HZ BOOKS 华章IT

Java领域影响力超群的著作之一，与《Java编程思想》齐名，10余年全球畅销不衰，广受好评

针对Java 11全面更新，系统讲解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 系列

Pearson

Java 核心技术 卷II

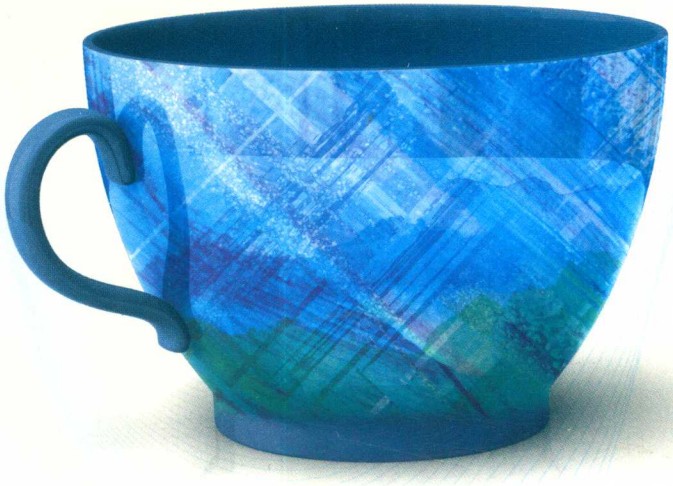
高级特性（原书第11版）

Core Java Volume II—Advanced Features

（Eleventh Edition）

[美] 凯·S. 霍斯特曼（Cay S. Horstmann）著

陈昊鹏 译



机械工业出版社
China Machine Press

Java
核心技术
系列

Java

核心技术 卷II

高级特性（原书第11版）

Core Java Volume II—Advanced Features
（Eleventh Edition）

[美] 凯·S. 霍斯特曼（Cay S. Horstmann）著
陈昊鹏 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Ⅱ　高级特性（原书第 11 版）/（美）凯·S. 霍斯特曼（Cay S. Horstmann）著；陈昊鹏译．—北京：机械工业出版社，2019.12（2020.5 重印）
（Java 核心技术系列）
书名原文：Core Java, Volume II—Advanced Features (Eleventh Edition)

ISBN 978-7-111-64343-2

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

中国版本图书馆 CIP 数据核字（2019）第 268688 号

本书版权登记号：图字　01-2019-2815

Java 核心技术 卷Ⅱ 高级特性（原书第 11 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：殷　虹
印　　刷：北京市荣盛彩色印刷有限公司　　版　　次：2020 年 5 月第 1 版第 3 次印刷
开　　本：186mm×240mm　1/16　　印　　张：43.5
书　　号：ISBN 978-7-111-64343-2　　定　　价：149.00 元

客服电话：（010）88361066　88379833　68326294　　投稿热线：（010）88379604
华章网站：www.hzbook.com　　读者信箱：hzit@hzbook.com

内容简介

本书由拥有20多年教学与研究经验的资深Java技术专家撰写（获Jolt大奖），本版针对Java 11全面更新。

全书共分12章。第1章介绍了Java中的流库；第2章涵盖输入/输出处理，探讨了Java 11中引入的广受欢迎的改进和优化；第3章介绍了XML，展示如何解析XML文件、生成XML和使用XSL转换；第4章讲解了网络API，以及怎样连接到服务器、实现自己的服务器、创建HTTP连接，并讨论了新的HTTP客户端；第5章介绍了数据库编程，重点讲解JDBC，即Java数据库连接API；第6章涵盖如何使用新的日期和时间库来处理日历及时区；第7章讨论国际化；第8章介绍3种处理代码的技术；第9章讲解从Java 9开始引入的Java平台模块系统，以促进Java平台和核心类库的有序演化；第10章继续介绍Java安全模型，展示怎样编写类加载器和安全管理器，以及允许使用消息、代码签名、授权和认证及加密等重要特性的安全API；第11章讨论没有纳入卷 I 的所有Swing知识，包括树形构件、表格构件，以及Java 2D API；第12章介绍本地方法，这个功能支持你调用为微软Windows API这样的特殊机制而编写的各种方法。

作者简介

凯·S. 霍斯特曼

（Cay S. Horstmann ）

圣何塞州立大学计算机科学系教授、Java的倡导者。他是《Java核心技术》两卷本的作者，并著有*Core Java SE 9 for the Impatient, Second Edition*和*Scala for the Impatient, Second Edition*（均由Addison-Wesley出版）。他还为专业程序员和计算机科学专业的学生撰写过数十本其他图书。

封面设计　锡　彬

华章图书

一本打开的书，
一扇开启的门，
通向科学殿堂的阶梯，
托起一流人才的基石。

译 者 序

《Java 核心技术 卷 II 高级特性》(原书第 11 版) 就要出版了！本书在上一版的基础上针对 Java 11 进行了全面修订，除了之前版本中包含的高级 UI 特性、企业编程、网络、安全等主题之外，还新纳入了模块系统等内容，以全面反映 Java 11 的最新特性。从某种程度上说，这本书的版本演进反映了 Java 语言本身的演进过程。虽然在人工智能和大数据的发展风起云涌的大环境下，以 Python 为代表的新兴语言不断涌现并迅速被广为接受，但 Java 在企业级 Web 应用方面的优势还是非常明显，其完备的语言特性和丰富的生态圈使得它在编程语言排行榜上一直位居前列，深受广大程序员的青睐和追捧。可以预见，Java 在未来仍然会有广阔的市场空间，Java 程序员的数量会持续增长，Java 的生态圈也会不断地扩展壮大。

本书在写作上保留了以往的风格，并针对新内容进行了扩展，读者既可以将其作为学习 Java 语言的进阶教材，学习利用 Java 语言编写企业级 Web 应用的高级特性，又可以将其当作 Java 语言的 API 手册，在编程时作为查询 API 细节的工具书。本书提供的所有样例代码几乎都进行了更新，以反映 Java 11 中新的变化。

本书中文版是在之前版本的基础上完成的。在翻译本书的过程中，译者不但对新增加的内容进行了翻译，还对之前版本中存在的错误和不符合中文表达习惯的地方进行了修正，力求以准确和流畅的语言重现英文版的内容和韵味，希望读者在阅读本书时能够感受到 Java 的魅力和作者的风格。

Java 技术在物联网时代还会因其卓越的跨平台能力继续大放异彩，让我们一起拥抱 Java，拥抱新一代互联网技术，让世界变得更智慧！

陈昊鹏

前　言

致读者

本书是按照 Java SE 11 进行更新的。卷 I 主要介绍了 Java 语言的一些关键特性，而本卷主要介绍编程人员进行专业软件开发时需要了解的高级主题。因此，与卷 I 和之前的版本一样，我们仍将本书定位于用 Java 技术进行实际项目开发的编程人员。

编写任何一本书籍都难免会有一些错误或不准确的地方。我们非常乐意听到读者的意见。当然，我们更希望对相同问题的报告只出现一次。为此，我们创建了一个 FAQ、bug 修正以及应急方案的网站 http:// horstmann.com/corejava。你可以在 bug 报告网页的末尾处（鼓励读者阅读以前的报告）添加 bug 报告，以此来发布 bug 和问题并给出建议，以便我们提高本书将来版本的质量。

内容提要

本书中的章节大部分是相互独立的。你可以研究自己最感兴趣的主题，并可以按照任意顺序阅读这些章节。

在**第 1 章**中，你将学习 Java 的流库，它带来了现代风格的数据处理机制，即只需指定想要的结果，而无须详细描述应该如何获得该结果。这使得流库可以专注于优化的计算策略，对于优化并发计算来说，这显得特别有利。

第 2 章的主题是输入 / 输出处理。在 Java 中，所有 I/O 都是通过输入 / 输出流来处理的。这些流（不要与第 1 章的那些流混淆了）使你可以按照统一的方式来处理与各种数据源之间的通信，例如文件、网络连接或内存块。我们对各种读入器和写出器类进行了详细的讨论，它们使得对 Unicode 的处理变得很容易。我们还展示了如何使用对象序列化机制使保存和加载对象变得容易而方便，以及对象序列化机制背后的原理。然后，我们讨论了正则表达式以及如何操作文件与路径。该章通篇都包含了最新的 Java 版本中引入的广受欢迎的改进和优化。

第 3 章介绍 XML，展示怎样解析 XML 文件、怎样生成 XML 以及怎样使用 XSL 转换。在一个实用示例中，我们将展示怎样在 XML 中指定 Swing 窗体的布局。我们还讨论了 XPath API，它使得“在 XML 的干草堆中寻找绣花针”变得更加容易。

第 4 章介绍网络 API。Java 使复杂的网络编程工作变得很容易实现。我们将介绍怎样连接到服务器，怎样实现你自己的服务器，以及怎样创建 HTTP 连接。该章还讨论了新的 HTTP 客户端。

第 5 章介绍数据库编程，重点讲解 JDBC，即 Java 数据库连接 API，这是用于将 Java 程

序与关系数据库进行连接的 API。我们将介绍怎样通过使用 JDBC API 的核心子集，编写能够处理实际的数据库日常操作事务的实用程序。（如果要完整介绍 JDBC API 的功能，可能需要编写一本像本书一样厚的书才行。）最后我们简要介绍了层次数据库，探讨了 JNDI（Java 命名及目录接口）以及 LDAP（轻量级目录访问协议）。

Java 对于处理日期和时间的类库做出过两次设计，而在 Java 8 中做出的第三次设计则极富魅力。在**第 6 章**，你将学习如何使用新的日期和时间库来处理日历和时区的复杂性。

第 7 章讨论一个我们认为其重要性将会不断提升的特性——国际化。Java 编程语言是少数几种一开始就被设计为可以处理 Unicode 的语言之一，不过 Java 平台的国际化支持则走得更远。因此，你可以对 Java 应用程序进行国际化，使其不仅可以跨平台，而且还可以跨国界。例如，我们会展示怎样编写一个使用英语、德语和汉语的退休金计算器。

第 8 章讨论三种处理代码的技术。脚本机制和编译器 API 允许程序去调用使用诸如 JavaScript 或 Groovy 之类的脚本语言编写的代码，并且允许程序去编译 Java 代码。可以使用注解向 Java 程序中添加任意信息（有时称为元数据）。我们将展示注解处理器怎样在源码级别或者在类文件级别上收集注解，以及怎样运用注解来影响运行时的类行为。注解只有在工具的支持下才有用，因此，我们希望这些讨论能够帮助你根据需要选择有用的注解处理工具。

第 9 章讲解从 Java 9 开始引入的 Java 平台模块系统，以促进 Java 平台和核心类库的有序演化。这个模块系统提供了对包的封装和用于描述模块需求的机制。你将学习模块的属性，以便决定是否要在自己的应用程序中使用它们。即使你决定不使用，也需要了解这些新规则，这样你才能和 Java 平台以及其他模块化的类库交互。

第 10 章继续介绍 Java 安全模型。Java 平台一开始就是基于安全而设计的，该章会带你深入内部，查看这种设计是怎样实现的。我们将展示怎样编写用于特殊应用的类加载器和安全管理器。然后介绍允许使用消息、代码签名、授权和认证以及加密等重要特性的安全 API。最后，我们用一个使用 AES 和 RSA 加密算法的示例进行总结。

第 11 章讨论没有纳入卷 I 的所有 Swing 知识，尤其是重要但很复杂的树形构件和表格构件。我们还会介绍 Java 2D API，你可以用它来创建实际的图形和特殊的效果。当然，如今已经没有多少程序员需要编写 Swing 用户界面了，因此我们会将注意力放到在服务器端生成图像的实用特性上。

第 12 章介绍本地方法，这个功能支持你调用为微软 Windows API 这样的特殊机制而编写的各种方法。很显然，这种特性具有争议：使用本地方法，那么 Java 平台的跨平台特性将会随之消失。毫无疑问，每个为特定平台编写 Java 应用程序的专业开发人员都需要了解这些技术。有时，当你与不支持 Java 平台的设备或服务进行交互时，为了你的目标平台，你可能需要求助于操作系统 API。我们将通过展示如何从某个 Java 程序访问 Windows 注册表 API 来阐明这一点。

所有章节都按照最新版本的 Java 进行了修订，过时的材料都删除了，Java 9、10 和 11 的新 API 都详细地进行了讨论。

约定

我们使用等宽字体表示计算机代码，这种格式在众多的计算机书籍中极为常见。各种图标的含义如下：

注释：需要引起注意的地方。

提示：有用的提示。

警告：关于缺陷或危险情况的警告信息。

C++ 注释：本书中有许多这类提示，用于解释 Java 程序设计语言和 C++ 语言之间的不同。如果你对这部分不感兴趣，可以跳过。

Java 平台配备有大量的编程类库或者应用程序编程接口（API）。当第一次使用某个 API 时，我们在每一节的末尾都添加了简短的描述。这些描述可能有点不太规范，但是比官方在线 API 文档更具指导性。接口的名字以斜体显示，就像许多官方文档一样。类、接口或方法名后面的数字是 JDK 的版本，表示在该版本中才引入了相应的特性。

应用程序编程接口

本书示例代码以程序清单的形式列举出来，例如：

程序清单 1-1　ScriptTest.java

可以从网站 `http://horstmann.com/corejava` 下载示例代码。

致　谢

写一本书需要投入大量的精力，升级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。出版一本书会让很多人耗费很多心血，在此衷心地感谢《 Java 核心技术》小组的每一位成员。

Pearson 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，我希望大家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 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，感谢 Dmitry Kirsanov 和 Alina Kirsanova 完成手稿的编辑与排版工作。我还要感谢早期版本中我的合作者 Gary Cornell，他已经转向了其他行业。

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校小组，他们仔细地审阅我的手稿，使本书减少了许多错误。

这一版及以前版本是由以下人员评审的：Chuck Allison（特约编辑，*C/C++ Users Journal*）、Lance Anderson（Oracle）、Alec Beaton（PointBase, Inc.）、Cliff Berg（iSavvix Corporation）、Joshua Bloch、David Brown、Corky Cartwright、Frank Cohen（PushToTest）、Chris Crane（devXsolution）、Nicholas J. De Lillo 博士（曼哈顿学院）、Rakesh Dhoopar（Oracle）、Robert Evans（资深教师，约翰·霍普金斯大学应用物理实验室）、David Geary（Sabreware）、Jim Gish（Oracle）、Brian Goetz（Oracle）、Angela Gordon、Dan Gordon、Rob Gordon、John Gray（Hartford 大学）、Cameron Gregory（olabs.com）、Steve Haines、Marty Hall（约翰·霍普金斯大学应用物理实验室）、Vincent Hardy、Dan Harkey（圣何塞州立大学）、William Higgins（IBM）、Vladimir Ivanovic（PointBase）、Jerry Jackson（ChannelPoint Software）、Tim Kimmet（Preview Systems）、Chris Laffra、Charlie Lai、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（俄勒冈研究生院）、Mahesh Neelakanta（佛罗里达大西洋大学）、Hao Pham、Paul Philion、Blake Ragsdell、Ylber Ramadani（Ryerson 大学）、Stuart Reges（亚利桑那大学）、Simon Ritter、Rich Rosen（Interactive Data Corporation）、Peter Sanders（法国尼斯 ESSI 大学）、Paul Sanghera 博士（圣何塞州立大学和布鲁克斯学院）、Paul Sevinc（Teamup AG）、Yoshiki Shabata、Devang Shah、Richard Slywczak（NASA/Glenn 研究中心）、Bradley A. Smith、Steven Stelting、Christopher Taylor、Luke Taylor（Valtech）、George Thiruvathukal、Kim Topley（*Core JFC*，*Second Edition* 的作者）、Janet Traub、Paul Tyma（顾问）、Christian Ullenboom、Peter van der Linden、Burt Walsh、Joe Wang（Oracle）和 Dan Xu（Oracle）。

Cay Horstmann

2018 年 12 月于加州旧金山

目　录

第 1 章　Java 8 的流库

- ▲ 从迭代到流的操作
- ▲ 流的创建
- ▲ filter、map 和 flatMap 方法
- ▲ 抽取子流和组合流
- ▲ 其他的流转换
- ▲ 简单约简
- ▲ Optional 类型
- ▲ 收集结果
- ▲ 收集到映射表中
- ▲ 群组和分区
- ▲ 下游收集器
- ▲ 约简操作
- ▲ 基本类型流
- ▲ 并行流

与集合相比，流提供了一种可以让我们在更高的概念级别上指定计算任务的数据视图。通过使用流，我们可以说明想要完成什么任务，而不是说明如何去实现它。我们将操作的调度留给具体实现去解决。例如，假设我们想要计算某个属性的平均值，那么我们就可以指定数据源和该属性，然后，流库就可以对计算进行优化，例如，使用多线程来计算总和与个数，并将结果合并。

在本章中，你将会学习如何使用 Java 的流库，它是在 Java 8 中引入的，用来以“做什么而非怎么做”的方式处理集合。

1.1　从迭代到流的操作

在处理集合时，我们通常会迭代遍历它的元素，并在每个元素上执行某项操作。例如，假设我们想要对某本书中的所有长单词进行计数。首先，将所有单词放到一个列表中：

```
var contents = new String(Files.readAllBytes(
   Paths.get("alice.txt")), StandardCharsets.UTF_8); // Read file into string
List<String> words = List.of(contents.split("\\PL+"));
   // Split into words; nonletters are delimiters
```

现在，我们可以迭代它了：

```
int count = 0;
for (String w : words) {
   if (w.length() > 12) count++;
}
```

在使用流时，相同的操作看起来像下面这样：

```
long count = words.stream()
   .filter(w -> w.length() > 12)
   .count();
```