

PEARSON

JAVATM 核心技术

Core Java Volume I — Fundamentals, Ninth Edition

卷 I：基础知识 **上**

(第9版 · 英文版)



[美] Cay S. Horstmann Gary Cornell 著

JAVA 核心技术

Core Java Volume I — Fundamentals, Ninth Edition

卷 I：基础知识 上

(第9版·英文版)



[美] Cay S. Horstmann Gary Cornell 著

人民邮电出版社
北京

图书在版编目 (C I P) 数据

Java核心技术. 第1卷, 基础知识 : 第9版 : 英文 /
(美) 霍斯特曼 (Horstmann, C. S.), (美) 康奈尔
(Cornell, G.) 著. — 北京 : 人民邮电出版社, 2013. 7
ISBN 978-7-115-31947-0

I. ①J… II. ①霍… ②康… III. ①
JAVA语言—程序设计—英文 IV. ①TP312

中国版本图书馆CIP数据核字 (2013) 第099954号

内 容 提 要

本书是经典的《Java 核心技术 卷 I: 基础知识》的最新版。这一版针对 Java SE 7 平台进行了全面更新, 以反映 Java SE 7 的特性。

书中囊括了 Java 的全部基础知识, 提供了大量完整且具有实际意义的应用示例, 详细介绍了 Java 语言基础、面向对象编程、反射与代理、接口与内部类、事件监听器模型、使用 Swing GUI 工具进行图形用户界面程序设计、打包应用程序、异常处理、登录与调试、泛型编程、集合框架、多线程等内容。

本书适合想将 Java 应用于实际项目的软件开发人员、高等院校教师和学生参考阅读。

-
- ◆ 著 [美] Cay S. Horstmann Gary Cornell
责任编辑 杨海玲
责任印制 程彦红 杨林杰
 - ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子邮件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京中新伟业印刷有限公司印刷
 - ◆ 开本: 800×1000 1/16
印张: 62.75
字数: 1197 千字 2013 年 7 月第 1 版
印数: 1—2 500 册 2013 年 7 月北京第 1 次印刷

著作权合同登记号 图字: 01-2013-3461 号

定价: 109.00 元 (上、下册)

读者服务热线: (010) 67132692 印装质量热线: (010) 67129223

反盗版热线: (010) 67171154

版权声明

Original edition, *Core Java Volume I—Fundamentals, Ninth Edition*, 9780137081899, by Cay S. Horstmann and Gary Cornell, published by Pearson Education, Inc., publishing as Prentice-Hall, 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 Inc.

English reprint published by Pearson Education North Asia Limited and Posts & Telecommunication Press, Copyright © 2013.

This edition is manufactured in the People's Republic of China, and is authorized for sale and distribution in the People's Republic of China exclusively (except Taiwan, Hong Kong SAR and Macau SAR).

本书封面贴有 **Pearson Education** 出版集团激光防伪标签，无标签者不得销售。

前言^①

致读者

1995 年底，Java 编程语言异军突起，登上 Internet 舞台，并且一跃成名。Java 技术希望成为连接用户与信息的“万能胶”，而不论这些信息来自哪里——是来自 Web 服务器、数据库、信息提供商，还是来自其他任何信息源。而且确实，要实现这一目标，Java 具有得天独厚的条件。Java 是极为稳健的语言，得到了除微软公司之外所有主流厂商的支持。其固有的安全性不仅令 Java 程序员放心，也令使用 Java 程序的用户放心。Java 甚至内置了对网络编程、数据库连接和多线程等高级编程任务的支持。

自 1995 年起，Java 开发工具包（Java Development Kit, JDK）已经发布了 8 个主要版本。在过去的 17 年中，应用编程接口（API）已经从大约 200 个类扩展到 3000 多个类。API 现在已覆盖多个不同领域，如用户界面构建、数据库管理、国际化、安全和 XML 处理等。

现在你手中的这本书是《Java 核心技术》一书第 9 版的卷 I。本书的每一版都紧跟一个 JDK 版本，而且为了利用最新的 Java 特性，我们每一次都会对书进行重写。这一版也做了更新，以反映 Java SE 7 的特性。

与本书之前的版本一样，本书的读者定位仍是那些希望将 Java 技术应用于实际项目的程序员。本书假设你是基础扎实的程序员，已掌握除 Java 之外的某种编程语言，并且不希望书中充斥着玩具式的示例（比如烤面包机、动物园里的动物甚至跳动的文字）。本书不会使用任何这样的玩具示例。我们的目标是让你充分理解 Java 语言及其库，而不是造成一种理解的错觉。

本书会提供大量的示例代码，用以说明我们所讨论的几乎每一个语言和库特性。我们有意让示例代码尽量简单，以便可以专注于主要问题。不过绝大多数代码既非杜撰而成，也未偷工减料。对你编写自己的代码而言，它们是很好的起点。

① 本前言由臧秀涛翻译。

我们假定你愿意，甚至是渴望学习 Java 提供的所有高级特性。举例来说，我们会详细讨论如下内容：

- 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 工具包进行图形用户界面（Graphical User Interface, GUI）设计；
- 异常处理；
- 泛型编程；
- 集合框架；
- 并发。

随着 Java 类库的爆炸式增长，一本书已经无法涵盖程序员所需了解的所有 Java 特性了。因此，我们决定将《Java 核心技术》一书分为两卷。现在你手中的是卷 I，主要介绍 Java 语言的基本概念以及图形用户界面编程的基础知识。卷 II（《Java 核心技术 卷 II：高级特性》）将深入介绍企业级特性和高级图形界面编程，其中会详细讨论如下内容：

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

写书的过程中，难免会有错误和不准确的地方。如果你在本书中发现了错误和不准确的地方，非常希望你能通知我们。当然，我们希望每个问题只被提交一次。我们为此特意创建了一个网站 <http://horstmann.com/corejava>，用于支持 FAQ、更正错误和解决方案。在错误提交页面上，我们希望你先阅读已经提交的错误，因此我们把可以用来提交错误和改进建议的表格放在了页面末尾。如果我们未回答每个问题，或者未及时回复，请不要失望。

我们会认真地阅读所有来信，感谢你的建议使本书将来的版本更清晰、更有指导价值。

本书概览

第 1 章概述与其他编程语言相比 Java 所具有的独特功能。这一章将介绍 Java 语言的设计初衷，以及取得了哪些成功。然后，我们简单了解一下这门语言诞生和发展的历史。

第 2 章介绍如何下载与安装 JDK 和本书的示例代码。我们会指导你编译并运行 3 个典型的 Java 程序（一个控制台应用、一个图形界面应用和一个 applet），这里我们会用到纯 JDK、支持 Java 的文本编辑器和 Java IDE（集成开发环境）等工具。

第 3 章开始讨论 Java 语言。这一章将介绍变量、循环及简单函数等基础知识。如果你是 C 或 C++ 程序员，这一章学起来会很容易，因为这些语言特性的语法本质上与 C 语言相同。如果没有 C 语言背景，比如说你熟悉的是 Visual Basic，则应仔细阅读这一章。

面向对象编程（Object-oriented programming, OOP）是当今主流的编程实践，而 Java 正是一种面向对象编程语言。第 4 章介绍封装，这是面向对象技术的两个基本构建模块之一。这一章还将介绍 Java 用于实现封装的语言机制，也就是类和方法。除了 Java 语言的规则之外，我们还会就如何进行合理的 OOP 设计给出一些建议。这一章最后会介绍奇妙的 javadoc 工具，它可以将代码注释格式化为一组通过超链接关联起来的网页。如果你熟悉 C++ 可以快速浏览这一章。没有面向对象编程背景的程序员在进一步学习 Java 之前，应该花点时间掌握 OOP 概念。

类和封装只是 OOP 的一部分，第 5 章将介绍 OOP 的另一个重要概念，也就是继承。继承使我们可以利用现有的类，并根据自己的需求加以修改。这是基本的 Java 编程技术。Java 的继承机制与 C++ 的非常相似。再强调一次，C++ 程序员只需关注两种语言的不同之处即可。

第 6 章介绍如何使用 Java 的接口。接口可以让我们理解第 5 章的简单继承模型之外的东西。掌握了接口，就可以充分发挥 Java 的面向对象编程能力了。这一章还将介绍 Java 的一个有用的技术特性——内部类，该特性可以使代码更清晰、更简洁。

第 7 章，我们将真正开始应用程序设计。每个 Java 程序员都应该懂一点 GUI 编程知识，而本书就提供了相应的基础。这一章将阐述如何创建窗口，如何在窗口中绘图，如何绘制几何图形，如何用多种字体格式化文本，以及如何显示图像。

第 8 章详细讨论抽象窗口工具包 (AWT) 的事件模型。这一章将介绍如何编写代码来响应鼠标点击或键盘敲击等事件，同时还将介绍如何处理按钮和面板等基本的 GUI 元素。

第 9 章将非常详细地讨论 Swing GUI 工具包。我们可以利用 Swing 工具包构建跨平台的图形用户界面。这一章将介绍与各种按钮、文本组件、边框、滑块、列表框、菜单和对话框等控件有关的所有信息。不过一些更高级的组件将在卷 II 中讨论。

第 10 章介绍如何部署自己编写的应用程序或 applet。这一章将描述如何将程序打包到 JAR 文件中，以及如何使用 Java Web Start 和 applet 机制在 Internet 上发布应用程序。这一章最后会解释 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。良好的程序也会出现问题，而异常处理就是 Java 用来处理这种状况的健壮机制。异常提供了一种将正常处理代码与错误处理代码隔离的有效手段。当然，即使程序处理了所有异常条件，仍然有可能无法按照预期方式工作。这一章后半部分会给出大量有用的调试技巧。最后将指导你完成一个示例调试会话。

第 12 章将简要地介绍一下泛型编程，这是 Java SE 5.0 引入的一项重要改进。泛型编程改进了程序的可读性和安全性。这一章将介绍如何使用强类型，并去掉难看且不安全的强制类型转换，还将介绍如何处理为兼容旧版本的 Java 而带来的复杂问题。

第 13 章的主题是 Java 平台的集合框架。当需要将多个对象收集到一起，并在过后加以检索时，应该选择最合适的集合，而不是将所有元素一股脑儿地扔到数组中。这一章将阐述如何使用 Java 预先建立好的标准集合。

第 14 章是本书的最后一章，讨论多线程。借助多线程，可以编程让多个任务并行执行。(线程是程序中的控制流。)这一章将介绍如何创建线程，以及如何处理线程同步。在 Java SE 5.0 中，多线程机制有了很大改进，这一章将介绍新机制相关的所有信息。

附录列出了 Java 语言的保留字。

约定

如同很多计算机图书一样，我们使用等宽字体表示计算机代码。



注意 (NOTE): 注意会使用这样的“注意”图标进行标记。



提示 (TIP): 提示会使用这样的“提示”图标进行标记。



警告 (CAUTION): 如果存在危险，我们会使用“警告”图标警告你。



C++注意 (C++ NOTE): 用于解释 Java 和 C++的不同，如果没有 C++ 背景，或者虽有相关背景，但却视如梦魇，也可以跳过。

Java 提供了一个大型的编程库，即 API。在第一次使用某个 API 调用时，我们会在该节末尾添加一个简短的描述，这些描述不那么正式，但希望它们能比官方的联机 API 文档更具指导性。接口名用斜体表示。类名、接口名或方法名后的数字表示引入该特性的 JDK 版本。

Application Programming Interface 1.2

程序的源代码已包含在本书的配套网站中，它们会以代码清单的形式列出。例如：

Listing 1.1 InputTest/InputTest.java

示例代码

本书的配套网站 <http://horstmann.com/corejava> 提供了书中所有示例代码，代码是以压缩文件格式提供的。你既可以使用常用的解压程序解压，也可以使用 JDK 提供的 `jar` 工具处理。至于如何安装 JDK，以及如何使用示例代码，详见第 2 章。

Acknowledgments

Writing a book is always a monumental effort, and rewriting it doesn't seem to be much easier, especially with the continuous change in Java technology. Making a book a reality takes many dedicated people, and it is my great pleasure to acknowledge the contributions of the entire Core Java team.

A large number of individuals at Prentice Hall provided valuable assistance but managed to stay behind the scenes. I'd like them all to know how much I appreciate their efforts. As always, my warm thanks go to my editor, Greg Doench, for steering the book through the writing and production process, and for allowing me to be blissfully unaware of the existence of all those folks behind the scenes. I am very grateful to Julie Nahil for production support, and to Dmitry Kirsanov and Alina Kirsanova for copyediting and typesetting the manuscript. My thanks also to my coauthor of earlier editions, Gary Cornell, who has since moved on to other ventures.

Thanks to the many readers of earlier editions who reported embarrassing errors and made lots of thoughtful suggestions for improvement. I am particularly grateful to the excellent reviewing team that went over the manuscript with an amazing eye for detail and saved me from many more embarrassing errors.

Reviewers of this and earlier editions include Chuck Allison (Utah Valley University), 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 (Manhattan College), Rakesh Dhoopar (Oracle), David Geary (Clarity Training), Jim Gish (Oracle), Brian Goetz (Oracle), Angela Gordon, Dan Gordon (Electric Cloud), Rob Gordon, John Gray (University of Hartford), Cameron Gregory (olabs.com), Marty Hall (coreservlets.com, Inc.), Vincent Hardy (Adobe Systems), Dan Harkey (San Jose State University), 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 University), Mark Lawrence, Doug Lea (SUNY Oswego), Gregory Longshore, Bob Lynch (Lynch Associates), Philip Milne (consultant), Mark Morrissey (The Oregon Graduate Institute), Mahesh Neelakanta (Florida Atlantic University), Hao Pham, Paul Philion, Blake Ragsdell, Stuart Reges (University of Arizona), Rich Rosen (Interactive Data Corporation), Peter Sanders (ESSI University, Nice, France), Dr. Paul Sanghera (San Jose State University and

Brooks College), 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 Tyma (consultant), Peter van der Linden (Motorola Mobile Devices), Burt Walsh, Dan Xu (Oracle), and John Zavgren (Oracle).

*Cay Horstmann
San Francisco, California
September 2012*

Contents

Preface xix

Acknowledgments xxv

Chapter 1: An Introduction to Java 1

1.1 Java As a Programming Platform 1

1.2 The Java “White Paper” Buzzwords 2

1.2.1 Simple 3

1.2.2 Object-Oriented 4

1.2.3 Network-Savvy 4

1.2.4 Robust 5

1.2.5 Secure 5

1.2.6 Architecture-Neutral 6

1.2.7 Portable 7

1.2.8 Interpreted 7

1.2.9 High-Performance 8

1.2.10 Multithreaded 8

1.2.11 Dynamic 8

1.3 Java Applets and the Internet 9

1.4 A Short History of Java 10

1.5 Common Misconceptions about Java 13

Chapter 2: The Java Programming Environment 17

2.1 Installing the Java Development Kit 18

2.1.1 Downloading the JDK 18

2.1.2 Setting the Executable Path 20

2.1.3 Installing the Library Source and Documentation 22

2.1.4 Installing the *Core Java* Program Examples 23

2.1.5 Navigating the Java Directories 24

2.2 Choosing a Development Environment 24

2.3 Using the Command-Line Tools 25

2.3.1	Troubleshooting Hints	27
2.4	Using an Integrated Development Environment	28
2.4.1	Locating Compilation Errors	32
2.5	Running a Graphical Application	33
2.6	Building and Running Applets	36
Chapter 3: Fundamental Programming Structures in Java		41
3.1	A Simple Java Program	42
3.2	Comments	45
3.3	Data Types	46
3.3.1	Integer Types	47
3.3.2	Floating-Point Types	48
3.3.3	The char Type	49
3.3.4	The boolean Type	51
3.4	Variables	52
3.4.1	Initializing Variables	53
3.4.2	Constants	54
3.5	Operators	55
3.5.1	Increment and Decrement Operators	56
3.5.2	Relational and boolean Operators	57
3.5.3	Bitwise Operators	58
3.5.4	Mathematical Functions and Constants	59
3.5.5	Conversions between Numeric Types	60
3.5.6	Casts	61
3.5.7	Parentheses and Operator Hierarchy	62
3.5.8	Enumerated Types	63
3.6	Strings	64
3.6.1	Substrings	64
3.6.2	Concatenation	64
3.6.3	Strings Are Immutable	65
3.6.4	Testing Strings for Equality	67
3.6.5	Empty and Null Strings	68
3.6.5	Code Points and Code Units	68
3.6.6	The String API	69
3.6.7	Reading the Online API Documentation	72
3.6.8	Building Strings	74

3.7	Input and Output	76
3.7.1	Reading Input	76
3.7.2	Formatting Output	79
3.7.3	File Input and Output	84
3.8	Control Flow	86
3.8.1	Block Scope	86
3.8.2	Conditional Statements	87
3.8.3	Loops	91
3.8.4	Determinate Loops	95
3.8.5	Multiple Selections—The switch Statement	99
3.8.6	Statements That Break Control Flow	102
3.9	Big Numbers	105
3.10	Arrays	107
3.10.1	The “for each” Loop	109
3.10.2	Array Initializers and Anonymous Arrays	110
3.10.3	Array Copying	111
3.10.4	Command-Line Parameters	112
3.10.5	Array Sorting	113
3.10.6	Multidimensional Arrays	116
3.10.7	Ragged Arrays	120
Chapter 4:	Objects and Classes	125
4.1	Introduction to Object-Oriented Programming	126
4.1.1	Classes	127
4.1.2	Objects	128
4.1.3	Identifying Classes	129
4.1.4	Relationships between Classes	129
4.2	Using Predefined Classes	132
4.2.1	Objects and Object Variables	132
4.2.2	The <code>GregorianCalendar</code> Class of the Java Library	136
4.2.3	Mutator and Accessor Methods	138
4.3	Defining Your Own Classes	145
4.3.1	An <code>Employee</code> Class	145
4.3.2	Use of Multiple Source Files	148
4.3.3	Dissecting the <code>Employee</code> Class	149
4.3.4	First Steps with Constructors	150

4.3.5	Implicit and Explicit Parameters	152
4.3.6	Benefits of Encapsulation	153
4.3.7	Class-Based Access Privileges	156
4.3.8	Private Methods	156
4.3.9	Final Instance Fields	157
4.4	Static Fields and Methods	157
4.4.1	Static Fields	157
4.4.2	Static Constants	158
4.4.3	Static Methods	159
4.4.4	Factory Methods	161
4.4.5	The <code>main</code> Method	161
4.5	Method Parameters	164
4.6	Object Construction	171
4.6.1	Overloading	171
4.6.2	Default Field Initialization	172
4.6.3	The Constructor with No Arguments	173
4.6.4	Explicit Field Initialization	174
4.6.5	Parameter Names	175
4.6.6	Calling Another Constructor	176
4.6.7	Initialization Blocks	176
4.6.8	Object Destruction and the <code>finalize</code> Method	181
4.7	Packages	182
4.7.1	Class Importation	182
4.7.2	Static Imports	184
4.7.3	Addition of a Class into a Package	185
4.7.4	Package Scope	188
4.8	The Class Path	190
4.8.1	Setting the Class Path	192
4.9	Documentation Comments	193
4.9.1	Comment Insertion	193
4.9.2	Class Comments	194
4.9.3	Method Comments	195
4.9.4	Field Comments	196
4.9.5	General Comments	196
4.9.6	Package and Overview Comments	197

4.9.7	Comment Extraction	198
4.10	Class Design Hints	199
Chapter 5: Inheritance		203
5.1	Classes, Superclasses, and Subclasses	204
5.1.1	Inheritance Hierarchies	212
5.1.2	Polymorphism	213
5.1.3	Dynamic Binding	214
5.1.4	Preventing Inheritance: Final Classes and Methods	217
5.1.5	Casting	218
5.1.6	Abstract Classes	221
5.1.7	Protected Access	227
5.2	Object: The Cosmic Superclass	228
5.2.1	The equals Method	228
5.2.2	Equality Testing and Inheritance	230
5.2.3	The hashCode Method	234
5.2.4	The toString Method	236
5.3	Generic Array Lists	243
5.3.1	Accessing Array List Elements	246
5.3.2	Compatibility between Typed and Raw Array Lists	249
5.4	Object Wrappers and Autoboxing	251
5.5	Methods with a Variable Number of Parameters	254
5.6	Enumeration Classes	256
5.7	Reflection	258
5.7.1	The Class Class	259
5.7.2	A Primer on Catching Exceptions	261
5.7.3	Using Reflection to Analyze the Capabilities of Classes	263
5.7.4	Using Reflection to Analyze Objects at Runtime	269
5.7.5	Using Reflection to Write Generic Array Code	274
5.7.6	Invoking Arbitrary Methods	278
5.8	Design Hints for Inheritance	282
Chapter 6: Interfaces and Inner Classes		285
6.1	Interfaces	286
6.1.1	Properties of Interfaces	292
6.1.2	Interfaces and Abstract Classes	294