

Java领域最有影响力和价值的著作之一，与《Java编程思想》齐名，10余年全球畅销不衰，广受好评

根据Java SE 7全面更新，系统全面讲解Java语言的核心概念、语法、重要特性和开发方法，包含大量案例，实践性强

PEARSON



Java


核心技术 卷 I

基础知识 (原书第9版)

Core Java Volume I—Fundamentals
(Ninth Edition)

Cay S. Horstmann Gary Cornell 著
周立新 陈波 叶乃文 邝劲筠 杜永萍 译



 机械工业出版社
China Machine Press



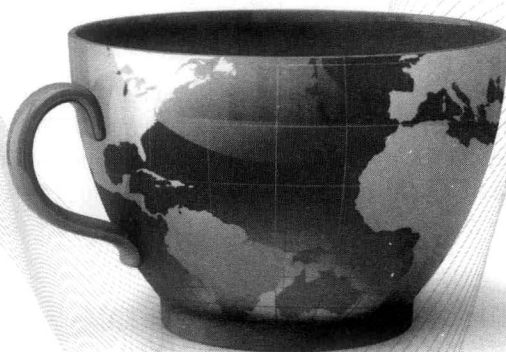
Java

核心技术 卷 I

基础知识 (原书第9版)

Core Java Volume I—Fundamentals
(Ninth Edition)

Cay S. Horstmann Gary Cornell 著
周立新 陈波 叶乃文 邝劲筠 杜永萍 译



机械工业出版社
China Machine Press

图书在版编目 (CIP) 数据

Java 核心技术 卷 I 基础知识 (原书第 9 版)/(美) 霍斯特曼 (Horstmann, C. S.), 科内尔 (Cornell, G.) 著; 周立新等译. —北京: 机械工业出版社, 2013.11

(Java 核心技术系列)

书名原文: Core Java Volume I—Fundamentals (Ninth Edition)

ISBN 978-7-111-44514-2

I. J… II. ①霍… ②科… ③周… III. JAVA 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字 (2013) 第 251592 号

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问 北京市展达律师事务所

本书版权登记号: 图字: 01-2013-2596

Authorized translation from the English language edition, entitled *Core Java Volume I—Fundamentals (Ninth Edition)*, 9780137081899 by Cay S. Horstmann, Gary Cornell, published by Pearson Education, Inc., Copyright © 2013 Oracle and/or its affiliates.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

Chinese Simplified language edition published by Pearson Education Asia Ltd., and China Machine Press Copyright © 2014.

本书中文简体字版由 Pearson Education (培生教育出版集团) 授权机械工业出版社在中华人民共和国境内 (不包括中国台湾地区和香港、澳门特别行政区) 独家出版发行。未经出版者书面许可, 不得以任何方式抄袭、复制或节录本书中的任何部分。

本书封底贴有 Pearson Education (培生教育出版集团) 激光防伪标签, 无标签者不得销售。

Java 领域最有影响力和价值的著作之一, 由拥有 20 多年教学与研究经验的资深 Java 技术专家撰写 (获 Jolt 大奖), 与《Java 编程思想》齐名, 10 余年全球畅销不衰, 广受好评。第 9 版根据 Java SE 7 全面更新, 同时修正了第 8 版中的不足, 系统全面讲解了 Java 语言的核心概念、语法、重要特性和开发方法, 包含大量案例, 实践性强。

本书共 14 章。第 1 章概述 Java 语言与其他程序设计语言不同的性能; 第 2 章讲解如何下载和安装 JDK 及本书的程序示例; 第 3 章介绍变量、循环和简单的函数; 第 4 章讲解类和封装; 第 5 章介绍继承; 第 6 章解释接口和内部类; 第 7 章概述图形用户界面程序设计知识; 第 8 章讨论 AWT 的事件模型; 第 9 章探讨 Swing GUI 工具箱; 第 10 章讲解如何部署自己的应用程序或 applet; 第 11 章讨论异常处理; 第 12 章概要介绍泛型程序设计; 第 13 章讲解 Java 平台的集合框架; 第 14 章介绍多线程。本书最后还有一个附录, 其中列出了 Java 语言的保留字。

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: 关 敏

廊坊市京瑞印刷有限公司印刷

2014 年 1 月第 1 版第 1 次印刷

186mm × 240mm · 45 印张

标准书号: ISBN 978-7-111-44514-2

定 价: 119.00 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

投稿热线: (010) 88379604

购书热线: (010) 68326294 88379649 68995259

读者信箱: hzsjj@hzbook.com

译者序

书写 Java 传奇的 Sun Microsystems 曾经堪称“日不落”帝国，但服务器市场的萎缩却让这个声名赫赫的庞大帝国从蓬勃走向落寞。在 2009 年被 Oracle 收购之后，Sun 逐渐淡出了人们的视线，而与此同时，我们也在很长一段时间内没能看到 Java 当初活跃的身影。

Java 就这样退出历史舞台了吗？当然不是！2011 年 Oracle 终于发布了 Java 的新版本，这就是 Java 7。相较于之前的版本，尽管这一版的改动不大，但让广大 Java 程序员看到了希望，有了前进的动力。

值得一提的是，2009 年之前，伴随着 Java 的成长，《Java 核心技术》也从第 1 版一直更新到第 8 版，得到了广大 Java 程序设计人员的青睐，成了一本畅销不衰的 Java 经典图书。经过几年的蛰伏，为 Java 7 打造的《Java 核心技术》第 9 版终于问世，第 9 版对上一版做了全面修订和更新，以反映 Java 7 增补、删改的内容。它将续写从前的辉煌，使人们能及时跟上 Java 前进的脚步。

本书由周立新、陈波等主译，程芳、刘晓兵、张练达、陈峰、江健、谢连宝、张雷生、杨健康、张莹参与了全书的修改整理，并完善了关键部分的翻译。全体人员共同完成了本书的翻译工作。特别需要说明的是，按照出版社的要求，这一版的翻译是在第 8 版中文版的基础上完成的，因此尤其要感谢第 8 版的译者叶乃文、邝劲筠和杜永萍，他们的辛勤工作作为新版本的翻译奠定了很好的基础。

书中文字与内容力求忠实于原著，不过由于译者水平有限，译文肯定有不当之处，敬请批评指正。

译者

2013 年 10 月于北京

前 言

致读者

1995年年底，Java语言在Internet舞台一亮相便名声大噪。其原因在于它将有成为连接用户与信息的万能胶，而不论这些信息来自Web服务器、数据库、信息提供商，还是任何其他渠道。事实上，就发展前景而言，Java的地位是独一无二的。它是一种完全可信赖的程序设计语言，得到了除微软之外的所有厂家的认可。其固有的可靠性与安全性不仅令Java程序员放心，也令使用Java程序的用户放心。Java内建了对网络编程、数据库连接、多线程等高级程序设计任务的支持。

1995年以来，已经发布了Java开发工具箱（Java Development Kit）的8个主要版本。在过去的17年中，应用程序编程接口（API）已经从200个类扩展到超过3000个类，并覆盖了用户界面构建、数据库管理、国际化、安全性以及XML处理等各个不同的领域。

本书是《Java核心技术》第9版的卷I。自《Java核心技术》出版以来，每个新版本都尽可能快地跟上Java开发工具箱发展的步伐，而且每一版都重新改写了部分内容，以便适应Java的最新特性。在这一版中，已经反映了Java标准版（Java SE 7）的特性。

与前几版一样，本版仍然将读者群定位在那些打算将Java应用到实际工程项目中的程序设计人员。本书假设读者是一名具有程序设计语言（除Java之外）坚实背景知识的程序设计人员，并且不希望书中充斥着玩具式的示例（诸如，烤面包机、动物园的动物或神经质的跳动文本）。这些内容绝对不会在本书中出现。本书的目标是让读者充分理解书中介绍的Java语言及Java类库的相关特性，而不会产生任何误解。

在本书中，我们选用大量的示例代码演示所讨论的每一个语言特性和类库特性。我们有意使用简单的示例程序以突出重点，然而，其中的大部分既不是赝品也没有偷工减料。它们将成为读者自己编写代码的良好开端。

我们假定读者愿意（甚至渴望）学习Java提供的所有高级特性。本书将详细介绍下列内容：

- 面向对象程序设计
- 反射与代理
- 接口与内部类
- 事件监听器模型
- 使用Swing UI工具箱进行图形用户界面设计
- 异常处理
- 泛型程序设计
- 集合框架
- 并行操作

随着Java类库的爆炸式增长，一本书无法涵盖程序员需要了解的所有Java特性。因此，我们决定将本书分为两卷。卷I（本书）集中介绍Java语言的基本概念以及图形用户界面程

序设计的基础知识。卷 II ——高级特性，涉及企业特性以及高级的用户界面程序设计。其中详细讨论下列内容：

- 文件与流
- 分布式对象
- 本地方法
- XML 处理
- 网络编程
- 高级图形
- 数据库
- 高级 GUI 组件
- 国际化
- JavaBeans
- 注释

在编写本书的过程中，难免出现错误和不准确之处。我们很想知道这些错误，当然，也希望同一个问题只被告知一次。我们在网页 <http://horstmann.com/corejava> 中以列表的形式给出了常见的问题、bug 修正和解决方法。在勘误页（建议先阅读一遍）最后附有用来报告 bug 并提出修改意见的表单。如果我们不能回答每一个问题或没有及时回复，请不要失望。我们会认真地阅读所有的来信，感谢您的建议使本书后续的版本更清晰、更有指导价值。

关于本书

第 1 章概述 Java 与其他程序设计语言不同的性能。解释这种语言的设计初衷，以及在哪些方面达到了预期的效果。然后，简要叙述 Java 诞生和发展的历史。

第 2 章详细地论述如何下载和安装 JDK 以及本书的程序示例。然后，通过编译和运行三个典型的 Java 程序（一个控制台应用、一个图形应用、一个 applet），指导读者使用简易的 JDK、可启用 Java 的文本编辑器以及一个 Java IDE。

第 3 章开始讨论 Java 语言。这一章涉及的基础知识有变量、循环以及简单的函数。对于 C 或 C++ 程序员来说，学习这一章的内容将会感觉一帆风顺，因为这些语言特性的语法本质上与 C 语言相同。对于没有 C 语言程序设计背景，但使用过其他程序设计语言（如 Visual Basic）的程序员来说，仔细地阅读这一章是非常必要的。

面向对象程序设计（Object-Oriented Programming, OOP）是当今程序设计的主流，而 Java 是一种完全面向对象的语言。第 4 章将介绍面向对象两个基本成分中最重要的——封装，以及 Java 语言实现封装的机制，即类与方法。除了 Java 语言规则之外，还对如何正确地进行 OOP 设计给出了忠告。最后，介绍奇妙的 javadoc 工具，它将代码注释转换为超链接的网页。熟悉 C++ 的程序员可以快速地浏览这一章，而没有面向对象程序设计背景的程序员，应在进一步学习 Java 之前花一些时间了解 OOP 的有关概念。

类与封装仅仅是 OOP 中的一部分，第 5 章将介绍另一部分——继承。继承使程序员可以使用现有的类，并根据需要进行修改。这是 Java 程序设计中的基础。Java 中的继承机制与 C++ 的继承机制十分相似。C++ 程序员只需关注两种语言的不同之处即可。

第 6 章展示如何使用 Java 的接口。接口可以让你的理解超越第 5 章的简单继承模型。掌握接口的使用将可以获得 Java 的完全的面向对象程序设计的能力。本章还将介绍 Java 的一个有用的技术特性——内部类。内部类可以使代码更清晰、更简洁。

第 7 章开始细致地讨论应用程序设计。每一个 Java 程序员都应该了解一些图形用户界面程序设计的知识，本卷包含了其中的基本内容部分。本章将展示如何制作窗口、如何在窗口中绘图、如何用几何图形作画、如何用多种字体格式化文本以及如何显示图像。

第 8 章详细讨论 AWT (Abstract Window Toolkit) 的事件模型。我们将介绍如何编写代码来响应鼠标点击或按键等事件。同时，还将介绍如何处理基本的 GUI 元素，比如：按钮和面板。

第 9 章详细讨论 Swing GUI 工具箱。Swing 工具箱允许建立一个跨平台的图形用户界面。本章将介绍如何建立各种各样的按钮、文本组件、边框、滑块、列表框、菜单以及对话框等。一些更高级的组件将在卷 II 中讨论。

第 10 章阐述如何部署自己编写的应用程序或 applet。在这里将描述如何将应用程序打包到 JAR 文件中，以及如何使用 Java 的 Web Start 和 applet 机制在 Internet 上发布应用程序。最后，将解释 Java 程序部署之后如何存储和检索配置信息。

第 11 章讨论异常处理，即 Java 的健壮机制，它用于处理调试好的程序可能出现意外的情况。异常提供了一种将正常的处理代码与错误处理代码分开的有效手段。当然，即使程序包含处理所有异常情况的功能，依然有可能无法按照预计的方式工作。这一章的后半部分将给出大量的实用调试技巧。最后，将指导你完成一个完整的示例调试过程。

第 12 章概要介绍泛型程序设计，这是 Java SE 5.0 的一项重要改进。泛型程序设计使得程序拥有更好的可读性和安全性。在这里，将展示如何使用强类型机制，而舍弃不安全的强制类型转换，以及如何处理与旧版本 Java 兼容而带来的复杂问题。





第 13 章介绍 Java 平台的集合框架。当需要将大量对象收集到一起，并在以后要对它们进行检索时，可能会想要使用集合，这是目前最为合适的做法，它取代了将这些元素放置在数组中的做法。本章将介绍如何使用预先建立好的标准集合。

第 14 章是本书的最后一章。在这一章中将介绍多线程，这是一种可以让程序任务并行执行的特性（线程是程序中的控制流），并阐述如何建立线程、如何处理线程的同步问题。从 Java SE 5.0 开始，多线程有了很大的改进，本章将介绍所有这些新的机制。

附录列出了 Java 语言的保留字。

约定

本书使用以下图标表示特殊内容。

-  **注释：**“注释”信息会用这样的“注释”图标标识。
-  **提示：**“提示”信息会用这样的“提示”图标标识。
-  **警告：**对于可能出现的危险，我们用一个“警告”图标做出警示。
-  **C++ 注释：**在本书中有许多用来解释 Java 与 C++ 之间不同的 C++ 注释。对于没有 C++ 程序设计背景，或者不擅长 C++ 程序设计、把它当做一场噩梦不愿再想起的程序员来说，可以跳过这些注释。

API 应用程序编程接口

Java 带有一个很大的程序设计库，即应用程序编程接口。第一次使用 API 调用时，将会在该节的结尾给出一个概要描述。这些描述十分通俗易懂，希望能够比联机 API 文档提供更多的信息。类、接口或方法名后面的编号是介绍该特性的 JDK 版本号。

程序（源代码见本书网站）以如下形式给出：

程序清单 1-1 inputTest/InputTest.java**示例代码**

本书的网站 <http://horstmann.conl/corejava> 以压缩的形式提供了书中的所有示例代码。可以用熟悉的解压缩程序或者用 Java 开发工具箱中的 jar 实用程序解压这个文件。有关安装 Java 开发工具箱和示例程序的详细信息请参看第 2 章。

致 谢

写一本书需要投入大量的精力，改写一本书也并不像想象的那样轻松，尤其是 Java 一直在持续不断地更新。编著一本书让很多人耗费了很多心血，在此衷心地感谢《Java 核心技术》编写小组的每一位成员。

Prentice Hall 公司的许多人提供了非常有价值的帮助，却甘愿做幕后英雄。在此，我希望每一位都能够知道我对他们努力的感恩。与以往一样，我要真诚地感谢我的编辑，Prentice Hall 公司的 Greg Doench，从本书的写作到出版一直给予了指导，同时感谢那些不知其姓名的为本书做出贡献的幕后人士。非常感谢 Julie Nahil 在图书制作方面给予的支持，还要感谢 Dmitry Kirsanov 和 Alina Kirsanova 完成手稿的编辑和排版工作。我还要感谢早期版本中我的合作者，Gary Cornell，他已经转向其他的事业。

感谢早期版本的许多读者，他们指出了许多令人尴尬的错误并给出了许多具有建设性的修改意见。我还要特别感谢本书优秀的审阅小组，他们仔细地审阅我的手稿，使本书减少了许多错误。

本书及早期版本的审阅专家包括：Chuck Allison (UtahValley 大学)、Lance Andersen (Oracle)、Alec Beaton (IBM)、Cliff Berg、Joshua Bloch、David Brown、Corky Cartwright、Frank Cohen (PushToTest)、Chris Crane (devXsolution)、Dr. Nicholas J. De Lillo (Manhatta 学院)、Rakesh Dhoopar (Oracle)、David Geary (Clarity Training)、Jim Gish (Oracle)、Brian Goetz (Oracle)、Angela Gordon、Dan Gordon (Electric Cloud)、Rob Gordon、John Cray (Hartford 大学)、Cameron Gregory (olabs.com)、Marty Hall (coreservlets.com、Inc.)、Vincent Hardy (Adobe Systems)、Dan Harkey (San Jose 州立大学)、William Higgins (IBM)、Vladimir Ivanovic (PointBase)、Jerry Jackson (CA Technologies)、Tim Kimmet (Walmart)、Chris Laffra、Charlie Lai (Apple)、Angelika Langer、Doug Langston、Hang Lau (McGill 大学)、Mark Lawrence、Doug Lea (SUNY Oswego)、Gregory Longshore、Bob Lynch (Lynch Associates)、Philip Milne (顾问)、Mark Morrissey (Oregon 研究院)、Mahesh Neelakanta (Florida Atlantic 大学)、Hao Pham、Paul Phillion、Blake Ragsdell、Stuart Reges (Arizona 大学)、Rich Rosen (Interactive Data Corporation)、Peter Sanders (法国尼斯 ESSI 大学)、Paul Sanghera 博士 (San Jose 州立大学和 Brooks 学院)、Paul Sevinc (Teamup AG)、Devang Shah (Sun Microsystems)、Bradley A. Smith、Steven Stelting (Oracle)、Christopher Taylor、Luke Taylor (Valtech)、George Thiruvathukal、Kim Topley (StreamingEdge)、Janet Traub、Paul Tymal (顾问)、Peter van der Linden (Motorola Mobile Devices)、Burt Walsh、Dan Xu (Oracle) 和 John Zavgren (Oracle)。

Cay Horstmann

2012 年 9 月于加州旧金山

目 录

译者序	
前言	
致谢	
第 1 章 Java 程序设计概述	1
1.1 Java 程序设计平台	1
1.2 Java “白皮书”的关键术语	2
1.2.1 简单性	2
1.2.2 面向对象	3
1.2.3 网络技能	3
1.2.4 健壮性	3
1.2.5 安全性	4
1.2.6 体系结构中立	4
1.2.7 可移植性	5
1.2.8 解释型	5
1.2.9 高性能	5
1.2.10 多线程	6
1.2.11 动态性	6
1.3 Java applet 与 Internet	6
1.4 Java 发展简史	7
1.5 关于 Java 的常见误解	10
第 2 章 Java 程序设计环境	13
2.1 安装 Java 开发工具箱	13
2.1.1 下载 JDK	13
2.1.2 设置执行路径	14
2.1.3 安装库源文件和文档	16
2.1.4 安装本书中的示例	17
2.1.5 导航 Java 目录	17
2.2 选择开发环境	18
2.3 使用命令行工具	18
2.4 使用集成开发环境	20
2.5 运行图形化应用程序	23
2.6 建立并运行 applet	25
第 3 章 Java 的基本程序设计结构	29
3.1 一个简单的 Java 应用程序	29
3.2 注释	32
3.3 数据类型	33
3.3.1 整型	33
3.3.2 浮点类型	34
3.3.3 char 类型	35
3.3.4 boolean 类型	36
3.4 变量	37
3.4.1 变量初始化	37
3.4.2 常量	38
3.5 运算符	39
3.5.1 自增运算符与自减运算符	40
3.5.2 关系运算符与 boolean 运算符	40
3.5.3 位运算符	41
3.5.4 数学函数与常量	42
3.5.5 数值类型之间的转换	43
3.5.6 强制类型转换	43
3.5.7 括号与运算符级别	44
3.5.8 枚举类型	45
3.6 字符串	45
3.6.1 子串	46
3.6.2 拼接	46
3.6.3 不可变字符串	46
3.6.4 检测字符串是否相等	47
3.6.5 空串与 Null 串	48
3.6.6 代码点与代码单元	49
3.6.7 字符串 API	49
3.6.8 阅读联机 API 文档	51

3.6.9 构建字符串	53	4.3.4 从构造器开始	110
3.7 输入输出	54	4.3.5 隐式参数与显式参数	111
3.7.1 读取输入	54	4.3.6 封装的优点	112
3.7.2 格式化输出	56	4.3.7 基于类的访问权限	114
3.7.3 文件输入与输出	60	4.3.8 私有方法	114
3.8 控制流程	61	4.3.9 final 实例域	115
3.8.1 块作用域	62	4.4 静态域与静态方法	115
3.8.2 条件语句	62	4.4.1 静态域	115
3.8.3 循环	65	4.4.2 静态常量	116
3.8.4 确定循环	68	4.4.3 静态方法	117
3.8.5 多重选择: switch 语句	71	4.4.4 工厂方法	118
3.8.6 中断控制流程语句	74	4.4.5 main 方法	118
3.9 大数值	76	4.5 方法参数	121
3.10 数组	78	4.6 对象构造	126
3.10.1 for each 循环	79	4.6.1 重载	126
3.10.2 数组初始化以及匿名数组	80	4.6.2 默认域初始化	126
3.10.3 数组拷贝	80	4.6.3 无参数的构造器	127
3.10.4 命令行参数	81	4.6.4 显式域初始化	128
3.10.5 数组排序	82	4.6.5 参数名	129
3.10.6 多维数组	85	4.6.6 调用另一个构造器	129
3.10.7 不规则数组	87	4.6.7 初始化块	130
第 4 章 对象与类	91	4.6.8 对象析构与 finalize 方法	134
4.1 面向对象程序设计概述	91	4.7 包	134
4.1.1 类	92	4.7.1 类的导入	134
4.1.2 对象	93	4.7.2 静态导入	136
4.1.3 识别类	93	4.7.3 将类放入包中	136
4.1.4 类之间的关系	94	4.7.4 包作用域	139
4.2 使用预定义类	95	4.8 类路径	140
4.2.1 对象与对象变量	95	4.9 文档注释	143
4.2.2 Java 类库中的 Gregorian- Calendar 类	98	4.9.1 注释的插入	143
4.2.3 更改器方法与访问器方法	100	4.9.2 类注释	144
4.3 用户自定义类	106	4.9.3 方法注释	144
4.3.1 Employee 类	106	4.9.4 域注释	145
4.3.2 多个源文件的使用	108	4.9.5 通用注释	145
4.3.3 剖析 Employee 类	109	4.9.6 包与概述注释	146
		4.9.7 注释的抽取	146

4.10 类设计技巧	147	6.4 内部类	231
第 5 章 继承	150	6.4.1 使用内部类访问对象状态	232
5.1 类、超类和子类	150	6.4.2 内部类的特殊语法规则	235
5.1.1 继承层次	156	6.4.3 内部类是否有用、必要和 安全	236
5.1.2 多态	157	6.4.4 局部内部类	238
5.1.3 动态绑定	158	6.4.5 由外部方法访问 final 变量	239
5.1.4 阻止继承: final 类和方法	160	6.4.6 匿名内部类	241
5.1.5 强制类型转换	161	6.4.7 静态内部类	244
5.1.6 抽象类	163	6.5 代理	247
5.1.7 受保护访问	168	第 7 章 图形程序设计	253
5.2 Object: 所有类的超类	169	7.1 Swing 概述	253
5.2.1 equals 方法	169	7.2 创建框架	257
5.2.2 相等测试与继承	171	7.3 框架定位	259
5.2.3 hashCode 方法	173	7.3.1 框架属性	261
5.2.4 toString 方法	175	7.3.2 确定合适的框架大小	262
5.3 泛型数组列表	181	7.4 在组件中显示信息	265
5.3.1 访问数组列表元素	183	7.5 处理 2D 图形	270
5.3.2 类型化与原始数组列表 的兼容性	186	7.6 使用颜色	277
5.4 对象包装器与自动装箱	187	7.7 文本使用特殊字体	280
5.5 参数数量可变的方法	189	7.8 显示图像	287
5.6 枚举类	191	第 8 章 事件处理	291
5.7 反射	192	8.1 事件处理基础	291
5.7.1 Class 类	193	8.1.1 实例: 处理按钮点击事件	293
5.7.2 捕获异常	195	8.1.2 建议使用内部类	297
5.7.3 利用反射分析类的能力	196	8.1.3 创建包含一个方法调用的 监听器	299
5.7.4 在运行时使用反射分析对象	201	8.1.4 实例: 改变观感	300
5.7.5 使用反射编写泛型数组代码	206	8.1.5 适配器类	303
5.7.6 调用任意方法	209	8.2 动作	306
5.8 继承设计的技巧	212	8.3 鼠标事件	312
第 6 章 接口与内部类	215	8.4 AWT 事件继承层次	318
6.1 接口	215	第 9 章 Swing 用户界面组件	322
6.1.1 接口的特性	220	9.1 Swing 和模型 - 视图 - 控制器 设计模式	322
6.1.2 接口与抽象类	222	9.1.1 设计模式	322
6.2 对象克隆	222		
6.3 接口与回调	228		

9.1.2 模型 - 视图 - 控制器模式	323	9.7.3 数据交换	406
9.1.3 Swing 按钮的模型 - 视图 - 控制器分析	326	9.7.4 文件对话框	411
9.2 布局管理概述	327	9.7.5 颜色选择器	421
9.2.1 边框布局	329	第 10 章 部署应用程序和 applet	426
9.2.2 网格布局	331	10.1 JAR 文件	426
9.3 文本输入	334	10.1.1 清单文件	427
9.3.1 文本域	334	10.1.2 可运行 JAR 文件	428
9.3.2 标签和标签组件	336	10.1.3 资源	429
9.3.3 密码域	337	10.1.4 密封	431
9.3.4 文本区	338	10.2 Java Web Start	432
9.3.5 滚动窗格	338	10.2.1 沙箱	435
9.4 选择组件	340	10.2.2 签名代码	436
9.4.1 复选框	340	10.2.3 JNLP API	438
9.4.2 单选按钮	342	10.3 applet	445
9.4.3 边框	345	10.3.1 一个简单的 applet	445
9.4.4 组合框	349	10.3.2 applet 的 HTML 标记和属性	448
9.4.5 滑动条	352	10.3.3 object 标记	451
9.5 菜单	357	10.3.4 使用参数向 applet 传递信息	451
9.5.1 菜单创建	357	10.3.5 访问图像和音频文件	456
9.5.2 菜单项中的图标	359	10.3.6 applet 上下文	457
9.5.3 复选框和单选按钮菜单项	360	10.4 应用程序首选项存储	460
9.5.4 弹出菜单	361	10.4.1 属性映射	460
9.5.5 快捷键和加速器	362	10.4.2 Preferences API	464
9.5.6 启用和禁用菜单项	364	第 11 章 异常、断言、日志和调试	471
9.5.7 工具栏	368	11.1 处理错误	471
9.5.8 工具提示	369	11.1.1 异常分类	473
9.6 复杂的布局管理	370	11.1.2 声明已检查异常	474
9.6.1 网格组布局	372	11.1.3 如何抛出异常	476
9.6.2 组布局	380	11.1.4 创建异常类	477
9.6.3 不使用布局管理器	388	11.2 捕获异常	478
9.6.4 定制布局管理器	388	11.2.1 捕获多个异常	480
9.6.5 遍历顺序	392	11.2.2 再次抛出异常与异常链	481
9.7 对话框	393	11.2.3 finally 子句	482
9.7.1 选项对话框	393	11.2.4 带资源的 try 语句	486
9.7.2 创建对话框	402	11.2.5 分析堆栈跟踪元素	487
		11.3 使用异常机制的技巧	490

11.4 使用断言	492	12.6.7 不能抛出或捕获泛型类的实例	543
11.4.1 启用和禁用断言	493	12.6.8 注意擦除后的冲突	545
11.4.2 使用断言完成参数检查	494	12.7 泛型类型的继承规则	546
11.4.3 为文档假设使用断言	495	12.8 通配符类型	547
11.5 记录日志	496	12.8.1 通配符的超类型限定	549
11.5.1 基本日志	496	12.8.2 无限定通配符	551
11.5.2 高级日志	497	12.8.3 通配符捕获	551
11.5.3 修改日志管理器配置	499	12.9 反射和泛型	553
11.5.4 本地化	500	12.9.1 使用 Class<T> 参数进行类型匹配	554
11.5.5 处理器	500	12.9.2 虚拟机中的泛型类型信息	555
11.5.6 过滤器	503	第 13 章 集合	560
11.5.7 格式化器	504	13.1 集合接口	560
11.5.8 日志记录说明	504	13.1.1 将集合的接口与实现分离	560
11.6 调试技巧	512	13.1.2 Java 类库中的集合接口和迭代器接口	562
11.7 GUI 程序排错技巧	516	13.2 具体的集合	567
11.8 使用调试器	523	13.2.1 链表	568
第 12 章 泛型程序设计	527	13.2.2 数组列表	576
12.1 为什么要使用泛型程序设计	527	13.2.3 散列集	576
12.2 定义简单泛型类	529	13.2.4 树集	579
12.3 泛型方法	531	13.2.5 对象的比较	580
12.4 类型变量的限定	532	13.2.6 队列与双端队列	585
12.5 泛型代码和虚拟机	534	13.2.7 优先级队列	586
12.5.1 翻译泛型表达式	535	13.2.8 映射表	587
12.5.2 翻译泛型方法	536	13.2.9 专用集与映射表类	591
12.5.3 调用遗留代码	537	13.3 集合框架	595
12.6 约束与局限性	538	13.3.1 视图与包装器	598
12.6.1 不能用基本类型实例化类型参数	538	13.3.2 批操作	604
12.6.2 运行时类型查询只适用于原始类型	539	13.3.3 集合与数组之间的转换	605
12.6.3 不能创建参数化类型的数组	539	13.4 算法	606
12.6.4 Varargs 警告	540	13.4.1 排序与混排	607
12.6.5 不能实例化类型变量	541	13.4.2 二分查找	609
12.6.6 泛型类的静态上下文中类型变量无效	542	13.4.3 简单算法	610
		13.4.4 编写自己的算法	612

13.5 遗留的集合.....	613	14.5.11 死锁.....	658
13.5.1 Hashtable 类.....	613	14.5.12 线程局部变量.....	660
13.5.2 枚举.....	613	14.5.13 锁测试与超时.....	661
13.5.3 属性映射表.....	614	14.5.14 读 / 写锁.....	663
13.5.4 栈.....	615	14.5.15 为什么弃用 stop 和 suspend 方法.....	663
13.5.5 位集.....	615	14.6 阻塞队列.....	665
第 14 章 多线程.....	620	14.7 线程安全的集合.....	672
14.1 什么是线程.....	620	14.7.1 高效的映射表、集合和队列.....	672
14.2 中断线程.....	630	14.7.2 写数组的拷贝.....	674
14.3 线程状态.....	633	14.7.3 较早的线程安全集合.....	674
14.3.1 新创建线程.....	633	14.8 Callable 与 Future.....	675
14.3.2 可运行线程.....	633	14.9 执行器.....	679
14.3.3 被阻塞线程和等待线程.....	634	14.9.1 线程池.....	680
14.3.4 被终止的线程.....	634	14.9.2 预定执行.....	683
14.4 线程属性.....	636	14.9.3 控制任务组.....	684
14.4.1 线程优先级.....	636	14.9.4 Fork-Join 框架.....	686
14.4.2 守护线程.....	637	14.10 同步器.....	688
14.4.3 未捕获异常处理器.....	637	14.10.1 信号量.....	689
14.5 同步.....	638	14.10.2 倒计时门栓.....	689
14.5.1 竞争条件的一个例子.....	638	14.10.3 障栅.....	689
14.5.2 竞争条件详解.....	642	14.10.4 交换器.....	690
14.5.3 锁对象.....	644	14.10.5 同步队列.....	690
14.5.4 条件对象.....	647	14.11 线程与 Swing.....	690
14.5.5 synchronized 关键字.....	651	14.11.1 运行耗时的任务.....	692
14.5.6 同步阻塞.....	655	14.11.2 使用 Swing 工作线程.....	696
14.5.7 监视器概念.....	656	14.11.3 单一线程规则.....	701
14.5.8 Volatile 域.....	657	附录 Java 关键字.....	703
14.5.9 final 变量.....	658		
14.5.10 原子性.....	658		

第 1 章 Java 程序设计概述

- ▲ Java 程序设计平台
- ▲ Java “白皮书”的关键术语
- ▲ Java applet 与 Internet
- ▲ Java 发展简史
- ▲ 关于 Java 的常见误解

1996 年 Java 第一次发布就引起了人们的极大兴趣。关注 Java 的人士不仅限于计算机出版界，还有诸如《纽约时报》、《华盛顿邮报》、《商业周刊》这样的主流媒体。Java 是第一种也是唯一一种在 National Public Radio 上占用了 10 分钟时间来进行介绍的程序设计语言，并且还得到了 \$100 000 000 的风险投资基金。这些基金全部用来支持用这种特别的计算机语言开发的产品。重温那些令人兴奋的日子是很有意思的。本章将简要地介绍一下 Java 语言的发展历史。

1.1 Java 程序设计平台

本书的第 1 版是这样描写 Java 的：“作为一种计算机语言，Java 的广告词确实有点夸大其辞。然而，Java 的确是一种优秀的程序设计语言。作为一个名副其实的程序设计人员，使用 Java 无疑是一个好的选择。有人认为：Java 将有望成为一种最优秀的程序设计语言，但仍需要一个相当长的发展时期。一旦一种语言应用于某个领域，与现存代码的相容性问题就摆在了人们的面前。”

我们的编辑手中有许多这样的广告词。这是 Sun 公司高层的某位不愿透露姓名的人士提供的。然而，现在看起来，当初的这些预测还是有一定准确性的。Java 有许多非常优秀的语言特性，本章稍后将会详细地讨论这些特性。由于相容性这个严峻的问题确实存在于现实中，所以，或多或少地还是有一些“累赘”被加到语言中，这就导致 Java 并不如想象中的那么完美无瑕。

但是，正像我们在第 1 版中已经指出的那样，Java 并不只是一种语言。在此之前出现的那么多语言也没有能够引起那么大的轰动。Java 是一个完整的平台，有一个庞大的库，其中包含了很多可重用的代码和一个提供诸如安全性、跨操作系统的可移植性以及自动垃圾收集等服务的执行环境。

作为一名程序设计人员，常常希望能够有一种语言，它具有令人赏心悦目的语法和易于理解的语义（C++ 不是这样的）。与许多其他的优秀语言一样，Java 恰恰满足了这些要求。有些语言提供了可移植性、垃圾收集等，但是，没有提供一个大型的库。如果想要有奇特的绘图功能、网络连接功能和数据库存取功能就必须自己动手编写代码。Java 这种功能齐全的出

色语言，具有高质量的执行环境以及庞大的库。正是因为它集多种优势于一身，所以对广大的程序设计人员有着不可抗拒的吸引力。


1.2 Java “白皮书”的关键术语

Java 的设计者已经编写了颇有影响力的“白皮书”，用来解释设计的初衷以及完成的情况，并且发布了一个简短的摘要。这个摘要用下面 11 个关键术语进行组织：

- 1) 简单性
- 2) 面向对象
- 3) 网络技能 (Network-Savvy)
- 4) 健壮性
- 5) 安全性
- 6) 体系结构中立
- 7) 可移植性
- 8) 解释型
- 9) 高性能
- 10) 多线程
- 11) 动态性

本节将论述下列主要内容：

- 给出白皮书中对每个关键术语的概述，这是 Java 设计者对相关术语的论述。
- 凭借 Java 当前版本的使用经验，给出对这些术语的理解。

 **注释：**写这本书时，白皮书可以在 www.oracle.com/technetwork/java/langenv-140151.html 上找到。对于 11 个关键术语的论述请参看 <http://labs.oracle.com/features/tenyeares/volcd/papers/7Gosling.pdf>。

1.2.1 简单性

人们希望构建一个无须深奥的专业训练就可以进行编程的系统，并且要符合当今的标准惯例。因此，尽管人们发现 C++ 不太适用，但在设计 Java 的时候还是尽可能地接近 C++，以便系统更易于理解。Java 剔除了 C++ 中许多很少使用、难以理解、易混淆的特性。在目前看来，这些特性带来的麻烦远远多于其带来的好处。

的确，Java 语法是 C++ 语法的一个“纯净”版本。这里没有头文件、指针运算（甚至指针语法）、结构、联合、操作符重载、虚基类等（请参阅本书各个章节给出的 C++ 注释，那里比较详细地解释了 Java 与 C++ 之间的区别）。然而，设计者并没有试图清除 C++ 中所有不适当的特性。例如，switch 语句的语法在 Java 中就没有改变。如果知道 C++ 就会发现可以轻而易举地将其转换成 Java。

如果已经习惯于使用可视化的编程环境（例如 Visual Basic），你就不会觉得 Java 简单了。Java 有许多奇怪的语法（尽管掌握其要领并不需要很长时间），更重要的是，使用 Java 需要自己编写大量的程序。Visual Basic 的魅力在于它的可视化设计环境几乎自动地为应用程序提供了大量的基础结构。而使用 Java 实现同样的功能却需要手工编制代码，通常代码量还相当大。然而，已经有一些支持“拖放”风格程序开发的第三方开发环境。