HZ BOOKS

Sun 公司核心技术丛书

Mc Graw Hill Education

深入Java虚拟机（原书第2版）

Inside the Java Virtual Machine, Second Edition

(美) Bill Venners 著

曹晓钢 蒋靖 译

机械工业出版社
China Machine Press

Sun公司核心技术丛书

深入Java虚拟机

（原书第2版）

（美） Bill Venners 著

曹晓钢 蒋 靖 译

机械工业出版社
China Machine Press

本书作者曾因本书荣获专业技术杂志《Java Report》评选的优秀作者奖，细心的读者可以从网上找到许多对本书第1版的赞誉。

作者以易于理解的方式深入揭示了Java虚拟机的内部工作原理，深入理解这些内容，将对读者更快速地编写更高效的程序大有裨益！

本书共分20章，第1～4章解释了Java虚拟机的体系结构，包括Java栈、堆、方法区、执行引擎等；第5～20章深入描述了Java技术的内部细节，包括垃圾收集、Java安全模型、Java的连接模型和动态扩展机制、class文件、运算及流程控制等等，其中第6章和附录A～C完全可以作为class文件和指令集的参考手册。本书还附带光盘，光盘中包含用以辅助说明正文内容的交互式例示applet及示例源代码。

本书版权登记号：图字：01-2003-4928

图书在版编目（CIP）数据

深入Java虚拟机（原书第2版）/（美）文纳斯（Venners, B.）著；曹晓钢，蒋靖译. -北京：机械工业出版社，2003.9

（Sun公司核心技术丛书）

书名原文：Inside the Java Virtual Machine, Second Edition

ISBN 7-111-12805-2

Ⅰ.深… Ⅱ.①文… ②曹… ③蒋… Ⅲ.Java语言-程序设计 Ⅳ.TP312

中国版本图书馆CIP数据核字（2003）第068436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：刘立卿

北京牛山世兴印刷厂印刷 · 新华书店北京发行所发行

2003年9月第1版第1次印刷

787mm × 1092mm 1/16 · 30.25印张

印数：0 001- 4 000册

定价：58.00元（附光盘）

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线电话：（010）68326294

对本书第1版的赞誉

作者卓有成效地深入解释了Java虚拟机（JVM）的内部工作原理，对这个错综复杂的软件中的许多部分都给出了可能的实现，这是对Sun的官方规范的精彩补充。每一个概念都很清晰，一般都有例子作辅助说明。随书光盘中还包含了许多富有启发的示例，它们演示了虚拟机内部工作的情况。这本书得到虚拟机实现者的极高评价，相信任何有兴趣了解虚拟机核心部分的人都会获益匪浅。

——Antoine Trux，芬兰赫尔辛基诺基亚研究中心项目经理，《Java Report》杂志，1998年12月。《深入Java虚拟机》一书的作者因本书获《Java Report》杂志1998年优秀作者奖。

在我钻研本书的结构和内容之前，我很高兴提到Venners的书给我印象最深刻的一点：对细节的全心关注和对内容的精确协调。

从第5章到第20章都包含花很多心思编写的动态交互式applet，它们为每章的主题带来了活力。比如说垃圾收集这一章，不仅介绍了许多现代垃圾收集算法，还附带了一个“鱼堆”applet，让读者真正理解垃圾收集中的设计问题及可能的解决方案。

简单地说，Venners的书是卓越的，是一本我必须推荐的书。

——Laurence Vanhelsuwé，《Java World杂志》，1998年3月

感谢你写出这么优秀的书。我已经编写Java程序很多年了，这本书真的帮助我洞察了这门语言的内脏。再次为了美妙的阅读体验感谢你！

——Noah S. Friedland 博士

最近购买了你的书，它比JVM规范易读、易懂多了！我还喜欢你的applet，它们让事情变得简明易懂。

——Paul Bathen

《深入Java虚拟机》这本书，是我所有Java图书收藏中编写得最好和最有帮助的书之一。

——Louis Barton

我刚刚读完你的《深入Java虚拟机》，感谢你富有帮助的工作！

——Antoine Trux

一本关于Java虚拟机的详尽而系统的书。假如你准备开始编写自己的JVM，或者你对“在执行.class文件的时候究竟发生了什么事情”感兴趣的话，就必须拥有这本书。对所有读过Java虚拟机规范后还想寻找更多资料的人来说，这本书是受欢迎的、减轻痛苦的良药。

——Gopal Ananthraman

我真的在阅读你的书的时候感到愉悦。它有很多很好的内容，我觉得它们会使我成为更好的Java程序员。

——Joel Nylund，美国管理系统公司

我购买了一本《深入Java虚拟机》。虽然我只阅读了第7章和第8章，但我感到非常愉快，并且对中间的细节印象深刻。你回答了我所遇到的很多问题，包括“在调用ClassLoader.findSystemClass()的时候，在动态类装载器委派责任中，对于已装载的类，虚拟机会解释哪一个类装载器？”

我以前在Lotus开发公司工作的时候，我与别人合作为Prentice-Hall写过一本叫做《深入Lotus Add-in 工具包》的书。我们讨论的技术和Java很相似——一个平台中立的、拥有部分复杂性的语言（其字节码需要一个运行时虚拟机来执行）。

作为作者，我们的目标是在描述整个技术的时候保持精确性和幽默感。我们在技术上花费了大量劳动，对精确性和技术细节特别关注——如同开发者一样，我们希望文章是有用的、正确的；如同读者一样，我们精通英语的用法，因为大部分流行的技术文章都不敢恭维。

这些方面都是我对你的工作表示激赏的。当一个作者花时间来写完整的句子，采用通俗的语气，保持专业术语的一致性，并且提供真正有价值的内容，而不仅仅是重复公开的规范（通常还是不精确的），我向这样的作者致敬。

——David McCall

如果你真的希望揭开Java的面纱，这是最好的Java书。如果你真的希望了解JVM的输入输出，《深入Java虚拟机》是一本值得敬畏的著作。我被作为技术作家的Bill先生的能力打动了，对任何认真的Java开发者，如果想深入理解Java，我强烈推荐这本书。

——Rashid Jilani，发表于AMAZON.COM

一本伟大的书。

这是我到目前为止读过的最好的Java书。Bill是一个伟大的软件工程师，也是作家。如果你希望了解JVM的内幕，这本书是必不可少的。

——Michael Young，发表于AMAZON.COM

译 者 序

计算机艺术的魅力在于其严谨性和复杂性。无论是股票交易还是人机国际象棋大战，所有的程序运行所需要的底层机器指令都只是有限的若干条。从大型的UNIX机器到桌面个人计算机，无不基于那些设计精良而优美的指令集。但是这些指令集之间互不兼容，这就使得程序的移植变得非常困难，所需时间甚至超过了重新编写一遍的时间。于是，虚拟机的概念出现了。Java虚拟机（JVM）在多个平台上实现统一语言；.NET的虚拟机（目前）在单一平台上实现多种语言。但无论如何，它们都是抽象的计算机。尽管它们都有自己的指令集，自己的内存体系。但它们却往往比实际的硬件机器简单明了。分析这样的一个虚拟机，对提高读者对底层硬件和虚拟机平台的理解大有裨益。

Java之所以得以大行其道，除了它是一门面向对象、构造精美的语言之外，更重要的原因在于：它摆脱了具体机器的束缚，使跨越不同平台编写程序成为可能。Java语言丰富的开放式类库大量使用设计模式，成功地改变了很多程序员的编程思想和习惯。但是，诸如class文件是如何被调入内存执行的、类的静态方法和静态变量的初始化是按照什么步骤进行的、对象的垃圾收集是如何以及在什么时候发生的，这些具体问题却不是每个人都清楚的。很多人对此只是有一个模糊的印象，稍微深入就难以回答。假若能够有一本书简明扼要地解释这些JVM运行的细节，而又不需要去钻研艰深的JVM规范，那该有多好！读者现在看到的这本书就可以满足这些要求。

本书作者是一位JVM领域的顶尖高手，他写的这本书却一点儿都不难懂，任何人都可以从书中领会到Java的真正精髓——无需别人告诉你，从作者娓娓道来的分析中，就可以自己得出那些高手们才能理解的结论。

本书在Java专业书籍中的地位是无可替代的，简单地说，读者不容易找到第二本像本书这样如此细致地讨论Java虚拟机的书。假如你真的想深入了解Java的精髓，理解在别的书中花费很长时间讲解的“道理”或者“难点”，那么读读这本书吧！你可能比那些所谓的高手理解得更加深刻。

本书由曹晓钢和蒋靖翻译。其中蒋靖翻译了第6章、第10～19章以及附录，曹晓钢翻译了前言、第2～5章、第7～9章和第20章，第1章由两人合译。

由于时间仓促，加上译者水平有限，书中难免有翻译不妥之处，希望广大读者和同行批评指正。

译 者

2003.5

前　言

我写本书的主要目的是向Java程序员解释Java虚拟机——包括几个和虚拟机紧密相关的核心Java API。虽然Java虚拟机使用了许多有效技术——这些技术已在Java语言之前的其他语言中被尝试和证明过，但其中采用的很多技术还没有被普遍使用。因此很多程序员在开始使用Java编程的时候，都感觉是第一次接触这些技术。垃圾收集、多线程、异常处理、动态扩展，甚至使用虚拟机本身，这些对于很多程序员来说都是全新的。本书的目的是为了帮助程序员理解这些东西的工作方式，并在这个过程中帮助他们更加适应用Java编程。

编写本书的另外一个目的是为了试验改变文本的意义。网页有三个有趣的特性，这些特性使得它们和纸质文本有所区别：它们是动态的（可以随时间变化），它们是交互式的（特别是在上面嵌入Java applet后），还有，它们是相互链接的（可以很容易地在它们之间漫游）。除了传统的文本和图表，本书还包括几个Java applet（在随书光盘给出的迷你Web站点中），用它们作为交互式例示以补充文中所述概念。除此之外，我还在Internet上维护一个Web网站 artima.com，读者可以以此为起点找到与本书主题有关的更多、更新的参考资料。本书的构成包括文本、图表、交互式例示，还有网上链接，这样做的目的是为了方便读者深入阅读。

本书介绍

本书讲述了Java虚拟机——运行所有Java程序的抽象计算机，还讲了几种与虚拟机密切相关的核心Java API。本书通过分析讲解、可运行的示例、参考资料和applet（它作为文中所述概念的交互式例示），提供了Java技术的深入概览。

Java编程语言似乎将要成为继C和C++之后的下一门流行的主流商业软件开发语言，之所以这样的一个基本原因是，Java的体系结构能帮助程序员适应发展的硬件环境，Java具有在硬件环境中按照要求切换的特性，这都是由Java虚拟机提供的能力。

编程语言革命由硬件的发展所推动（当然还有更多推动力）。硬件在飞速发展，变得更加廉价且功能更加强大，软件变得越来越庞大、越来越复杂。从汇编语言到结构化语言的转变（比如C），以及到面向对象语言的转变（比如C++），在很大程度上是为了满足管理更高复杂度软件的需要——不断强大的硬件使得复杂度可能更高。

今天，获得更廉价、更快速、更强大硬件的势头仍在继续，软件复杂度不断增长的势头也在继续。在C和C++基础上，Java帮助程序员解决了一些复杂性，因为一些在C和C++中常见的固定类型的bug不再存在了。Java与生俱来的内存安全性——垃圾收集、取消了指针算法、在使用引用的时候进行运行时检查，避免了可能曾出现在Java程序中的大多数内存bug。Java的内存安全性使程序员生产效率更高，并在复杂度管理方面给他们提供了帮助。

除了持续增长的硬件能力之外，另外一个基础的硬件环境变化就是网络。网络把越来越多的计算机和设备连接起来，对软件提出了新的要求。随着网络的兴起，平台无关性和安全性也

变得更加重要了。

Java虚拟机负责Java程序设计语言的内存安全、平台无关和安全特性。虽然虚拟机在Java之前已经出现一段时间了，但是没有进入主流。然而，在今天不断变化的硬件环境现实面前，软件开发者需要一种使用虚拟机的编程语言。Sun用Java打开了这个市场的窗口。

也就是说，Java虚拟机为未来数年装备了正确的软件特性。本书会帮助读者理解Java虚拟机以及密切相关的几种Java API。有了这些知识，再通过自己的努力，就能使Java独一无二的体系结构发挥出更大的效能。

本书读者对象

本书主要是针对想了解Java技术的专业软件开发者和学生编写的。我假设读者对Java语言已经比较熟悉（但不需要精通），阅读本书会帮助读者深入理解Java编程知识。如果你是编写Java编译器或者编写Java虚拟机实现的少数精英之一，本书可以看作是对Java虚拟机规范的补充，书中对规范做出了解释。

如何使用本书

本书由五个部分组成：

1)对Java体系结构的介绍（第1～4章）。

2) Java内部细节的深入技术教程（第5～20章）。

3) class文件和指令集的索引参考（第6章和附录A～C)。

4) 交互式例示和示例源代码（在随书光盘中）。

5) 资源页（http://www.artima.com/insidejvm/resources/)。

对Java体系结构的介绍

第1～4章（本书的第一部分）给出了Java体系结构的总览，包括隐藏在Java体系结构设计背后的动机。这几章展示了Java虚拟机是如何与Java体系结构的其他组成部分（class文件、API和编程语言）相互关联的。如果想对Java技术有一个基础的了解，请阅读这些章节。下面是这部分的提要。

第1章“Java体系结构介绍”，在Java体系结构的概览和内部细节讨论上做了合理取舍。

第2章“平台无关”，讨论了平台无关的确切含义，Java体系结构是如何支持这个特性的，以及创建平台无关的Java程序的步骤。

第3章“安全”，描述了Java核心体系内置的安全模型，包含一个经精心制作的、可运行的例子，该例子示范了1.2版Java安全框架中的细粒访问控制的好处。

第4章“网络移动性”，讨论了网络移动软件的新范型。

Java内部技术教程

第5～20章(本书的第二部分)给出了Java虚拟机和相关核心Java API内部工作的深入技术描述，这些章节会帮助读者理解Java程序的实际运作情况。第二部分内容按照教程的方式组织，有很多示例。下面是这部分的提要。

第5章“Java虚拟机”，给出了对Java虚拟机内部工作的全面概览。

第6章“Java class文件”，是一份关于class文件格式的完整的教程和参考。如果你正在解析、生成或者比较关注Java class文件，那么这一章非看不可。

第7章“类型的生命周期”，讨论了类在Java虚拟机中的完整的生命周期，包含类被卸载的环境。

第8章“连接模型”，完整解释了Java的连接模型，包括使用forName（）和类装载器的例子，以便在运行时用新类型对Java应用程序进行动态扩展。

第9章“垃圾收集”，讨论了垃圾收集和终结（finalization），解释了什么是软、弱和影子引用，也提出了如何使用终结方法。

第10～19章是关于Java虚拟机指令集的教程。

第20章“线程同步”，解释了什么叫做监视器，以及如何使用它们编写线程安全的Java代码。

class文件和指令集参考

第6章除了作为Java class文件的教程之外，同时也是class文件格式的完整参考。同样，第10～20章构成了Java虚拟机指令集的教程，而附录A～C是指令集的完整参考。如果读者需要查阅有关指令的内容，请参见这些章节和附录。

交互式例示和示例源代码

本书的大多数章节在随书光盘上都找得到相关的材料——比如示例代码或者模拟applet。

随书光盘的applets目录中包含了一个叫做“Interactive Illustrations Web Site”的迷你Web网站，其中包含了15个Java applet，它们描绘了文中叙述的概念。这些交互式例示是本书的整体组成部分，其中11个applet模拟了Java虚拟机执行字节码，其他的演示了垃圾收集、二进制补码和IEEE 754标准的浮点数以及装载class文件的过程。这些applet可以在任何平台上、使用任何具有Java能力的浏览器浏览。这些模拟applet的源代码也包含在随书光盘上。

在“Interactive Illustrations Web Site”目录中的HTML，.java和.class文件根据版权声明允许读者把它们张贴到网络上（包括Internet）——只是必须遵守一些简单的规则。比如，必须完整地张贴整个站点（不能做任何修改），并且不能向浏览这个站点的人收费。版权声明的全文会在下面给出。

随书光盘中所有的示例源代码都包含源代码形式和编译过的形式(class文件)。如果对文本中的某个例子有兴趣或感到好奇，可以自己试验一下。

大部分示例代码都是用于解释目的的，除了帮助读者理解Java之外没有什么实际价值。不管怎样，读者可以从示范代码中随意拷贝、粘贴，用在自己的程序中，或者用二进制形式（比如Java class文件）发布。关于示例源代码的版权声明的全文会在下面给出。

Java虚拟机资源页面

为了让读者了解更多的信息，跟上时代的变化，我在artima.com维护一些页面，其中包含一些链接，指向与本书内容相关的阅读材料。这些链接页面的主页面是“Java虚拟机资源页面”，URL是 http://www.artima.com/insidejvm/resources/。

每章综述

第1章　对Java技术做了介绍，给出了Java体系结构的纵览，讨论了为什么Java很重要以及Java的优缺点。

第2章　展示了Java体系结构是如何让程序在任何平台上运行的，讨论了决定Java程序实际可移植性的要素，还考察了如何在可移植性及性能方面保持相应的平衡。

第3章 对内置于Java核心体系中的安全模型进行了深入概述，追踪了Java安全模型的演变过程——从1.0版本的沙箱到1.1版本的代码签名和验证，再到1.2版本的细粒度访问控制。

第4章 考察了Java带来的网络移动软件的新范型，并且展示了Java体系结构是如何让这项功能得以实现的。

第5章 给出了Java虚拟机内部体系的详细概述。随书光盘上与该章对应的applet叫做“Eternal Math”，它模拟了Java虚拟机执行一小段Java字节码的情况。

第6章 讲述了class文件的内容，包括常量池的结构和格式。这一章既可以作为Java class文件格式的教程，也可以作为class文件的完整参考。随书光盘上与该章对应的applet叫做“Getting Loaded”，它模拟了Java虚拟机装载一个Java class文件的过程。

第7章 对一个类型（类或者接口）的生命周期（从类型进入虚拟机到它最终退出）进行跟踪。该章还讨论了装载、连接和初始化的过程；还有如何创建对象示例，垃圾收集和终结；以及类型卸载。

第8章 深入考察了Java的连接模型，描述了类装载器的双亲委派模型、常量池解析、命名空间和装载约束。这一章还揭示了如何使用forName（）和类装载器，以便可以在运行时动态扩展Java应用程序。

第9章 讲述了垃圾收集的几种不同技术，解释了虚拟机中垃圾收集的工作原理——包含对火车算法以及对软引用、弱引用和影子引用的讨论。随书光盘上与该章对应的applet叫做“Heap of Fish”，它模拟了一个压缩的、“标记并清除”的垃圾收集堆。

第10章 讲述了用于操作数栈的Java虚拟机指令——把常数压入栈、进行通常的栈操作、在局部变量和栈之间互相传递数值等等。随书光盘上对应该章的applet是“Fibonacci Forever”，它模拟了Java虚拟机执行一个方法（该方法产生斐波那契序列）的过程。

第11章 讲述了在主要类型之间互相转换数值的指令。随书光盘上对应该章的applet是“Conversion Diversion”，它模拟了Java虚拟机执行一个方法（它进行类型转换）的过程。

第12章 讲述了Java虚拟机中的整数算法，解释了二进制补码算法，列出了用于整数计算的指令集。随书光盘上对应该章的有两个applet，它们以交互式例示形式描绘了该章的内容：其中一个applet叫做“Inner Int”，它可以让读者查看并操作二进制补码；另一个叫做“Prime Time”，它模拟了Java虚拟机执行一个方法（它生成质数）的过程。

第13章 讲述了Java虚拟机内部进行逐位运算、逻辑运算的指令，这些指令包括对整数进行小数点移位和Boolean(布尔)操作的操作码。随书光盘上对应该章的applet叫做“Logical Results”，它模拟了Java虚拟机执行一个方法（该方法使用一些逻辑操作码）的过程。

第14章 介绍了浮点数和Java虚拟机中执行浮点运算的指令，包括对strictfp关键字的讨论和出现在第2版Java虚拟机规范中的修改过后的浮点规则。随书光盘上有两个使用交互式例示来阐述该章内容的applet，一个名为“Inner Float”的applet能用来对组成浮点数的各个部分进行观察和操作，另一个名为“Circle of Squares”的applet能对Java虚拟机进行模拟，模拟它执行一个方法（它使用浮点操作码）的情况。

第15章 讲述了创建和操作对象和数组的Java虚拟机指令。随书光盘中为该章准备了一个名为“Three Dimensional Array”的applet，它模拟了Java虚拟机执行一个方法（它分配和初始化

三维数组）的过程。

第16章　介绍了控制Java虚拟机在同一个方法中进行条件或者无条件分支操作的指令。随书光盘中为该章准备了一个名为“Saying Tomato”的applet，它模拟了Java虚拟机执行一个方法的过程，该方法包含完成表跳转的字节码（Java源代码中switch语句编译后的版本）。

第17章　对字节码实现异常的方式、显式抛出异常的指令、异常表以及catch子句的工作方式进行了描述。随书光盘中为该章准备了一个名为“Play Ball!”的applet，它模拟了Java虚拟机执行一个方法（它抛出及捕获异常）的过程。

第18章　介绍finally子句在字节码中实现的方式，并举例介绍了相关指令。该章还对Java源代码中finally子句所展现的一些令人惊讶的特性进行了描述，并在字节码层对此特性做出了解释。随书光盘中为该章准备了一个名为“Hop Around”的applet，它模拟了Java虚拟机执行一个方法（它包含finally子句）的过程。

第19章　对Java虚拟机用来调用方法的四条指令以及使用这四条指令的环境进行了介绍。

第20章　讲述了监视器（Java用来支持同步的机制），并阐述了Java虚拟机使用它们的方式。该章还描述了指令集在数据的锁定和解锁方面对监视器的支持。

附录A　按照助记符的字母顺序列出操作码。对于每个操作码，该附录都给出了助记符、操作码字节值、指令格式（操作数，如果有的话）、指令执行前后堆栈快照，还描述了指令执行过程。附录A可作为指令集参考手册。

附录B　按照功能分组操作码。该附录中所使用的组织方式与指令在第10～20章中出现的顺序相对应。

附录C　按照操作码字节值的数字顺序排列操作码。对于每一个数值，该附录都给出了对应的助记符。

附录D　描述了最后一个applet，名为“Slices of Pi”，此applet模拟了Java虚拟机计算π的过程。

版权声明

下面是每个示例源代码文件（光盘中除了applets和jdk两个子目录之外的所有内容）都适用的版权声明：

LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR PARTICULAR PURPOSE, OR NON-INFRINGEMENT. BILL VENNERS SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY A LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR ITS DERIVATIVES.

Interactive Illustrations网站的HTML页（包括applet）和Java源文件（存储在光盘的applets目录中）都遵循下列版权声明：

All the web pages and Java applets delivered in the applets directory of the CD-ROM, consisting of ".html," ".gif," ".class," and ".java" files, are copyrighted © 1996, 1997 by Bill Venners, and all rights are reserved. This material may be copied and placed on any commercial or non-commercial web server on any network (including the internet) provided that the following guidelines are followed:

a. All the web pages and Java Applets (".html," ".gif," ".class," and ".java" files), including the source code, that are delivered in the applets directory of the CD-ROM that accompanies the book must be published together on the same web site.
b. All the web pages and Java Applets (".html," ".gif," ".class," and ".java" files) must be published "as is" and may not be altered in any way.
c. All use and access to this web site must be free, and no fees can be charged to view these materials, unless express written permission is obtained from Bill Venners.
d. The web pages and Java Applets may not be distributed on any media, other than a web server on a network, and may not accompany any book or publication.

BILL VENNERS MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR PARTICULAR PURPOSE, OR NON-INFRINGEMENT. BILL VENNERS SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY A LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR ITS DERIVATIVES.

一些术语

在本书中，我尝试使用与Java语言和Java虚拟机规范相一致的术语。考虑到有些读者可能不熟悉这套术语，因此先简单阐明一些术语。

首先，在本书中我尝试细分使用术语，"类型"和"类"。在Java的术语中，变量和表达式都有类型，对象和数组都有类。Java程序中的每一个变量和表达式都有编译时可以确认的类型——或者为基本类型（int、long、float、double等等），或者为引用类型（类、接口或者数组）。变量或表达式的类型决定了它所拥有的值的范围和种类、所支持的操作以及这些操作的含义。

在运行时，每一个对象和数组都有一个类。虽然对象是它的类和所有超类的实例，但是它只拥有一个类。一个对象的类可以是以下情形之一：

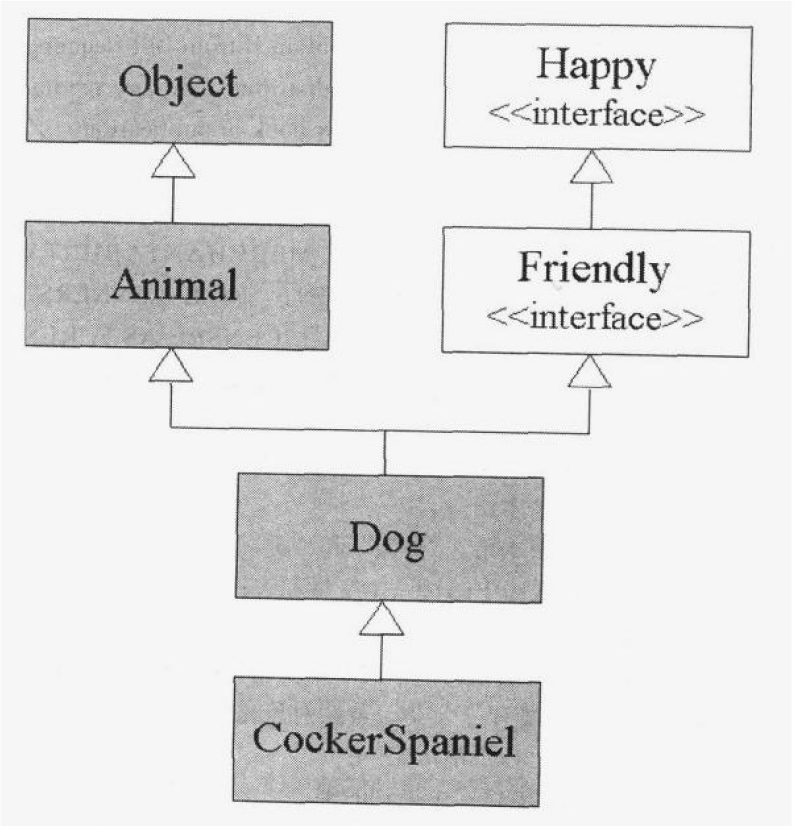
- 创建对象时所使用的创建实例表达式中提到的类。
- 使用newInstance（）方法创建对象时所使用的Class对象所代表的类。

- 使用clone（）方法创建对象时被克隆对象的类。
- 对以前序列化的对象解除序列化时，创建的对象具有与原对象相同的类。

数组类的名字是如[D或者[[[[I样的形式，这在Java语言中是不合法的（数组类的名字在第6章描述）。如果在运行时变量有一个非null的引用类型，那个变量就会指向一个对象，该对象的类和变量的类型必须是兼容的。

更复杂一点来说，在规范中包含对“类型”这个术语的另一个不同用法。因为变量可以声明自己的类型是某个类或者接口，所以类和接口可以定义程序使用的新类型（当然，创造新类型的能力是面向对象编程的基本概念之一）。在全书中，我只针对类使用“类”这个术语（而不是包含类和接口二者）。同样，我只针对接口使用“接口”这个术语。当我想同时表达二者的时候，我有时候说“类和接口”，但更经常使用“类型”。比如说，当我说“当类装载器装载了一个新的类型……”的时候，我的意思是“当类装载器装载了一个新的类或者接口”。在这句话中，类型不是指编译时一个变量声明的类型，而是指每个类和接口定义都代表的新的类型。

另一些我想事先澄清的术语是，在Java规范中描述两个类型（类或者接口）之间在继承图中的关系时所用的术语。考虑在下图中展示的继承图，在这个图中，类CockerSpaniel扩展自类Dog，Dog又是扩展自类Animal，而后者从类Object扩展而来。除此之外，接口Friendly扩展自接口Happy，类Dog实现了接口Friendly。



继承图

在Java术语中，在继承图中位于更高位置的类叫做“超类”，在较低位置的叫做“子类”。在上图中，Dog的超类是Animal和Object，Dog是Animal和Object二者的子类。在继承图上直接和类相连、紧邻上方的超类叫做“直接超类”。同理，直接连接在下方的叫做“直接子类”。例如，Animal是Dog的直接超类，CockerSpaniel是Dog的直接子类。

术语“子”和“超”的概念也可应用于接口，例如，接口Happy和Friendly是Dog和CockerSpaniel的超接口，接口Friendly是Dog的直接超接口及Happy的直接子接口。使用“子”和“超”术语概念的最后一种情况是表示“类”和“接口”的组合概念时，即“类型”概念中，在上图中，Friendly、Dog和CockerSpaniel是Happy的“子类型”，而Object、Animal、Happy、Friendly和Dog都是CockerSpanied的“超类型”。

Java版本和规范版本

本书的文本针对Java 2 SDK 版本1.2和第2版Java虚拟机规范。虽然本书包含的内容在版本1.0和1.1之间变动不多，但是在版本1.1和1.2之间有很大变化。第2版Java虚拟机规范要比第1版规范澄清了很多问题，也做了一些修订。

JDK版本1.0.2引进的一个变化是invokespecial指令的语义，这在第19章和附录A中描述。版本1.1中在class文件格式中增加了两个属性，用来支持内部类，还增加了一个属性用来支持@deprecated javadoc标签，它们都在第6章描述。版本1.1中还扩展了ClassLoader类的API，第8章演示了这个API的1.1版本。

本书第1版中讲述的Java API 1.1版本，和本书第2版中讲述的1.2版本在某些方面发生了很大变动。对本书影响最大的变动可能要算1.2版本的安全模型了。关于1.2版本的安全模型组件——基本沙箱、代码签名和验证、策略和策略文件、权限、代码源、保护域以及访问控制器的栈检查算法，这些都在第3章详细讲述。在Java语言1.2版中加入的strictfp关键字以及在class文件格式中加入的相应的访问标志，这些在第6章解释。1.2版本中引入的类装载器的双亲委派模型，还有1.2版本中引入的Java.lang.Class类和java.lang.ClassLoader类的几个新方法，在第8章讲述。软、弱和影子引用是在1.2版本的Java API中加入到java.lang.ref包中的，它们在第9章讲述。

除了对Java版本1.2中新加入的几种API进行介绍外，本书还解释了第2版Java虚拟机规范对原规范所做的很多澄清和修正。比如，第2版Java虚拟机规范中记录了新的装载约束，用于在存在多个类装载器时保证类型安全连接，这些装载约束在第8章进行了描述，并且有代码示例。第2版Java虚拟机规范中指出的对Java虚拟机浮点数规则的修订，在第14章中做了解释。本书的第2版还对class文件的版本号、方法调用、装载、连接和初始化类型方面的很多修正和澄清进行解释。

本书全书所使用的字节码示例都是用Sun公司的JDK版本1.1中附带的javac编译器生成的。记住，编译类可以有很多种方法。不同的编译器，或者是同一个编译器的不同版本都可能产生不同的结果。

本书中使用的模拟applet（交互式例示）的源代码遵守Java版本1.0。就像将在第2章中讨论的那样，Java平台无关的承诺所面对的现实之一就是，在使用一个特定的目标Java平台版本时，必须自行判断何时这个版本已经被广泛安装，何时才是值得使用这个平台的时机。虽然我1997年就有针对1.1版本Java虚拟机的模拟applet了，但在为本书第1版配CD-ROM时，我还是决定回

到1.0版本。因为在那时，不管是Netscape Communicator还是Microsoft Internet Explorer，都还没有完全支持版本1.1。因为这些applet并非是源代码示例，而是专用的软件产品，所以我觉得使用1.1版本发布它们没什么意义。因此，这些applet在支持版本1.0、1.1及1.2的浏览器中都可以运行，很可能在将来的很多版本上也可以运行。

请求指正

如果读者对本书有改进建议，请访问http://www.artima.com/insidejvm/feedback.html，该页面会指导读者如何提交评注意见。

目　录