

# Java 核心技术

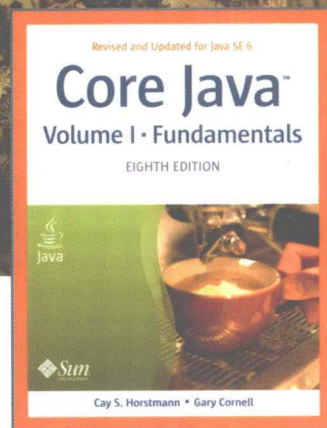
## 卷I: 基础知识

(第8版)

[美] Cay S. Horstmann 著  
Gary Cornell



Core Java™, Volume I-Fundamentals



# Java核心技术

卷 I：基础知识（第8版）（评注版）

Core Java™, Volume I-Fundamentals, 8E

[美] Cay S. Horstmann 著  
Gary Cornell

公飞 评注

电子工业出版社

Publishing House of Electronics Industry

北京•BEIJING

## 内 容 简 介

《Java 核心技术》(Core Java)自第1版出版以来,一直备受广大Java程序设计人员的青睐,畅销不衰,是Java经典书籍。第8版针对Java SE 6平台进行了全面更新,囊括了Java平台标准版(Java SE/J2SE)的全部基础知识,提供了大量完整且具有实际意义的应用实例,详细介绍了Java语言基础知识、面向对象程序设计、接口与内部类、事件监听器模型、swing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、泛型程序设计、集合框架、多线程等内容。

本评注版力邀国内资深专家执笔,在英文原著基础上增加了点评和注释,并对原书内容进行了精简,删除了Java图形窗口编程的相关内容,即原书的第7~10章,同时对章节序号进行了相应的调整,从而使本书更集中于Java核心语言。

本评注版的目的在于以先行者的学研心得与实践感悟,对读者的阅读和学习加以点拨、指明捷径。

书中示例程序经过精心设计,不但具有实用价值,而且易于阅读理解,可以作为初学者自己编写程序的良好开端,也能够帮助程序员快速地了解Java SE 6的新特性,或迅速从其他语言转向Java语言。

Authorized Adaptation from the English language edition, entitled CORE JAVA., VOLUME I—FUNDAMENTALS, 8E, 9780132354769 by HORSTMANN, CAY S.; CORNELL, GARY, published by Pearson Education, Inc, publishing as Prentice Hall, Copyright © 2008 Sun Microsystems, Inc.

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.

ENGLISH language adaptation edition published by PEARSON EDUCATION ASIA LTD. And PUBLISHING HOUSE OF ELECTRONICS INDUSTRY Copyright ©2011 ENGLISH language adaptation edition is manufactured in the People's Republic of China, and is authorized for sale only in the People's Republic of China excluding Hong Kong and Macau.

本书影印改编版由Pearson Education 培生教育出版亚洲有限公司授予电子工业出版社出版。专有出版权受法律保护。

本书影印改编版在中国大陆地区生产,仅限于在中国大陆地区销售。

本书影印改编版贴有Pearson Education 培生教育出版集团激光防伪标签,无标签者不得销售。

版权贸易合同登记号: 图字: 01-2011-1888

### 图书在版编目(CIP)数据

Java 核心技术 = Core Java™, Volume I-Fundamentals, 8E: 评注版. 第1卷, 基础知识: 第8版 / (美)霍斯特曼(Horstmann, C.S.), (美)科内尔(Cornell, G.)著; 公飞评注. —北京: 电子工业出版社, 2011.6

(传世经典书丛)

ISBN 978-7-121-13362-6

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

中国版本图书馆CIP数据核字(2011)第073086号

策划编辑: 张春雨

责任编辑: 李利健

印 刷: 三河市鑫金马印装有限公司

装 订:

出版发行: 电子工业出版社

北京市海淀区万寿路173信箱 邮编 100036

开 本: 787×980 1/16 印张: 40.5 字数: 972千字

印 次: 2011年6月第1次印刷

定 价: 99.00元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zlt@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。

# 悦读上品 得乎益友

孔子云：“取乎其上，得乎其中；取乎其中，得乎其下；取乎其下，则无所得矣”。

对于读书求知而言，这句古训教我们去读好书，最好是好书中的上品——经典书。其中，科技人员要读的技术书，因为直接关乎客观是非与生产效率，阅读选材本更应慎重。然而，随着技术图书品种的日益丰富，发现经典书越来越难，尤其对于涉世尚浅的新读者，更为不易，而他们又往往是最需要阅读、提升的重要群体。

所谓经典书，或说上品，是指选材精良、内容精练、讲述生动、外延丰盈、表现手法体贴入微的读品，它们会成为读者的知识和经验库中的重要组成部分，并且拥有从不断重读中汲取养分的空间。因此，选择阅读上品的问题便成了有效阅读的首要问题。当然，这不只是效率问题，上品促成的既是对某一种技术、思想的真正理解和掌握，同时又是一种感悟或享受，是一种愉悦。

与技术本身类似，经典 IT 技术书多来自国外。深厚的积累、良好的写作氛围，使一批大师为全球技术学习者留下了璀璨的智慧瑰宝。就在那个年代即将远去之时，无须回眸，也能感受到这一部部厚重而深邃的经典著作，在造福无数读者后从未蒙尘的熠熠光辉。而这些凝结众多当今国内技术中坚美妙记忆与绝佳体验的技术图书，虽然尚在国外图书市场上大放异彩，却已逐渐淡出国人的视线。最为遗憾的是，迟迟未有可以填补空缺的新书问世。而无可替代，不正是经典书被奉为主臬的原因？

为了不让国内读者，尤其是即将步入技术生涯的新一代读者，就此错失这些滋养过先行者们的好书，以出版 IT 精品图书，满足技术人群需求为己任的我们，愿意承担这一使命。本次机遇惠顾了我们，让我们有机会携手权威的 Pearson 公司，精心推出“传世经典书丛”。

在我们眼中，“传世经典”的价值首先在于——既适合喜爱科技图书的读者，也符合专家们挑剔的标准。幸运的是，我们的确找到了这些堪称上品的佳作。丛书带给我们的幸运颇多，细数一下吧。

### 得以引荐大师著作

有恐思虑不周，我们大量参考了国外权威机构和网站的评选结果，并得到了 Pearson 的专业支持，又进

一步对符合标准之图书的国内外口碑与销售情况进行细致分析,也听取了国内技术专家的宝贵建议,才有幸选出对国内读者最富有技术养分的大师上品。

### 向深邃的技术内涵致敬

中外技术环境存在差异,很多享誉国外的好书未必适用于国内读者;且技术与应用瞬息万变,很容易让人心生迷惘或疲于奔命。本丛书的图书遴选,注重打好思考方法与技术理念的根基,旨在帮助读者修炼内功,提升境界,将技术真正融入个人知识体系,从而可以一通百通,从容面对随时涌现的技术变化。

### 翻译与评注的双项选择

引进优秀外版著作,将其翻译为中文供国内读者阅读,较为有效与常见。但另有一些外语水平较高、喜好阅读原版的读者,苦于对技术理解不足,不能充分体会原文表述的精妙,需要有人指导与点拨。而一批本土技术精英经过长期经典熏陶及实践锤炼,已足以胜任这一工作。有鉴于此,本丛书在翻译版的同时推出融合英文原著与中文点评、注释的评注版,供不同志趣的读者自由选择。

### 承蒙国内一流译(注)者的扶持

优秀的英文原著最终转化为真正的上品,尚需跨越翻译鸿沟,外版图书的翻译质量一直屡遭国内读者诟病。评注版的增值与含金量,同样依赖于评注者的高卓才具。好在,本丛书得到了久经考验的权威译(注)者的认可和支持,首肯我们选用其佳作,或亲自参与评注工作。正是他们的参与保证了经典的品质,既再次为我们的选材把关,更提供了一流的中文表述。

### 期望带给读者良好的阅读体验

一本好书带给人的愉悦不止于知识收获,良好的阅读感受同样不可缺少,且对学业不无助益。为让读者收获与上品相称的体验,我们在图书装帧设计与选材用料上同样不敢轻率,惟愿送到读者手中的除了珠玑章句,还有舒适与熨帖的视觉感受。

所有参与丛书出版的人员,尽管能力有限,却无不心怀严谨之心与完美愿望。如果读者朋友能从潜心阅读这些上品中偶有获益,不啻为对我们工作的最佳褒奖。若有阅读感悟,敬请拨冗告知,以鼓励我们继续在这一道路上贡献绵薄之力。如有不周之处,也请不吝指教。

电子工业出版社博文视点

# 评注者序



能有机会成为这本 Java 经典之作的评注者，实乃幸事。刚好我也曾是本书第 5 版的读者，还曾不止一次向 Java 新人推荐此书，今天重读本书倍感亲切。而今多年已过，当年研习 Java 的场景仍历历在目，然世事变迁，Java 已经历了多个版本，本书也已更新至第 8 版，甚至还有日落西山、Java 易主 Oracle 的事件发生。

## 成功的 Java

回首 Java 历史，颇具戏剧色彩。虽有为人做嫁衣的 Sun、低效率的 Swing、笨重的 EJB 等不尽如人意之处，然而从整体上说，Java 的巨大成功无可辩驳。在 Tiobe 的程序编程语言排名中，除了曾经在很短一段时间里被 C 语言超过外，Java 已连续多年稳居第一位。在国内也有 45% 的软件开发人员在使用 Java，并且除微软外的绝大多数 IT 巨头们都选择了支持 Java。

Java 的成功既有技术的原因，也有非技术的原因，例如：一次编译到处运行；开放与 Sun 控制之间的平衡（开放使其具有不断创新与前进的动力，而 Sun 的控制又避免了因多方利益无法一致、难以妥协而裹步不前的问题出现）；开源运动的推动（大量基于 Java 的框架、工具等为 Java 的发展提供了强大的推动力量）；IBM、Oracle、SAP 等大量 IT 巨头的支持。

十年前就有人断言 Java 已死，然而 Java 非但没死，反而一直如日中天，

呈现出旺盛的生命力。正是因为 Java 的巨大成功，IT 巨头们在 Java 上已做了大量的投资，Java 过去没有死掉，现在如日中天，将来也会前途光明。

## 去除浮躁学 Java

评注期间，我在网上浏览一些信息时发现关于 Java 与 C#（或基于 Java 开发与基于微软平台开发）对比的讨论依然火热，其中不乏很多带有感情色彩而又片面的言论。

我早年也曾基于微软平台做过开发工作，VB、VC、ASP、PB、Delphi 等都曾用过。其实 Java 和微软都是在相互学习的过程中发展而来的，正所谓“你中有我，我中有你”。例如，JDBC 与 ODBC、JSP 与 ASP、C#与 Java 都有相似性，.NET 也借鉴了 Java JVM 的架构模式。单从技术上说，除了 C#对操作系统的限制，Java 和 C#都可以实现相同的东西。

Java 阵营与微软的差别主要在于产品理念。RAD（Rapid Application Develop）是微软产品制胜的法宝，以让开发者快速上手为目标，更倾向于让开发者傻瓜化，无须开发者关注架构，而且微软提供的一切还独此一家。Java 开发阵营则更显缤纷多彩，而且特别注重架构的合理性，平台、架构、框架等存在大量的选择，开发资源丰富。基于 Java 开发，会“被强化训练”，从而可能促进读者对底层核心技术的理解，在架构设计等方面的能力更能得到锻炼。从这个意义上说，学习 Java 更容易练成高手。

都是在相互借鉴学习，所以完全没有必要对立起来看。对我们说，需要的是理解精髓，融会贯通。前些天刚好在网上看到一篇名为《年薪 217 万技术牛人牛新庄的成长历程》的文章，其中有一个 16 字的要诀“去除浮躁，认真学习，不断积累，寻找机遇”。借用其“去除浮躁”的说法，“停止 Java 和 C#孰优孰劣的浮躁比较，上路学习吧”。而且要想成为高手，必须静下心来，仔细研读类似本书这样大部头的经典之作。

## 本书的学习与 Java 高手的修炼

要想成为武林高手，如果没有基本功和内功作为基础，任何套路招数都将成为浮云。而且一旦练就了扎实的基本功和内功，任何套路招数都可一点即通。本书正是成为 Java 高手、练就基本功和内功的经典之作。其一，本书专注于 Java 语言核心（即 Core Java），这是最基本和最关键之处，可称为 Java 基本功和内功的精髓；其二，本书内容由浅入深，介绍全面、细致、准

确，语言风格朴实易懂。对初学者来说，可以轻松阅读本书，并深入理解和掌握 Java。对于已初步掌握 Java 语言的读者来说，则可以通过本书的学习提升自己的 Java 功底。

下面借本书简单谈谈我对 Java 学习要点的理解，也算是对本书内容的一个导读：

1. 第 1 章的 Java 概要介绍包含了富有启发意义的创新思想和方法。第 3、7 章的 Java 基础编程结构和异常处理机制当然是成就 Java 高手的基本功之一，自不必说。第 8~10 章属于稍微高级一些的内容，不再属于基本功和内功之列。其中，第 10 章的多线程是现代程序设计中非常重要的内容，对网络服务端程序的开发尤其具有重要的意义，而且多线程编程通常比较复杂，需要仔细理解。

2. 第 2 章的 Java 代码编写、编译、运行机制是成就 Java 高手的基本功之一，也是重要的内功之一，应深入掌握。

3. 面向对象是成为 Java 高手的最高层级内功。第 4~6 章正对应于这部分内容，一定要深刻领会。

虽称为评注者，然而本人才疏学浅，不可与脂砚斋、金圣叹等同日而语。仅希望能通过评注这种形式与 Java 初学者形成一种交流，通过对 Java 学习要点的一些提示和解释，以及对原书内容的一些补充，让 Java 初学者能够深入理解并快速掌握 Java。希望本书能够对你学习 Java 带来帮助。

另外要说明一下，因目前 Java Swing 应用较少，我们为了更集中介绍 Java 核心语言，本评注版对原书内容进行了精简，删除了原书第 7~10 章，并对原书的章节序号进行了相应的调整。也即是说，本评注版中的第 7~10 章是原书的第 11~14 章。而且，由于本书篇幅较大，为了节省成本和便于读者对照原书阅读，本评注版也标出了原书对应的页码，本书的索引所列页码（包括正文的页码引用）为原英文版页码。

公飞

2011 年 3 月



# Preface



## To the Reader

In late 1995, the Java programming language burst onto the Internet scene and gained instant celebrity status. The promise of Java technology was that it would become the *universal glue* that connects users with information, whether that information comes from web servers, databases, information providers, or any other imaginable source. Indeed, Java is in a unique position to fulfill this promise. It is an extremely solidly engineered language that has gained acceptance by all major vendors, except for Microsoft. Its built-in security and safety features are reassuring both to programmers and to the users of Java programs. Java even has built-in support that makes advanced programming tasks, such as network programming, database connectivity, and multithreading, straightforward.

Java可以从狭义和广义两个角度来理解：

狭义的Java是指Java语言，它与C++等一样属于编程语言的一种，本书的核心即为Java语言。

广义的Java则是指从Java语言到Java EE (J2EE) 等所构成的一个软件开发体系。本段所描述的“通过信息无缝连接用户，而无论这些信息是来自Web服务器、数据库、信息提供者，还是任何其他渠道。”就是指广义的Java。Java并不是唯一致力于实现此愿景的解决方案，CORBA、EAI、SOA等都是实现这一愿景的架构方案，而且Java也对CORBA、EAI、SOA等架构方案提供了支持，只是Java并不是唯一支持CORBA、EAI、SOA这些架构的解决方案。

当然，狭义的Java语言正是广义Java软件体系的基础。

如今，Java已得到除Microsoft之外的所有厂家的认可。不过，在历史上，微软也

曾支持过Java，在比较早的IE版本中，默认情况下使用的就是微软的Java虚拟机。

在此提及了Java学习的几个高级主题：网络编程、数据库连接、多线程。

Since 1995, Sun Microsystems has released seven major revisions of the Java Development Kit. Over the course of the last eleven years, the Application Programming Interface (API) has grown from about 200 to over 3,000 classes. The API now spans such diverse areas as user interface construction, database management, internationalization, security, and XML processing.

由于Java的开放性和它的快速发展，Java的各种接口、框架已经发展到了泛滥的地步，而且即使只有3 000个类，恐怕也不是Java初学者所能快速接受的。所以，对于Java初学者来说，建议从最基础的内容开始，按照循序渐进的方式进行学习，例如：第一步学习Java程序语言最基本的内容（即本书中的内容）；第二步学习最基础的Servlet和JSP，不要使用任何Web框架；第三步根据自己工作和学习的需要，有选择地学习所需的框架、类库等。

The book you have in your hands is the first volume of the eighth edition of *Core Java™*. With the publishing of each edition, the book followed the release of the Java Development Kit as quickly as possible, and each time, we rewrote the book to take advantage of the newest Java features. This edition has been updated to reflect the features of Java Standard Edition (SE) 6.

As with the previous editions of this book, we *still target serious programmers who want to put Java to work on real projects*. We think of you, our reader, as a programmer with a solid background in a programming language other than Java, and we assume that you don't like books filled with toy examples (such as toasters, zoo animals, or "nervous text"). You won't find any of these in this book. Our goal is to enable you to fully understand the Java language and library, not to give you an illusion of understanding.

In this book you will find lots of sample code that demonstrates almost every language and library feature that we discuss. We keep the sample programs purposefully simple to focus on the major points, but, for the most part, they aren't fake and they don't cut corners. They should make good starting points for your own code.

We assume you are willing, even eager, to learn about all the advanced features that Java puts at your disposal. For example, we give you a detailed treatment of:

- Object-oriented programming
- Reflection and proxies
- Interfaces and inner classes
- The event listener model
- Graphical user interface design with the Swing UI toolkit
- Exception handling
- Generic programming
- The collections framework
- Concurrency

此处所列包含了学习Java程序语言需要重点理解的一些内容：面向对象程序设

计、反射和代理、接口和内部类、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、范型程序设计、集合框架、并行操作。尤其是面向对象程序设计和接口是特别重要的内容，它们是实现其他几个方面的基础。

With the explosive growth of the Java class library, a one-volume treatment of all the features of Java that serious programmers need to know is no longer possible. Hence, we decided to break up the book into two volumes. The first volume, which you hold in your hands, concentrates on the fundamental concepts of the Java language, along with the basics of user-interface programming. The second volume, *Core Java™, Volume II—Advanced Features* (forthcoming, ISBN: 978-0-13-235479-0), goes further into the enterprise features and advanced user-interface programming. It includes detailed discussions of:

- Files and streams
- Distributed objects
- Databases
- Advanced GUI components
- Native methods
- XML processing
- Network programming
- Advanced graphics
- Internationalization
- JavaBeans
- Annotations

上面几方面的内容是第二卷中非常重要并且常用的，事实上，也是非常基础的内容。建议读者在学习完第一卷内容后进一步学习：数据库、XML处理、网络编程、JavaBeans。

In this edition, we reshuffled the contents of the two volumes. In particular, multi-threading is now covered in Volume I because it has become so important, with Moore's law coming to an end.

When writing a book, errors and inaccuracies are inevitable. We'd very much like to know about them. But, of course, we'd prefer to learn about each of them only once. We have put up a list of frequently asked questions, bugs fixes, and workarounds in a web page at <http://horstmann.com/corejava>. Strategically placed at the end of the errata page

(to encourage you to read through it first) is a form you can use to report bugs and suggest improvements. Please don't be disappointed if we don't answer every query or if we don't get back to you immediately. We do read all e-mail and appreciate your input to make future editions of this book clearer and more informative.

## A Tour of This Book

**Chapter 1** gives an overview of the capabilities of Java that set it apart from other programming languages. We explain what the designers of the language set out to do and to what extent they succeeded. Then, we give a short history of how Java came into being and how it has evolved.


In **Chapter 2**, we tell you how to download and install the JDK and the program examples for this book. Then we guide you through compiling and running three typical Java programs, a console application, a graphical application, and an applet, using the plain JDK, a Java-enabled text editor, and a Java IDE.

**Chapter 3** starts the discussion of the Java language. In this chapter, we cover the basics: variables, loops, and simple functions. If you are a C or C++ programmer, this is smooth sailing because the syntax for these language features is essentially the same as in C. If you come from a non-C background such as Visual Basic, you will want to read this chapter carefully.

Object-oriented programming (OOP) is now in the mainstream of programming practice, and Java is completely object oriented. **Chapter 4** introduces encapsulation, the first of two fundamental building blocks of object orientation, and the Java language mechanism to implement it, that is, classes and methods. In addition to the rules of the Java language, we also give advice on sound OOP design. Finally, we cover the marvelous javadoc tool that formats your code comments as a set of hyperlinked web pages. If you are familiar with C++, then you can browse through this chapter quickly. Programmers coming from a non-object-oriented background should expect to spend some time mastering OOP concepts before going further with Java.

Classes and encapsulation are only one part of the OOP story, and **Chapter 5** introduces the other, namely, *inheritance*. Inheritance lets you take an existing class and modify it according to your needs. This is a fundamental technique for programming in Java. The inheritance mechanism in Java is quite similar to that in C++. Once again, C++ programmers can focus on the differences between the languages.

**Chapter 6** shows you how to use Java's notion of an *interface*. Interfaces let you go beyond the simple inheritance model of Chapter 5. Mastering interfaces allows you to have full access to the power of Java's completely object-oriented approach to programming. We also cover a useful technical feature of Java called *inner classes*. Inner classes help make your code cleaner and more concise.



与本书的观点不同，注解者认为Interface（接口）并不是继承的一种简化模式，虽然Interface最初是出于实现多继承能力的考虑而引入的概念，但是就实际的作用来说，Interface与继承有着完全不同的作用。相对于C++等编程语言来说，Interface的引入可以说是Java语言最重要的一个创新，正是Interface的引入，为Java在编程语言层面实现真正的构件化编程提供了关键性的基础。继承还是面向对象的核心概念，Interface则已经进入面向构件的核心概念。对此，本书会在第6章具体阐述。

**Chapter 7** discusses *exception handling*, Java's robust mechanism to deal with the fact that bad things can happen to good programs. Exceptions give you an efficient way of separating the normal processing code from the error handling. Of course, even after hardening your program by handling all exceptional conditions, it still might fail to work as expected. In the second half of this chapter, we give you a large number of useful debugging tips. Finally, we guide you through a sample debugging session.

**Chapter 8** gives an overview of *generic programming*, a major advance of Java SE 5.0. Generic programming makes your programs easier to read and safer. We show you how you can use strong typing and remove unsightly and unsafe casts, and how you can

deal with the complexities that arise from the need to stay compatible with older versions of Java.

The topic of **Chapter 9** is the *collections framework* of the Java platform. Whenever you want to collect multiple objects and retrieve them later, you will want to use a collection that is best suited for your circumstances, instead of just tossing the elements into an array. This chapter shows you how to take advantage of the standard collections that are prebuilt for your use.

**Chapter 10** finishes the book, with a discussion on multithreading, which enables you to program tasks to be done in parallel. (A thread is a flow of control within a program.) We show you how to set up threads and how to deal with thread synchronization. Multithreading has changed a great deal in Java SE 5.0, and we tell you all about the new mechanisms.

Java的Windows图形界面编程很长一段时间内在市场上并不成功,尤其是相对于微软阵营的可视化开发工具,Java的Swing、AWT等图形界面运行效率低,占用资源多,缺少能与微软抗衡的开发工具。直到Eclipse的SWT诞生后,Java的图形界面编程的市场状况才有所改善。另一方面,虽然Java的诞生是源于图形界面的Applet,但是Java最成功的地方是服务器后端,而不是窗口图形界面。

对于Java学习者来说,如果你不是因为项目需要对基于Swing、AWT的图形界面开发有特别的需求,你甚至可以不学习Java的图形窗口编程。如果你有比较多的图形界面开发的需求,可以学习Eclipse的SWT。当然,一个完整的Java学习过程中,对Swing、ATW等图形界面编程有一个基本的了解仍然是非常有益的。

基于这些考虑,以及限于本书篇幅,本评注版对原书内容进行了精简,删除了Java图形窗口编程的相关内容,即原书第7~10章,并对章序号进行了相应的调整,从而使本书更集中于Java核心语言。需要学习Java图形窗口编程的读者可以在完成本书所述Java核心语言的基础上参考其他资料。

## Conventions

As is common in many computer books, we use monospace type to represent computer code.



NOTE: Notes are tagged with “note” icons that look like this.



TIP: Tips are tagged with the “tip” icon that look like this.



CAUTION: When there is danger ahead, we warn you with a “caution” icon.



C++ NOTE: There are many C++ notes that explain the difference between Java and C++. You can skip over them if you don't have a background in C++ or if you consider your experience with that language a bad dream of which you'd rather not be reminded.



### Application Programming Interface

Java comes with a large programming library or Application Programming Interface (API). When using an API call for the first time, we add a short summary description tagged with an API icon at the end of the section. These descriptions are a bit more informal but, we hope, also a little more informative than those in the official on-line API documentation. We now tag each API note with the version number in which the feature was introduced, to help the readers who don't use the "bleeding edge" version of Java.

Programs whose source code is on the Web are listed as examples, for instance

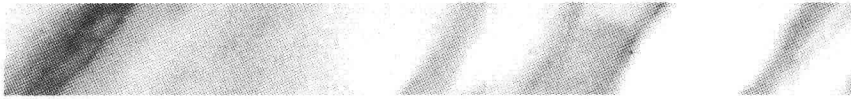
#### Listing 1-1

WelcomeApplet.java

### Sample Code

The web site for this book at <http://horstmann.com/corejava> contains all sample code from the book, in compressed form. You can expand the file either with one of the familiar unzipping programs or simply with the jar utility that is part of the Java Development Kit. See Chapter 2 for more information about installing the Java Development Kit and the sample code.

# Acknowledgments



Writing a book is always a monumental effort, and rewriting doesn't seem to be much easier, especially with 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 and Sun Microsystems Press provided valuable assistance, but they 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 of Prentice Hall, 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 grateful to Vanessa Moore for the excellent production support. 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 (Contributing Editor, *C/C++ Users Journal*), Alec Beaton (PointBase, Inc.), Cliff Berg (iSavvix Corporation), Joshua Bloch (Sun Microsystems), David Brown, Corky Cartwright, Frank Cohen (PushToTest), Chris Crane (devXsolution), Dr. Nicholas J. De Lillo (Manhattan College), Rakesh Dhoopar (Oracle), David Geary (Sabreware), Brian Goetz (Principal Consultant, Quiotix Corp.), Angela Gordon (Sun Microsystems), Dan Gordon (Sun Microsystems), Rob Gordon, John Gray (University of Hartford), Cameron Gregory (olabs.com), Marty Hall (The Johns Hopkins University Applied Physics Lab), Vincent Hardy (Sun Microsystems), Dan Harkey (San Jose State University), William Higgins (IBM), Vladimir Ivanovic (PointBase), Jerry Jackson (ChannelPoint Software), Tim Kimmet (Preview Systems), Chris Laffra, Charlie Lai (Sun

Microsystems), 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 Phillion, 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 (Sun Microsystems), Christopher Taylor, Luke Taylor (Valtech), George Thiruvathukal, Kim Topley (author of *Core JFC*), Janet Traub, Paul Tyma (consultant), Peter van der Linden (Sun Microsystems), and Burt Walsh.

*Cay Horstmann*  
*San Francisco, 2007*



# 目 录



Preface .....	XIX
Acknowledgements .....	XXV
Chapter 1 An Introduction to Java (新增批注共25条) .....	1
Java As a Programming Platform .....	2
The Java “White Paper” Buzzwords .....	3
Simple .....	3
Object Oriented .....	4
Network-Savvy .....	5
Robust .....	5
Secure .....	6
Architecture Neutral .....	7
Portable .....	8
Interpreted .....	9
High Performance .....	9
Multithreaded .....	10
Dynamic .....	10
Java Applets and the Internet .....	11