



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 8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



Java

核心技术 卷 I

基础知识 (原书第10版)

Core Java Volume I—Fundamentals
(10th Edition)

[美] 凯 S. 霍斯特曼 (Cay S. Horstmann) 著
周立新 陈波 叶乃文 邝劲筠 杜永萍 译





Java

核心技术 卷 I

基础知识 (原书第10版)

Core Java Volume I—Fundamentals
(10th Edition)

[美] 凯 S. 霍斯特曼 (Cay S. Horstmann) 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 10 版)/(美) 凯 S. 霍斯特曼 (Cay S. Horstmann) 著; 周立新等译. —北京: 机械工业出版社, 2016.8

(Java 核心技术系列)

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

ISBN 978-7-111-54742-6

I. J… II. ①凯… ②周… III. JAVA 语言—程序设计 IV. TP312.8

中国版本图书馆 CIP 数据核字 (2016) 第 211440 号

本书版权登记号: 图字: 01-2016-5145

Authorized translation from the English language edition, entitled *Core Java Volume I—Fundamentals (Tenth Edition)*, 9780134177304 by Cay S. Horstmann, published by Pearson Education, Inc., Copyright © 2016 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 © 2016.

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 (不包括香港、澳门特别行政区及台湾地区) 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 核心技术 卷 I 基础知识 (原书第 10 版)

出版发行: 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

责任校对: 董纪丽

印 刷: 北京诚信伟业印刷有限公司

版 次: 2016 年 9 月第 1 版第 1 次印刷

开 本: 186mm × 240mm 1/16

印 张: 45.5

书 号: ISBN 978-7-111-54742-6

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) 88379426 88361066

投稿热线: (010) 88379604

购书热线: (010) 68326294 88379649 68995259

读者信箱: hzit@hzbook.com

版权所有 • 侵权必究

封底无防伪标均为盗版

本书法律顾问: 北京大成律师事务所 韩光 / 邹晓东

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 公司收购之后，Sun 公司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！从 Sun 公司 2006 年 12 月发布 Java 6 后，经过 5 年多的不懈努力，终于在 2011 年 7 月底发布了 Java 7 正式版。3 年后，被冠名为“跳票王”的 Oracle 终于发布了 Java 8 的正式版，但对于很多开发者来说，Java 8 却比 Java 7 来得更漫长一些。主要是因为 Oracle 原本计划在 2013 年发布正式版 Java 8，却因受困于安全性的问题经过了两次“跳票”。无论如何，如今 Java 8 来了，全新“革命”而不只是“进化”的功能将会让无数开发者动容。

值得一提的是，伴随着 Java 的成长，《Java 核心技术》也从第 1 版到第 9 版一路走来，得到了广大 Java 程序设计人员的青睐，成为一本畅销不衰的 Java 经典图书。经过几年的蛰伏，针对 Java 8 打造的《Java 核心技术》第 10 版终于问世，这一版有了大幅的修订和更新，以反映 Java 8 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，这一版的翻译在老版本基础上完成，因此尤其感谢之前版本的译者叶乃文、邝劲筠和杜永萍，他们的辛勤工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

2016 年 6 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开发工具包（Java Development Kit）的9个主要版本。在过去的20年中，应用程序编程接口（API）已经从200个类扩展到超过4000个类。现在这些API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10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 8）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。例如，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
- 异常处理
- 泛型程序设计
- 集合框架

- 事件监听器模型
- 使用 Swing UI 工具箱进行图形用户界面设计
- 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：

- 流 API
- 文件处理与正则表达式
- 数据库
- XML 处理
- 注释
- 国际化
- 网络编程
- 高级 GUI 组件
- 高级图形
- 原生方法

本书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 3 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，还对如何完成合理的 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一组包含超链接的网页。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 程序设计中的一个基础技术。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。掌握接口可以充分获得 Java 的完全的面向对象程序设计能力。介绍接口之后，我们将转而介绍 *lambda* 表达式 (*lambda expression*)，这是一种简洁的方法，用来表述可以在以后某个时间点执行的代码块。本章还将介绍 Java 的一个有用的技术特性——内部类。

第 7 章讨论异常处理 (*exception handling*)，即 Java 的一种健壮机制，用于处理可正常运行程序可能出现意外的情况。异常提供了一种将正常处理代码与错误处理代码分开的有效手段。当然，即使程序能够处理所有异常条件，仍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实用的调试技巧。

第 8 章概要介绍泛型程序设计。泛型程序设计可以让程序更可读、更安全。我们会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 兼容所带来的复杂问题。

第 9 章讨论的是 Java 平台的集合框架。如果希望收集多个对象并在以后获取这些对象，就应当使用集合，而不要简单地把这些元素放在一个数组中，这是这种情况下最适用的做法。这一章会介绍如何充分利用内建的标准集合。

第 10 章开始介绍 GUI 程序设计。我们会讨论如何建立窗口、如何在窗口中绘图、如何利用几何图形绘图、如何采用多种字体格式化文本，以及如何显示图像。

第 11 章将详细讨论抽象窗口工具包 (*abstract window toolkit*, AWT) 的事件模型。你会看到如何编写代码来响应事件，如鼠标点击事件或按键事件。同时，你还会看到如何处理基本的 GUI 元素，如按钮和面板。

第 12 章详细讨论 Swing GUI 工具包。Swing 工具包允许建立跨平台的图像用户界面。在这里你会了解各种按钮、文本组件、边框、滑块、列表框、菜单以及对话框的有关内容。不过，一些更高级的组件会在卷 II 中讨论。





第 13 章介绍如何将程序部署为应用或 applet。在这里我们会描述如何将程序打包在 JAR 文件中，以及如何使用 Java Web Start 和 applet 机制在 Internet 上发布应用。另外还会解释 Java 程序部署之后如何存储和获取配置信息。

第 14 章是本书的最后一章，这一章将讨论并发，并发能够让程序任务并行执行。在当今这个时代，大多数处理器都有多个内核，你往往希望这些内核都在工作，并发是 Java 技术的一个重要而且令人振奋的应用。

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”图标标志。
-  **提示：**“提示”信息会用这样的“提示”图标标志。
-  **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++ 之间差别的 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。

Java 提供了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，我们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，如下例所示：

应用程序编程接口 1.2

程序（源代码见本书网站）以程序清单形式给出，例如：

程序清单 1-1 InputTest/InputTest.java

示例代码

本书网站 <http://horstmann.com/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 开发包中的 jar 实用程序解压这个文件。有关安装 Java 开发包和示例代码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 技术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他一直在给予我们指导，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 (Utah Valley 大学)、Lance Andersen (Oracle)、Paul Anderson (Anderson Software Group)、Alec Beaton (IBM)、Cliff Berg、Andrew Binstock (Oracle)、Joshua Bloch、David Brown、Corky Cartwright、Frank Cohen (PushToTest)、Chris Crane (devXsolution)、Dr. Nicholas J. De Lillo (Manhattan 学院)、Rakesh Dhoopar (Oracle)、David Geary (Clarity Training)、Jim Gish (Oracle)、Brian Goetz (Oracle)、Angela Gordon、Dan Gordon (Electric Cloud)、Rob Gordon、John Gray (Hartford 大学)、Cameron Gregory (olabs.com)、Marty Hall (coreservlets.com 公司)、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 (consultant)、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)、Dr. Paul Sanghera (San Jose 州立大学 Brooks 学院)、Paul Sevinc (Teamup AG)、Devang Shah (Sun Microsystems)、Yoshiki Shibata、Bradley A. Smith、Steven Stelling (Oracle)、Christopher Taylor、Luke Taylor (Valtech)、George Thiruvathukal、Kim Topley (StreamingEdge)、Janet Traub、Paul Tyma (consultant)、Peter van der Linden、Christian Ullenboom、Burt Walsh、Dan Xu (Oracle) 和 John Zavgren (Oracle)。

Cay Horstmann

2015 年 11 月于瑞士比尔

目 录

译者序		第 3 章 Java 的基本程序设计结构	28
前言		3.1 一个简单的 Java 应用程序	28
致谢		3.2 注释	31
第 1 章 Java 程序设计概述	1	3.3 数据类型	32
1.1 Java 程序设计平台	1	3.3.1 整型	32
1.2 Java “白皮书”的关键术语	2	3.3.2 浮点类型	33
1.2.1 简单性	2	3.3.3 char 类型	34
1.2.2 面向对象	2	3.3.4 Unicode 和 char 类型	35
1.2.3 分布式	3	3.3.5 boolean 类型	35
1.2.4 健壮性	3	3.4 变量	36
1.2.5 安全性	3	3.4.1 变量初始化	37
1.2.6 体系结构中立	4	3.4.2 常量	37
1.2.7 可移植性	4	3.5 运算符	38
1.2.8 解释型	5	3.5.1 数学函数与常量	39
1.2.9 高性能	5	3.5.2 数值类型之间的转换	40
1.2.10 多线程	5	3.5.3 强制类型转换	41
1.2.11 动态性	5	3.5.4 结合赋值和运算符	42
1.3 Java applet 与 Internet	6	3.5.5 自增与自减运算符	42
1.4 Java 发展简史	7	3.5.6 关系和 boolean 运算符	42
1.5 关于 Java 的常见误解	9	3.5.7 位运算符	43
第 2 章 Java 程序设计环境	12	3.5.8 括号与运算符级别	44
2.1 安装 Java 开发工具包	12	3.5.9 枚举类型	45
2.1.1 下载 JDK	12	3.6 字符串	45
2.1.2 设置 JDK	13	3.6.1 子串	45
2.1.3 安装库源文件和文档	15	3.6.2 拼接	46
2.2 使用命令行工具	16	3.6.3 不可变字符串	46
2.3 使用集成开发环境	18	3.6.4 检测字符串是否相等	47
2.4 运行图形化应用程序	21	3.6.5 空串与 Null 串	48
2.5 构建并运行 applet	23	3.6.6 码点与代码单元	49
		3.6.7 String API	50

3.6.8	阅读联机 API 文档	52	4.3.2	多个源文件的使用	105
3.6.9	构建字符串	54	4.3.3	剖析 Employee 类	106
3.7	输入输出	55	4.3.4	从构造器开始	106
3.7.1	读取输入	55	4.3.5	隐式参数与显式参数	108
3.7.2	格式化输出	58	4.3.6	封装的优点	109
3.7.3	文件输入与输出	61	4.3.7	基于类的访问权限	111
3.8	控制流程	63	4.3.8	私有方法	111
3.8.1	块作用域	63	4.3.9	final 实例域	112
3.8.2	条件语句	63	4.4	静态域与静态方法	112
3.8.3	循环	66	4.4.1	静态域	112
3.8.4	确定循环	69	4.4.2	静态常量	113
3.8.5	多重选择: switch 语句	72	4.4.3	静态方法	114
3.8.6	中断控制流程语句	74	4.4.4	工厂方法	115
3.9	大数值	76	4.4.5	main 方法	115
3.10	数组	78	4.5	方法参数	118
3.10.1	for each 循环	79	4.6	对象构造	123
3.10.2	数组初始化以及匿名数组	80	4.6.1	重载	123
3.10.3	数组拷贝	81	4.6.2	默认域初始化	123
3.10.4	命令行参数	81	4.6.3	无参数的构造器	124
3.10.5	数组排序	82	4.6.4	显式域初始化	125
3.10.6	多维数组	85	4.6.5	参数名	125
3.10.7	不规则数组	88	4.6.6	调用另一个构造器	126
第 4 章	对象与类	91	4.6.7	初始化块	127
4.1	面向对象程序设计概述	91	4.6.8	对象析构与 finalize 方法	130
4.1.1	类	92	4.7	包	131
4.1.2	对象	93	4.7.1	类的导入	131
4.1.3	识别类	93	4.7.2	静态导入	133
4.1.4	类之间的关系	94	4.7.3	将类放入包中	133
4.2	使用预定义类	95	4.7.4	包作用域	136
4.2.1	对象与对象变量	95	4.8	类路径	137
4.2.2	Java 类库中的 LocalDate 类	98	4.8.1	设置类路径	139
4.2.3	更改器方法与访问器方法	100	4.9	文档注释	140
4.3	用户自定义类	103	4.9.1	注释的插入	140
4.3.1	Employee 类	103	4.9.2	类注释	140
			4.9.3	方法注释	141

4.9.4 域注释	142	5.7.6 调用任意方法	205
4.9.5 通用注释	142	5.8 继承的设计技巧	208
4.9.6 包与概述注释	143	第 6 章 接口、lambda 表达式与 内部类	211
4.9.7 注释的抽取	143	6.1 接口	211
4.10 类设计技巧	144	6.1.1 接口概念	211
第 5 章 继承	147	6.1.2 接口的特性	217
5.1 类、超类和子类	147	6.1.3 接口与抽象类	218
5.1.1 定义子类	147	6.1.4 静态方法	218
5.1.2 覆盖方法	149	6.1.5 默认方法	219
5.1.3 子类构造器	150	6.1.6 解决默认方法冲突	220
5.1.4 继承层次	153	6.2 接口示例	222
5.1.5 多态	154	6.2.1 接口与回调	222
5.1.6 理解方法调用	155	6.2.2 Comparator 接口	224
5.1.7 阻止继承: final 类和方法	157	6.2.3 对象克隆	225
5.1.8 强制类型转换	158	6.3 lambda 表达式	231
5.1.9 抽象类	160	6.3.1 为什么引入 lambda 表达式	231
5.1.10 受保护访问	165	6.3.2 lambda 表达式的语法	232
5.2 Object: 所有类的超类	166	6.3.3 函数式接口	234
5.2.1 equals 方法	166	6.3.4 方法引用	235
5.2.2 相等测试与继承	167	6.3.5 构造器引用	237
5.2.3 hashCode 方法	170	6.3.6 变量作用域	237
5.2.4 toString 方法	172	6.3.7 处理 lambda 表达式	239
5.3 泛型数组列表	178	6.3.8 再谈 Comparator	242
5.3.1 访问数组列表元素	180	6.4 内部类	242
5.3.2 类型化与原始数组列表的 兼容性	183	6.4.1 使用内部类访问对象状态	244
5.4 对象包装器与自动装箱	184	6.4.2 内部类的特殊语法规则	247
5.5 参数数量可变的方法	187	6.4.3 内部类是否有用、必要和 安全	248
5.6 枚举类	188	6.4.4 局部内部类	250
5.7 反射	190	6.4.5 由外部方法访问变量	250
5.7.1 Class 类	190	6.4.6 匿名内部类	252
5.7.2 捕获异常	192	6.4.7 静态内部类	255
5.7.3 利用反射分析类的能力	194	6.5 代理	258
5.7.4 在运行时使用反射分析对象	198		
5.7.5 使用反射编写泛型数组代码	202		

6.5.1	何时使用代理	259	8.1.2	谁想成为泛型程序员	310
6.5.2	创建代理对象	259	8.2	定义简单泛型类	311
6.5.3	代理类的特性	262	8.3	泛型方法	313
第 7 章	异常、断言和日志	264	8.4	类型变量的限定	314
7.1	处理错误	264	8.5	泛型代码和虚拟机	316
7.1.1	异常分类	265	8.5.1	类型擦除	316
7.1.2	声明受查异常	267	8.5.2	翻译泛型表达式	317
7.1.3	如何抛出异常	269	8.5.3	翻译泛型方法	318
7.1.4	创建异常类	270	8.5.4	调用遗留代码	319
7.2	捕获异常	271	8.6	约束与局限性	320
7.2.1	捕获异常	271	8.6.1	不能用基本类型实例化 类型参数	320
7.2.2	捕获多个异常	273	8.6.2	运行时类型查询只适用于 原始类型	321
7.2.3	再次抛出异常与异常链	274	8.6.3	不能创建参数化类型的数组	321
7.2.4	finally 子句	275	8.6.4	Varargs 警告	322
7.2.5	带资源的 try 语句	278	8.6.5	不能实例化类型变量	323
7.2.6	分析堆栈轨迹元素	280	8.6.6	不能构造泛型数组	323
7.3	使用异常机制的技巧	282	8.6.7	泛型类的静态上下文中 类型变量无效	325
7.4	使用断言	285	8.6.8	不能抛出或捕获泛型类的 实例	325
7.4.1	断言的概念	285	8.6.9	可以消除对受查异常的检查	326
7.4.2	启用和禁用断言	286	8.6.10	注意擦除后的冲突	327
7.4.3	使用断言完成参数检查	287	8.7	泛型类型的继承规则	328
7.4.4	为文档假设使用断言	288	8.8	通配符类型	330
7.5	记录日志	289	8.8.1	通配符概念	330
7.5.1	基本日志	289	8.8.2	通配符的超类型限定	331
7.5.2	高级日志	289	8.8.3	无限定通配符	334
7.5.3	修改日志管理器配置	291	8.8.4	通配符捕获	334
7.5.4	本地化	292	8.9	反射和泛型	337
7.5.5	处理器	293	8.9.1	泛型 Class 类	337
7.5.6	过滤器	296	8.9.2	使用 Class<T> 参数进行 类型匹配	338
7.5.7	格式化器	296	8.9.3	虚拟机中的泛型类型信息	338
7.5.8	日志记录说明	296			
7.6	调试技巧	304			
第 8 章	泛型程序设计	309			
8.1	为什么要使用泛型程序设计	309			
8.1.1	类型参数的好处	309			

第 9 章 集合	344	9.5.6 编写自己的算法	395
9.1 Java 集合框架	344	9.6 遗留的集合	396
9.1.1 将集合的接口与实现分离	344	9.6.1 Hashtable 类	397
9.1.2 Collection 接口	346	9.6.2 枚举	397
9.1.3 迭代器	347	9.6.3 属性映射	398
9.1.4 泛型实用方法	349	9.6.4 栈	399
9.1.5 集合框架中的接口	352	9.6.5 位集	399
9.2 具体的集合	353	第 10 章 图形程序设计	403
9.2.1 链表	355	10.1 Swing 概述	403
9.2.2 数组列表	362	10.2 创建框架	407
9.2.3 散列集	363	10.3 框架定位	409
9.2.4 树集	366	10.3.1 框架属性	411
9.2.5 队列与双端队列	369	10.3.2 确定合适的框架大小	411
9.2.6 优先级队列	371	10.4 在组件中显示信息	415
9.3 映射	372	10.5 处理 2D 图形	419
9.3.1 基本映射操作	372	10.6 使用颜色	426
9.3.2 更新映射项	375	10.7 文本使用特殊字体	429
9.3.3 映射视图	376	10.8 显示图像	435
9.3.4 弱散列映射	377	第 11 章 事件处理	439
9.3.5 链接散列集与映射	378	11.1 事件处理基础	439
9.3.6 枚举集与映射	379	11.1.1 实例：处理按钮点击事件	441
9.3.7 标识散列映射	380	11.1.2 简洁地指定监听器	445
9.4 视图与包装器	381	11.1.3 实例：改变观感	447
9.4.1 轻量级集合包装器	382	11.1.4 适配器类	450
9.4.2 子范围	382	11.2 动作	453
9.4.3 不可修改的视图	383	11.3 鼠标事件	459
9.4.4 同步视图	384	11.4 AWT 事件继承层次	465
9.4.5 受查视图	384	11.4.1 语义事件和底层事件	466
9.4.6 关于可选操作的说明	385	第 12 章 Swing 用户界面组件	469
9.5 算法	388	12.1 Swing 和模型 - 视图 - 控制器	
9.5.1 排序与混排	389	设计模式	469
9.5.2 二分查找	391	12.1.1 设计模式	469
9.5.3 简单算法	392	12.1.2 模型 - 视图 - 控制器模式	470
9.5.4 批操作	394	12.1.3 Swing 按钮的模型 - 视图 -	
9.5.5 集合与数组的转换	394	控制器分析	473

12.2 布局管理概述	474	12.7.5 颜色选择器	569
12.2.1 边框布局	477	12.8 GUI 程序排错	573
12.2.2 网格布局	478	12.8.1 调试技巧	573
12.3 文本输入	481	12.8.2 让 AWT 机器人完成工作	576
12.3.1 文本域	482	第 13 章 部署 Java 应用程序	580
12.3.2 标签和标签组件	483	13.1 JAR 文件	580
12.3.3 密码域	484	13.1.1 创建 JAR 文件	580
12.3.4 文本区	485	13.1.2 清单文件	581
12.3.5 滚动窗格	485	13.1.3 可执行 JAR 文件	582
12.4 选择组件	488	13.1.4 资源	583
12.4.1 复选框	488	13.1.5 密封	585
12.4.2 单选钮	490	13.2 应用首选项的存储	586
12.4.3 边框	493	13.2.1 属性映射	586
12.4.4 组合框	496	13.2.2 首选项 API	591
12.4.5 滑动条	499	13.3 服务加载器	596
12.5 菜单	504	13.4 applet	598
12.5.1 菜单创建	504	13.4.1 一个简单的 applet	599
12.5.2 菜单项中的图标	507	13.4.2 applet HTML 标记和属性	602
12.5.3 复选框和单选钮菜单项	508	13.4.3 使用参数向 applet 传递	
12.5.4 弹出菜单	508	信息	603
12.5.5 快捷键和加速器	510	13.4.4 访问图像和音频文件	608
12.5.6 启用和禁用菜单项	511	13.4.5 applet 上下文	609
12.5.7 工具栏	515	13.4.6 applet 间通信	609
12.5.8 工具提示	516	13.4.7 在浏览器中显示信息项	610
12.6 复杂的布局管理	518	13.4.8 沙箱	611
12.6.1 网格组布局	520	13.4.9 签名代码	612
12.6.2 组布局	528	13.5 Java Web Start	614
12.6.3 不使用布局管理器	537	13.5.1 发布 Java Web Start 应用	614
12.6.4 定制布局管理器	537	13.5.2 JNLP API	617
12.6.5 遍历顺序	541	第 14 章 并发	624
12.7 对话框	541	14.1 什么是线程	624
12.7.1 选项对话框	542	14.1.1 使用线程给其他任务	
12.7.2 创建对话框	551	提供机会	629
12.7.3 数据交换	554	14.2 中断线程	632
12.7.4 文件对话框	559	14.3 线程状态	635

14.3.1	新创建线程	635	14.7	线程安全的集合	673
14.3.2	可运行线程	635	14.7.1	高效的映射、集和队列	674
14.3.3	被阻塞线程和等待线程	636	14.7.2	映射条目的原子更新	675
14.3.4	被终止的线程	636	14.7.3	对并发散列映射的批操作	676
14.4	线程属性	638	14.7.4	并发集视图	678
14.4.1	线程优先级	638	14.7.5	写数组的拷贝	679
14.4.2	守护线程	639	14.7.6	并行数组算法	679
14.4.3	未捕获异常处理器	639	14.7.7	较早的线程安全集合	680
14.5	同步	640	14.8	Callable 与 Future	681
14.5.1	竞争条件的一个例子	641	14.9	执行器	685
14.5.2	竞争条件详解	644	14.9.1	线程池	685
14.5.3	锁对象	646	14.9.2	预定执行	689
14.5.4	条件对象	648	14.9.3	控制任务组	690
14.5.5	synchronized 关键字	653	14.9.4	Fork-Join 框架	691
14.5.6	同步阻塞	656	14.9.5	可完成 Future	694
14.5.7	监视器概念	657	14.10	同步器	696
14.5.8	Volatile 域	658	14.10.1	信号量	696
14.5.9	final 变量	659	14.10.2	倒计时门栓	697
14.5.10	原子性	659	14.10.3	障栅	697
14.5.11	死锁	661	14.10.4	交换器	698
14.5.12	线程局部变量	663	14.10.5	同步队列	698
14.5.13	锁测试与超时	665	14.11	线程与 Swing	698
14.5.14	读 / 写锁	666	14.11.1	运行耗时的任务	699
14.5.15	为什么弃用 stop 和 suspend 方法	667	14.11.2	使用 Swing 工作线程	703
14.6	阻塞队列	668	14.11.3	单一线程规则	708
			附录 A	Java 关键字	710

第 1 章 Java 程序设计概述

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

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（Sun 是原先开发 Java 的公司）。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 完全满足了这些要求。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 具备所有这些特性，它是一种功能齐全的出色语言，是一个高质量的执行环境，还提供了一个庞大的库。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。