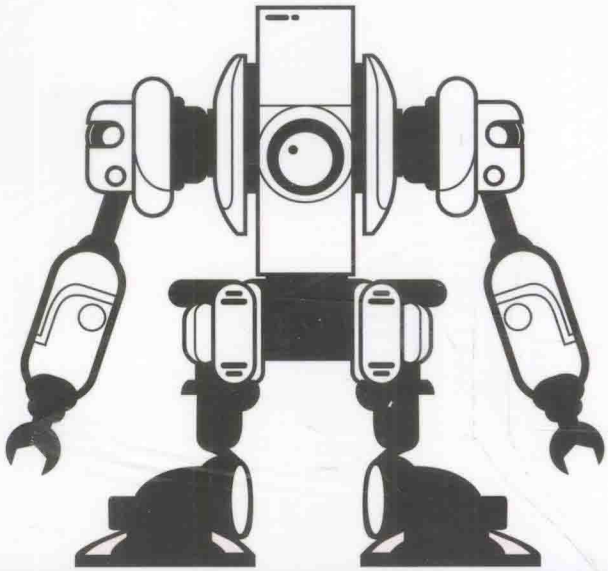
知名技术专家 **蔡学镛** 力荐

本书以精练的语句结合源码剖析的方式诠释了JVM的许多关键原理
阅读本书，你将有知其然并知其所以然的淋漓畅快感

Broadview®
www.broadview.com.cn



Java虚拟机精讲

如果你对JVM感兴趣，并且从未接触过JVM，
那么本书将会是你探索JVM世界的入门必备工具

高翔龙 编著

中国工信出版集团

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

Java虚拟机精讲

高翔龙 编著

電子工業出版社
Publishing House of Electronics Industry

内 容 简 介

HotSpot VM 是目前市面上高性能 JVM 的代表作之一，它采用解释器+JIT 编译器的混合执行引擎，使得 Java 程序的执行性能从此有了质的飞跃。本书以极其精练的语句诠释了 HotSpot VM 的方方面面，比如：字节码的编译原理、字节码的内部组成结构、通过源码的方式剖析 HotSpot VM 的启动过程和初始化过程、Java 虚拟机的运行时内存、垃圾收集算法、垃圾收集器（重点讲解了 Serial 收集器、ParNew 收集器、Parallel 收集器、CMS（Concurrent-Mark-Sweep）收集器和 G1（Garbage-First）收集器）、类加载机制，以及 HotSpot VM 基于栈的架构模型和执行引擎（解释器的工作流程、JIT 编译器的工作流程、分层编译策略、热点探测功能）等技术。

如果你对 JVM 感兴趣，并且从未接触过 JVM，那么本书将会是你探索 JVM 世界的必备入门工具。

图书在版编目（CIP）数据

Java 虚拟机精讲 / 高翔龙编著. —北京：电子工业出版社，2015.5
ISBN 978-7-121-25705-6

Ⅰ. ①J… Ⅱ. ①高… Ⅲ. ①JAVA 语言—程序设计 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字（2015）第 050785 号

责任编辑：孙学瑛
印　　刷：北京中新伟业印刷有限公司
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：17.5　字数：448 千字
版　　次：2015 年 5 月第 1 版
印　　次：2015 年 5 月第 1 次印刷
定　　价：69.00 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。

前言

大部分 Java 开发人员，除会在项目中使用到与 Java 平台相关的各种高精尖技术，对于 Java 技术的核心 Java 虚拟机了解甚少。这其中最主要的原因或许是在实际的开发过程中，开发人员根本没有机会或是没有必要与 Java 虚拟机等底层技术打交道，更多的只是简简单单地将 Java 虚拟机作为载体，让程序能够顺利运行其上即可。

笔者在面试的过程中，经常会对面试者询问一些与 Java 虚拟机相关的技术问题，但大部分开发人员对于笔者所提及的问题，几乎都有一个统一的答案，那就是不知道、不清楚。有些刚从校门出来的应届毕业生，甚至还包括一些有一定工作经验的开发人员，打心眼儿里觉得 Struts、Spring 和 Hibernate 等上层技术才是重点，基础技术并不重要，这其实是一种本末倒置的“病态”。这就好比金庸武侠小说《笑傲江湖》里的令狐冲，尽管独孤九剑看上去很酷炫，但由于本身内功修炼得不够，与敌人对抗时又能撑得了几个回合呢？

对于那些成熟的第三方开源产品，笔者始终只是把它们当作一种工具，用的时候是宝，不用的时候就丢弃，尽管有些残忍，但确实就是这么现实。以互联网项目为例，对于高性能和稳定性的要求往往大于企业级项目的规范化和流程化，因此架构师每天都会不停地思考，应该如何让我的系统更快？如何避免系统出现瓶颈？如果换作你来做架构师并负责解决这些问题，假如你对 Java 虚拟机一无所知，那么可想而知，你又有什么能力做到系统的性能调优？毕竟性能调优不仅仅只是单纯地从应用代码结构上进行调整，也不是纯粹地依靠物理堆机就能够解决的。

创作此书的目的

尽管并不是所有开发人员都能够在实际的项目开发过程中用到与 Java 虚拟机相关的优化技术，但这并不能够成为你不去了解 Java 虚拟机的理由。只要你从事的是与 Java 开发相关的

岗位，那么对 Java 虚拟机实现机制的了解就是你迟早必须攀爬的一座高山。如果你害怕，那么你将永远也无法屹立在山顶遥望最美的日出。

其实这几年国内也不乏一些比较优秀的技术作者创作了一些知名度比较高的有关 Java 虚拟机的作品。但是这些作品的创作初衷完全不同，一些完全是以理论为重心，而另外一些则完全是从底层源码实战出发为读者诠释 Java 虚拟机的实现细节。尽管这些作品都非常优秀，但笔者认为还不够简单，毕竟所面向的读者更多的是对 Java 虚拟机有一定程度了解的开发人员，对于从未接触过 Java 虚拟机的读者，或许会感觉到有些手足无措的挫败感。因此笔者创作此书的目的，在更大程度上是以更为精练的语句引读者入门 Java 虚拟机的世界。换句话说，你完全可以将本书看作一个跳板，当你熟知本书的内容后，如果有一种饥渴难耐的感觉，那么恭喜你，笔者建议你阅读难度更大的书籍，同时本书的目的也就达到了。

本书所面向的读者

本书适用于任何对 Java 虚拟机感兴趣的 Java 开发人员、系统架构师、Java 虚拟机爱好者。尤其是对于那些从未接触过 Java 虚拟机的 Java 开发人员，本书笔者竭尽所能用最精练和直接的语句诠释了有关 Java 虚拟机的方方面面，只要你熟练掌握了 Java 编程基础，那么阅读本书你将不会感觉到任何的吃力和枯燥乏味。

本书内容

本书的内容包括字节码的编译原理、字节码的内部组成结构、通过源码的方式剖析 HotSpot VM 的启动过程和初始化过程、Java 虚拟机的运行时内存、垃圾收集算法、垃圾收集器［重点讲解了 Serial 收集器、ParNew 收集器、Parallel 收集器、CMS（Concurrent-Mark-Sweep）收集器和 G1（Garbage-First）收集器］、类加载机制，以及 HotSpot VM 基于栈的架构模型和执行引擎（解释器的工作流程、JIT 编译器的工作流程、分层编译策略、热点探测功能）等技术。

从本书的第 1 章开始，笔者首先对 Java 的体系结构做了一个简单且全面的介绍，让大家深刻认识到了 Java 虚拟机在 Java 平台中所占的分量，然而这一章的重点则是在 OpenJDK 和 HotSpot VM 的编译实战任务上。当大家对 Java 虚拟机有了一个简单的了解后，本书的第 2 章则开始对字节码的编译原理进行了讲解，因为大家有必要了解 Java 语言规范与 JVM 规范之间的区别，以及 Java 代码究竟需要经历哪些步骤之后才能够被编译为一个有效的字节码文件。而本书的第 3 章则与上一章息息相关，当大家了解字节码的编译原理后，接下来笔者将会对字节码的内部组成结构进行深入讲解。本书的第 4～6 章涉及 HotSpot VM 的部分源码实现，从 Launcher 启动 HotSpot VM 开始，到 HotSpot VM 的初始化过程都是每一个 Java 开发人员必须

掌握和了解的。对于大部分 Java 开发人员而言，对 Java 虚拟机最感兴趣的内容莫过于内存管理和垃圾收集，本书用了大量的篇幅来对这些技术进行讲解，并且对 Java7 新增的 G1 收集器也做了介绍。在本书的第 7 章中，笔者对类加载机制进行了讲解，帮助大家对类的初始化过程理解透彻。然而在本书的最后一章中，笔者对 HotSpot VM 的架构模型和执行引擎进行了深入讲解，毕竟执行引擎是 Java 虚拟机中最重要同时也是最核心的部分，运行时编译技术使得 Java 程序的运行性能从此有了质的飞越。

参考文献

笔者在本书的创作过程中，从下面所列的这些参考资料中获取了极大的帮助，大家同时也可以通过如下信息找到更多关于 Java 虚拟机方面的资料，毕竟单靠一本书就想了解 Java 虚拟机的所有技术细节几乎是不可能的，更是不现实的。

- 《Java 虚拟机规范 Java SE7 版》 [美] Tim Lindholm、Frank Yellin、Gilad Bracha、Alex Buckley 著；
- 《深入 Java 虚拟机 第 2 版》 [美] Bill Venners 著；
- 《Java 性能优化权威指南》 [美]Charlie Hunt、Binu John 著；
- 《深入理解 Java 虚拟机 第 2 版》周志明 著；
- 《HotSpot 实战》 陈涛 著；
- 《程序员》2014 年 3 月刊《中间语言和虚拟机漫谈》 徐宥 著；
- 《JVM 分享：Java Program in Action》 Rednaxelafx（莫枢） 著；
- 《HotSpot 内存管理白皮书》。

感谢

此书献给我这辈子最爱的姥爷。从我诞生那天起，您就将您的慈爱毫不吝啬地给予了我，感谢您和姥姥从小对我的陪伴和照顾，是你们让我拥有了愉快的童年，让我体会到了亲情的温暖。我记得小时候家里的鹦鹉、画眉鸟是您最爱的宠物，您会每天带着它们出去散步，而如今却成为了我心中永远的回忆。姥爷，2014 年 05 月 09 日早上 7 点 20 分，当我最后在您耳边说完悄悄话后，您走了，带走了我的思念！姥爷，我一定会成为您这辈子的骄傲！除此之外，我们家可爱的小娇娇同学，谢谢你的支持和鼓励才让我有创作此书的勇气和动力，谢谢你，我爱你！

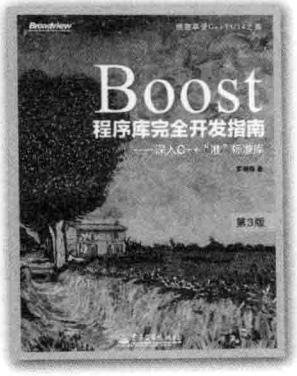
接下来要感谢的是 Rednaxelafx（莫枢）和蔡学镛，您二位在百忙之中抽空阅读了本书，提出了许多宝贵的意见，没有你们的支持或许本书至今也无法顺利出版。

最后还要感谢的是电子工业出版社博文视点的所有编辑们，谢谢你们无条件忍受着我一再的跳票，本书能够顺利出版离不开你们如此敬业的精神和一丝不苟的工作态度，由衷地谢谢你们。

高翔龙

2015 年 3 月

电子工业出版社计算机专业类畅销书



反侵权盗版声明

目录

第 1 章

Java体系结构

1.1 认识 Java

经历了多年的发展，Java 早已由一门单纯的计算机编程语言，演变为一套强大的技术体系平台。根据不同的技术规范，Java 设计者们将 Java 划分为 3 种结构独立但却又彼此依赖的技术体系分支，分别是 Java SE（标准版）、Java EE（企业版）和 Java ME（精简版）。在此大家需要注意，本书所提及的这 3 种技术体系分支，分别对应着不同的规范集合和组件。Java SE 活跃在桌面领域，主要包含了 Java API 组件。而 Java EE 则活跃在企业级领域，除了包含 Java API 组件外，还扩充有 Web 组件、事务组件、分布式组件、EJB 组件、消息组件等；综合这些技术，开发人员完全可以构建出一个具备高性能、结构严谨的企业级应用，并且 Java EE 也是用于构建 SOA[①]架构的首选平台。至于 Java ME 则活跃在嵌入式领域，之所以将其称之为精简版，那是因为该平台仅保留了 Java API 中的部分组件，以及适应设备的一些特有组件。

Java 在奠定了企业级领域的霸主地位后，目前正一步步朝着移动领域的方向大展拳脚，这不仅要感谢移动互联网的迅速崛起，还得多亏 Google 选择 Java 作为 Android 操作系统的应用层编程语言。就目前而言，Java 已经成为了全球开发人员使用最为广泛的一种编程语言。从随处可见的手持移动设备、嵌入式设备、个人电脑、高性能的集群服务器或大型机中，我们几乎随处都可以看见 Java 程序的身影。或许当你还在犹豫和怀疑 Java 能做什么的时候，Java 早已在企业级领域、互联网领域、移动领域、中间件领域，甚至是游戏领域都发展得

① SOA（Service-Oriented-Architecture，面向服务架构）作用于分布式的系统集成环境中，它将程序的内部功能通过定义良好的契约对外发布成体系结构中立的接口，以此满足不同系统之间的交互操作。

如火如荼。比如著名的开源 3D 游戏引擎 jME（j-Monkey-Engine）[2]就是完全采用 Java 语言编写的，该引擎可以算是目前 Java 平台上最流行，同样也是应用最广泛的 3D 游戏引擎。当然这所有的一切都离不开 Java 的运行支撑系统，那就是 Java 虚拟机，Java 与生俱来的通用性、安全性和高效性都建立在 Java 虚拟机之上。

从早期版本到每一个新版本的迭代，Java 都会不断完善自身缺陷，并进行语法增强，这无疑是带给开发人员最好的礼物。本书不仅会重点讲解与 Java 虚拟机相关的一些知识点，在本书的附录中笔者还为大家讲解了有关 Java7 在语法层面上的一些改变和扩充，让大家更全面地了解和掌握 Java 技术。

1.1.1 与生俱来的优点

面向对象的思想如今已经渗透到软件开发的各个领域，例如 OOA（Object Oriented Analysis，面向对象的分析）、OOD（Object Oriented Design，面向对象的设计），以及开发人员时常挂在嘴边的 OOP（Object Oriented Programming，面向对象的编程）。除了在急需注重性能与效率的应用场景下，开发人员大多数时候都是在使用面向对象等高级语言，比如 C#、C++、Ruby、PHP 等。这些高级语言无论是从设计原理或者是从实现细节上来看都是非常精妙的，那么与这些同样优秀的语言相比，Java 的优势主要体现在哪里呢？本书归纳了 Java 的 5 项重要优势：

- 体系结构中立；
- 安全性优越；
- 多线程；
- 分布式；
- 丰富的第三方开源组件。

Java 之所以能够实现“一次编译，处处运行”（Write Once，Run Anywhere），功不可没的首先当属字节码。和 C/C++等传统的编译性语言不同，Java 源代码的默认编译结果并非是可执行代码（本地机器指令），而是具有平台通用性的字节码。尽管不同平台 Java 虚拟机的内部实现机制不尽相同，但是它们共同解释出的字节码却是一样的，所以说字节码才是 Java 实现跨平台的关键要素，如图 1-1 所示。体系结构中立不仅使得 Java 天生具备跨平台的优势，同时还延伸了程序的安全性，因为 Java 程序始终只能够运行在 Java 虚拟机中，这与实际的物理宿主环境之间是相互“隔离”的，换句话说 Java 的安全模型可以禁止很多不安全的因素，有助于防止错误的发生，增强程序的可靠性。当然 Java 的部分语法限制也在某种意义上保障了程序的安全，比如废弃指针操作、自动内存管理、数组边界检查、类型

② jME（j-Monkey-Engine）官方地址：http://www.jmonkeyengine.org/。

转换检查、线程安全机制和物理环境访问限制等。

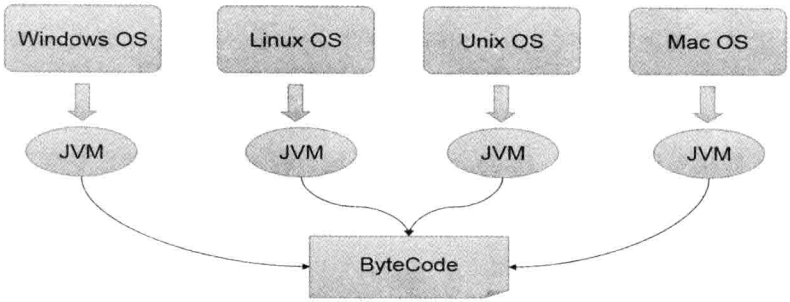


图 1-1　Java 跨平台特性

开发人员往往希望由自己编写的程序能够在性能上满足高效执行的要求，那么这就需要在程序中正确运用多线程技术去有效地并发执行操作任务。单线程是按照顺序结构进行串行执行的，我们暂且先不讨论程序的执行效率，一旦程序因执行某一个任务而发生异常，往往有可能导致程序终止而无法继续往下执行。当遇到这种情况的时候，就需要一种机制能够将这种任务从主线程中剥离出来，哪怕是发生异常也不会影响系统的整体运行。从另一个角度来说，我们可以使用多线程的并发机制将任务进行分散，而不是全部集中在主线程内，采用异步的方式去并发执行多项任务，这样的程序设计架构必然会有极高的执行效率。

由于单台服务器的处理能力很有限，甚至在某些情况下单台服务器的处理能力还存在性能瓶颈，因此在生产环境中架构师往往会考虑采用分布式架构的方式来部署应用。只有充分利用分布式环境中的每一个节点去协同处理任务，才能够换来较高的执行效率，并且还能有效降低单机负载以及提升稳定性和可用性。Java 与生俱来对分布式技术的支持就比较完善，比如 Java EE 规范中的 RMI（Java Remote Method Invocation，Java 远程方法调用）、JMS（Java Message Service，Java 消息服务）等技术。

框架这个名词相信对于绝大多数开发人员而言并不会感觉到陌生，它是对编码规范的一种抽象。在企业内部开发人员往往会为了提高生产效率和易于扩展从而选择使用框架，因为遵守框架定义的内部契约，不仅可以更高效地解决技术难题，同时还能够缩短项目的开发周期，总之如果能够在项目中正确选择和使用框架，那么所带来的好处将会是不言而喻的。

衍生在 Java 平台上的各类第三方开源框架几乎随时都保持着更新，可以毫不客气地说，在如今的整个领域模型中，随处可见成熟的第三方开源框架已经开始“泛滥”，当然本书仅以贯穿整个领域模型的 Spring 框架为例进行介绍。开发人员不仅可以选择使用 Spring 提供的全部功能，甚至还可以根据业务需求选择使用 Spring 的部分功能子集。如果想要使用

MVC[③]，Spring 提供支持 RESTful 风格的 Spring MVC，甚至允许与其他第三方 MVC 框架进行无缝集成；如果想要使用 Spring 的核心功能，那么 Spring 最新发布的 4.x 版本将会有全新体验；如果想对权限进行控制管理，Spring 提供 Spring Security；如果不想手动管理持久层事务，Spring 提供丰富的 ORM 集成策略，或者直接选用 Spring JDBC（Spring JDBC 仅仅只是针对传统的 JDBC 做了轻量级封装，熟悉 JDBC 的开发人员都能够迅速掌握并使用）作为持久层框架；如果想降低使用 Hadoop 进行海量数据处理所带来的复杂度和学习成本，Spring 提供 Spring Hadoop；甚至如果你是 Android 的开发人员，Spring 照样提供基于 Mobile 领域的各类规范和实现。

当然笔者并不是在为 Spring 打广告和做宣传，只是想告诉大家，Java 真正强大的地方是因为拥有全世界最多的技术拥护者和开源社区支持，他们无时无刻都保持着最充沛的体力与思维，一步一步地驱动着 Java 技术的走向，其实这才是 Java 最大的优势和财富。

1.1.2 语法结构和对象模型

Java 继承了 C 语言的语法结构，并改编了 C++语言的对象模型，所以注定了 Java 天生就适合书写优美的代码。我们都知道，类是最基本的封装单元，所有的操作都将发生在类中，那么定义一个类，其内部的组成结构又是怎样的呢？简单来说，属性和方法构成了一个简单的类，属性用于定义对象的各种“器官”，而方法则用于定义对象的一系列“行为”。虽然这样的描述很直观，却显得比较粗糙，如果将其细化分类后，大家或许会发现，Java 语法结构的设计是非常精妙的，同样又不失灵活性和简洁性。在类内部，开发人员可以定义许多元素特征，这些元素都统称为类成员，本书归纳了 Java 的一些基本类元素信息：

- □ 关键字；
- □ 标示符；
- □ 操作符（空白分隔符、普通分隔符）；
- □ 注解（@Annotation 类型、描述类型）；
- □ 数据类型（原始数据类型、引用类型）；
- □ 属性（常量、变量）；
- □ 运算符和表达式；
- □ 控制语句（流程控制语句、循环控制语句）；
- □ 异常处理；

③ MVC（Model-View-Controller，模型层-视图层-控制层）是一种编程模型，它将一个系统拆分为 3 大类，视图层仅只负责页面显示和数据显示工作，控制层则负责客户端的请求/响应和业务调用工作，而模型层则负责实际的业务操作。