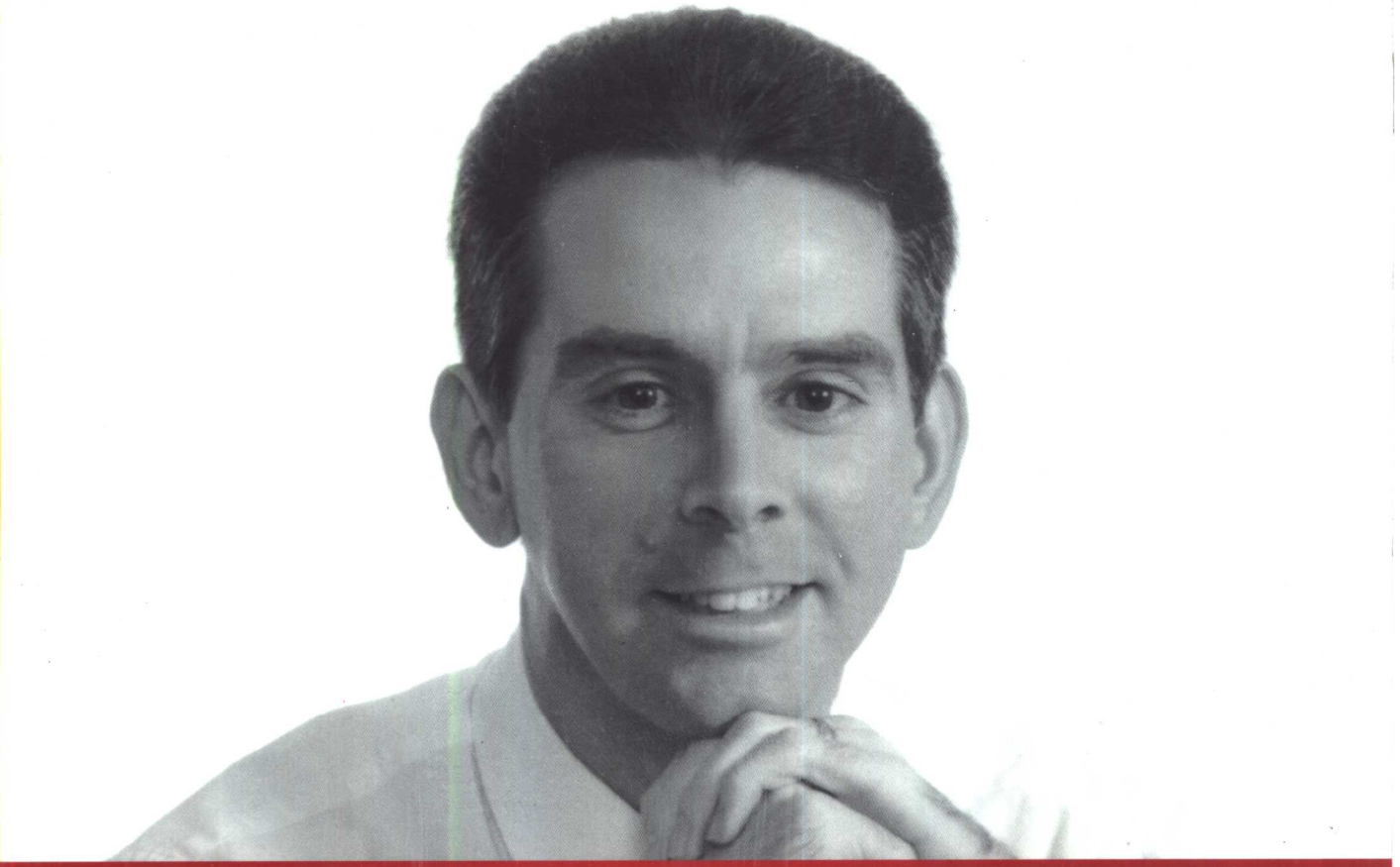
Professional Java Programming



Java
专业编程指南

〔美〕 Brett Spell 著

邱仲潘 等译

wrox 乐思
PROGRAMMER TO PROGRAMMER™

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

Professional Java Programming

Java专业编程指南

〔美〕 Brett Spell 著

邱仲潘 等译

電子工業出版社

Publishing House of Electronics Industry

北京 · BEIJING

内 容 提 要

本书是Java编程的实用指南，通过大量实例，深入浅出地介绍了如何使用Java语言和平台建立强大的应用程序，以及充分利用Java的跨平台性和易用性。首先介绍了编写高质量应用程序的各注意事项和用户界面组件；接着讲述了如何实现标准用户界面功能，如何处理分布式应用程序中的关键问题；最后介绍了有效的编码及完善它的要点。

图书在版编目（CIP）数据

Java专业编程指南/（美）斯泊尔（Spell，B.）著；邱仲潘等译. - 北京：电子工业出版社，2001. 10
书名原文：Professional Java Programming
ISBN 7-5053-7062-6

Ⅰ. J… Ⅱ. ①斯… ②邱… Ⅲ. JAVA语言 - 程序设计 Ⅳ. TP312

中国版本图书馆CIP数据核字（2001）第071555号

书　　名：**Java专业编程指南**
著　　者：〔美〕Brett Spell
译　　者：邱仲潘　等
责任编辑：杨　荟
印 刷 者：北京天竺颖华印刷厂
装 订 者：三河金马印装有限公司
出版发行：电子工业出版社　URL:http://www.phei.com.cn
　　北京市海淀区万寿路173信箱　邮编：100036　电话：68279077
　　北京市海淀区翠微东里甲2号　邮编：100036　电话：68252397
经　　销：各地新华书店
开　　本：787×1092 1/16　印张：61.5　字数：1570 千字
版　　次：2001年10月第1版　2001年10月第1次印刷
书　　号：ISBN 7-5053-7062-6
　　TP·4047
定　　价：98.00元

版权贸易合同登记号　图字：01-2001-2141

作 者 简 介

Brett Spell是路易安娜州Westlake市人，毕业于路易安娜州技术大学，从1996年开始从事Java编程，是Sun认证Java编程人员、开发人员和结构师。Brett住在德克萨斯州Plano市，为Frito-Lay国际总部的Pepsico业务方案组织工作。

谨将此书献给我的妻子Shari，她帮助我完成了这本书。还要感谢我的姐姐Gwendolyn，她鼓励我坚持到底，感谢母亲Evelyn给了我写作的信心。

简　介

欢迎词

欢迎使用《Java专业编程指南》，本书介绍如何用Java语言和平台建立精彩的应用程序，以及如何利用Java的跨平台性质和易开发性。

初看起来，要编写高级Java应用程序是很难的，需要生成功能强大、设计良好的代码，充分利用Swing的图形用户界面功能，交换XML数据或利用Java功能分配计算和访问自然代码。但是，稍后你就会发现，充分利用这些Java特性并不是很难。

本书并不准备介绍Java专业编程所涉及的全部课题，而是介绍一些最常见的问题及其解决方案。其中有些课题本身就可能独立成书，如面向对象设计、Swing组件、持久性、线程和分布式对象。本书并不想对每个领域的内容都面面俱到，而是主要介绍Java专业编程人员经常会遇到的一些问题。

本书读者

《Java专业编程指南》一书的读者对象是对Java语言和平台有一定经验的用户，旨在提高编程技巧。随着Java越来越普及，采用Java语言编程已成为专业开发人员的选择，越来越多的人开始学习Java语言。这样就使仅对Java一知半解的人更难开展工作，本书可以帮你提高Java编程技巧。

也许你具有使用Java的丰富经验，但想更上一层楼。也许你想用一个简单的方案来解决一个问题，但手头有多个差别不大的方案，不知如何选择最佳方案。本书能帮你做出选择。

本书内容

本书全面地介绍Java语言高级技术。本书包括二十章，分为五个部分：

- **编写高质量应用程序**　前三章介绍开始编写代码之前需了解的问题：Java虚拟机（JVM）如何工作、面向对象类和方法设计技术、处理异常的最佳做法和如何处理线程。
- **用户界面组件**　第4章到第7章分别介绍事件处理、布局管理器和JTable与JTree。这部分将介绍Java事件模型的工作原理，如何生成定制事件类型；如何充分利用标准布局管理器和生成自己的布局管理器；如何优化应用程序中表格的运用；并深入介绍树的复杂细节。
- **实现标准用户界面功能**　第8章讨论应用程序中实现剪切与粘贴功能的问题，第9章

介绍拖放技术，第10章介绍如何在应用程序中利用Java的打印功能，第11章介绍如何生成自定义组件。

- **建立分布式应用程序** 第12章到第15章介绍如何处理分布式应用程序中的关键问题，包括用JDBC与XML存储和访问数据、持久性以及分布式对象技术。
- **有效编码** 本书最后五章介绍向用户提供应用程序的重要方面。第16章介绍应用程序的安全性要点，第17章介绍性能与内存管理问题，然后介绍如何提供应用程序帮助系统、代码文档和为国际市场准备应用程序的重要问题。第20章介绍Java自然接口。

对读者的要求

本书的大部分代码是在Java 2 Platform, Standard Edition SDK（JDK 1.3）上测试过的，但有些章节还需要其他软件：

- 关系型数据库系统和适当的JDBC驱动程序，我们用Microsoft Access 2000和Java SDK所带的JDBC-ODBC桥驱动程序
- Java API for XML Parsing（JAXP），见http://java.sun.com/xml/
- Xalan XSLT处理器，见http://xml.apache.org/xalan/
- JavaHelp 1.1，见http://www.java.sun.com/products/javahelp/
- C++编译器，用于第20章的JNI例子，我们用Microsoft Visual C++ 6.0

本书的源代码可以从http://www.wrox.com/下载。

目　录

第1章 Java内幕

据Sun公司的定义，Java是个“简单、强大、面向对象、平台独立、多线程、动态和通用的编程环境”。这个定义比较简单，使Java能够增长和扩展到面目全非的新境界。如今，只要有微处理器的地方，几乎就有Java，最大的企业和最小的设备中都有Java，桌面计算机和超大型计算机中都有Java。为了让Java支持这种千差万别的环境，Sun公司开发了大量API和不同版本，但都建立在一组共同的核心类型上。

尽管如此，要成为好的Java编程人员，一定要掌握一些基础知识。能够建立相当复杂的用户界面虽然很好，但如果代码臃肿、很耗内存和效率低下，则用户决不会满意。本书不是介绍Java开发人员可以选择的大量开发选项，而是介绍Java开发人员经常遇到的常见任务，集中介绍一些核心语言特性，如线程与内存管理，这些特性使专业高质量Java应用程序脱颖而出。

Java被广为采用和迅速普及的核心在于其平台独立性。Java中的一句名言是“一次编写，到处运行（WORA，即Write Once，Run Anywhere）”。这个特性是Java本身的运行方式决定的，特别是利用抽象执行环境，使Java代码与基础操作系统分开。本书其余部分介绍Java编程语言和API，但本章先准备介绍Java操作的内幕和Java虚拟机（Java Virtual Machine）。了解Java操作的内幕有助于编程人员更好地了解这个语言，成为更好的编程人员。

我们将介绍下列内容：

- Java平台的各种组件
- Java虚拟机如何实现其平台独立性
- 运行Java程序时发生的情形
- Java类文件内容
- 使用Java虚拟机所需的关键工具

下面首先介绍Java平台的各种组件。

Java体系结构

也许你以为Java只是开发应用程序的编程语言，用于编写源代码文件并编译成字节码。但是，Java编程语言只是Java的组件之一，而Java体系结构则提供Java的许多优点，例如其平台独立性。

完整的Java体系结构实际上包括四个组件：

- Java编程语言
- Java类文件格式
- Java应用程序编程接口（API，Application Programming Interfaces）
- Java虚拟机

因此，用Java开发时，首先要用Java编程语言编写代码，然后编译成Java类文件，然后在Java虚拟机中执行。

Java虚拟机与核心类的组合形成Java平台，也称为Java运行时环境（JRE，Java Runtime Environment），位于所用操作系统之上，如图1.1所示。

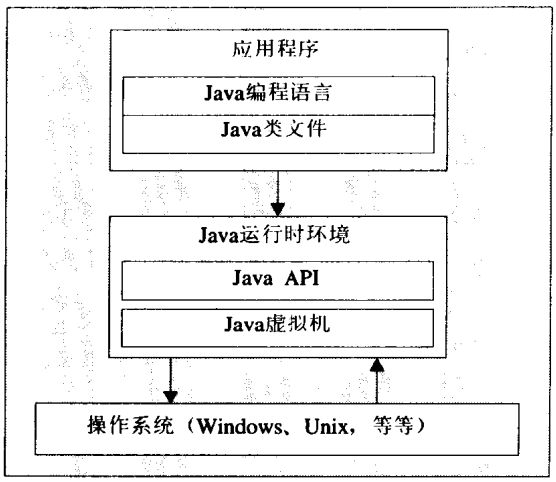


图1.1

核心类和支持文件（JRE）与将字节码转换成自然调用的解释器（Java虚拟机）通常是不同实体。要运行Java程序，则需要将JRE作为一个独立的操作来装入，以启动JVM。但是，在当前JVM版本中，JRE已经集成到JVM中，因此，启动JVM的同时也就装入了JRE。

Java应用程序编程接口（API，Application Programming Interface）是预先编好的代码，组成类似主题的包。Java应用程序编程接口分为三大平台：

- Java 2 Platform，Standard Edition（Java 2平台，标准版），简称J2SE。这个平台包含核心Java类和GUI类。
- Java 2 Platform，Enterprise Edition（Java 2平台，企业版），简称J2EE。这个包包含开发基于Web的应用程序的类与接口，包含servlet、JavaServer Pages和Enterprise JavaBeans类，等等。
- Java 2 Platform，Micro Edition（Java 2平台，微观版），简称J2ME。在这个包中，Java回到根，对传呼机、移动电话、汽车导航系统等产品提供优化的运行时环境。

Java虚拟机

介绍编写强大的Java应用程序的各个方面之前，先要花一些时间介绍实现这个功能的引擎。这个引擎称为Java虚拟机（Java Virtual Machine或JVM）。Java虚拟机是解释编译Java程序的抽象计算机器。

编译器通常是针对处理器和操作系统的，但对C或C++之类的其他编程语言，编译器则将源代码编译成执行程序，然后该执行程序就成为完全可以自给自足的运行程序在机器上运行。

这种方法的缺点是缺乏可移植性（portability）。在一个操作系统中编译的代码无法在另一个操作系统中运行。代码要在准备运行的每个不同的系统中重新编译。更糟的是，由于厂家特定的编译器特性，使得在某种处理器系列（如Intel x86、SPARC、Alpha）的某个操作系统中编译的代码可能无法运行在其他处理器的相同操作系统中。例如，在运行Windows NT的Alpha工作站上编译的C语言代码可能无法在运行NT的Intel PC机上运行。

人们开始对Internet编写应用程序时，这个问题变得更加尖锐。这些应用程序要面向各种平台、各种操作系统以及各种浏览器类型。要解决这个问题，只能开发一种平台独立语言。

20世纪90年代初，Sun微系统公司的开发人员准备开发在家电设备中使用的平台独立语言，由于还太超前，这个开发没有引起重视。随着Internet的出现，这些开发人员看到这个语言的巨大潜力，使Java应运而生。

Java语言可移植性的关键是，Java编译器的输出不是标准的可执行代码。Java编译器产生优化指令集，称为字节码（bytecode）程序。字节码是格式化字节序列，有关这部分内容，我们将在稍后更详细地进行介绍。字节码程序由Java运行的系统解释，也称为Java虚拟机。更妙的是，一个平台上产生的字节码程序可以在装有Java虚拟机的另一平台上运行。

即使不同平台中Java虚拟机的有些细节有所不同，但一个平台上产生的字节码程序总是可以在装有Java虚拟机的另一平台上运行。换句话说，在Unix工作站中编译的Java程序可以在PC机或Mac机上运行。源代码用Java语言按标准方式编写，编译成字节码程序，每个Java虚拟机将字节码程序解释为针对这个平台的自然调用，即特定处理器能够理解的语言。这个抽象使不同操作系统可以用标准化方式实现打印、文件访问和硬件处理之类的操作。

字节码程序的一个特性（也可以说是缺点）是它不是由所在机器上的处理器直接执行，而是通过Java虚拟机解释字节码程序，然后运行。因此Java也称为解释性语言（interpreted language）。解释性语言使Java具有平台独立性，但比标准可执行代码的性能要差。但是，最新版Java开发工具库JDK 1.3解决了性能问题，减少了Java程序与用其他编程语言编写的程序之间的速度差别。

值得一提的是，为了解决这个问题和其他感觉问题，有一个API提供Java应用程序与自然应用程序（用C之类的非Java语言编写的应用程序）之间的接口。这个API称为JNI，使开发人员可以从Java代码中调用用非Java语言编写的代码，从用非Java语言编写的代码中调用Java代码。但是，这样会损害Java的平台独立性，因为自然代码具有平台相关性，相应的Java代码也就具有平台相关性。详见第20章介绍。

为了实现机器可移植性，Java虚拟机必须严格定义。这是通过JVM规范（JVM specification）实现的。这个规范由Sun公司开发和控制，确定Java虚拟机认识的字节码格式和Java虚拟机要实现的特性与功能。JVM规范保证Java语言的平台独立性，见Sun公司Web站点http://java.sun.com/j2se/1.3/docs/index.html。

因此，介绍Java虚拟机时，可能有三个不同含义：

- 抽象规范
- 具体规范实现方法
- 运行时执行环境

不同JVM实现方法

Sun公司Web站点提供了取得Java技术许可证的公司名单。这些公司在特定计算机和操作系统平台上支持Java，这些公司包括IBM、Data General、Sequent Computer Systems、Hewlett-Packard、Silicon Graphics、Blackdown.com、Apple、Novell、Compaq、SCO、Wind River Systems和Digital Equipment Corporation。这些公司在其Web浏览器、服务器和操作系统中嵌入Java虚拟机版本。

Java虚拟机为什么有不同版本呢？记住，JVM规范确定了Java虚拟机必要的功能，但没有规定这些功能如何实现。为了充分扩大Java的用途，Sun公司让第三方具有一定的灵活性。但是，无论用怎样的实现方法，Java虚拟机都要符合JVM规范中建立的准则。就平台相关性而言，要求Java虚拟机能够解释任何其他平台中正确生成的字节码。

JVM运行时执行环境

每次运行Java应用程序时，实际上是在Java虚拟机的一个实例中运行这个应用程序，每个独立的应用程序有自己的Java虚拟机实例。前面介绍了Java使用源代码的解释形式称为字节码，但用Java编程语言编写的指令如何转换成基础操作系统理解的指令呢？

JVM规范对这个过程定义了一个抽象内部体系结构。稍后将介绍这个内部体系结构的组件，但从高层看，类文件（编译的Java文件有一个扩展名：a.class，称为类文件）装入Java虚拟机中，然后由执行引擎（execution engine）执行。执行字节码时，Java虚拟机通过自然方法（native methods）与基础操作系统交互，这些自然方法的实现方法将特定Java虚拟机版本与特定平台相联系，如图1.2所示。

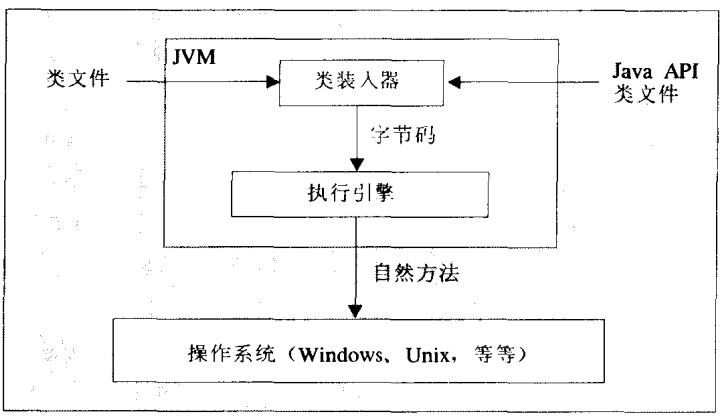


图1.2

此外，对于上述组件，Java虚拟机还需要用内存存储与代码执行有关的临时数据，如局部变量、执行的方法，等等。这个数据存放在Java虚拟机的运行时数据区，见下面介绍。

JVM运行时数据区

尽管不同平台中的各个实现方法可能有所不同，但每个Java虚拟机都要提供下列运行时组件，如图1.3所示。

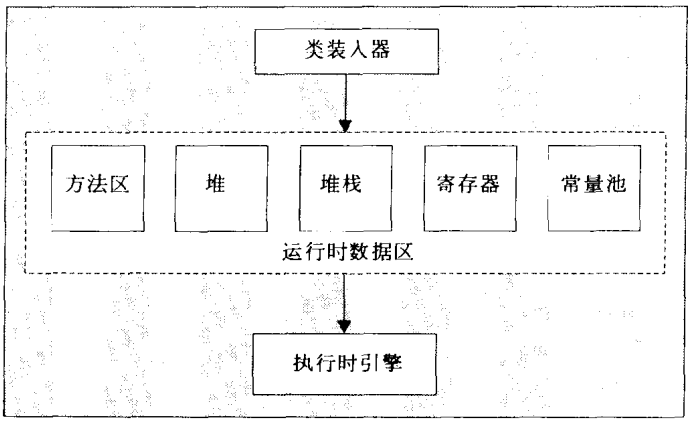


图1.3

堆

堆（heap）是个自由内存区域，常用于动态或临时内存分配。堆是个运行时数据区，对类和数组对象提供内存。当在Java中生成类和数组对象时，从堆中分配所需的内存，这个堆在启动JVM时生成。当对象和数组的引用不复存在时，一个自动化内存管理系统将堆内存释放，这个内存管理系统称为无用信息收集程序（garbage collector），见稍后介绍。

JVM规范没有规定堆的实现方法，给Java虚拟机的不同版本留有一定的创造性。堆的长度可以是固定的，也可以在当前长度太小或太大时增大与缩小。编程人员也可以指定堆的初始长度，例如，在Win32与Solaris系统中，这可以用-mx命令行选项实现。堆内存不一定是相连的。如果堆的内存超界，无法分配更多内存，则系统抛出OutOfMemoryError异常。

堆栈

Java堆栈帧（stack frame）存储方法调用的状态。堆栈帧存储数据和部分结果，包括方法的执行环境，方法调用使用的任何局部变量和方法的操作数堆栈（operand stack）。操作数堆栈存储大多数字节码指令的参数与返回值。执行环境包含方法调用中各个方面的指针。

帧是构成JVM堆栈的组件，用于存储方法调用返回值、数据和部分结果，也用于进行动态链接和发出运行时异常。帧在调用方法时生成，在方法因为出现任何异常而退出时删除。帧包括局部变量数组、操作数堆栈和对当前方法所在类中的运行时常量池的引用。

当JVM运行Java代码时，一次只激活一个帧，对应于当前执行的方法，称为当前帧。其表示的方法是当前方法，定义这个方法的类称为当前类。当线程调用一个方法时（每个线程有自己的堆栈），JVM建立新帧，成为当前帧，并将其推进这个线程的堆栈中。

和堆一样，JVM规范没有规定堆栈帧的具体实现方法。堆栈可以是定长的，或根据需要扩大与缩小。编程人员可以控制堆栈的初始长度及最大长度与最小长度。在Win32与Solaris系统中，这可以通过-ss和-oss命令行选项实现。如果计算要求的堆栈太大，则系统抛出StackOverflowError异常。