



国外经典教材·计算机科学与技术

PEARSON
Prentice
Hall

Java™ for Programmers

JAVA程序员指南

Douglas A. Lyon 著

朱剑平 汪青青 等译



清华大学出版社

第 2 版



Java™ for Programmers

JAVA 程序员指南

著者: R. S. Lipton 著
译者: 陈建明 等



机械工业出版社

国外经典教材·计算机科学与技术

Java 程序员指南

Douglas A. Lyon 著

朱剑平 汪青青 等 译

清华大学出版社

北 京

出版说明

近年来,我国的高等教育特别是计算机学科教育,进行了一系列大的调整和改革,急需一批门类齐全、具有国际先进水平的计算机经典教材,以适应当前我国计算机科学的教学需要。通过使用国外先进的经典教材,可以了解并吸收国际先进的教学思想和教学方法,使我国的计算机科学教育能够跟上国际计算机教育发展的步伐,从而培育出更多具有国际水准的计算机专业人才,增强我国计算机产业的核心竞争力。为此,我们从国外知名的出版集团 Pearson 引进这套“国外经典教材·计算机科学与技术”教材。

作为全球最大的图书出版机构, Pearson 在高等教育领域有着不凡的表现,其下属的 Prentice Hall 和 Addison Wesley 出版社是全球计算机高等教育的龙头出版机构。清华大学出版社与 Pearson 出版集团长期保持着紧密友好的合作关系,这次引进的“国外经典教材·计算机科学与技术”教材大部分出自 Prentice Hall 和 Addison Wesley 两家出版社。为了组织该套教材的出版,我们在国内聘请了一批知名的专家和教授,成立了一个专门的教材编审委员会。

教材编审委员会的运作从教材的选题阶段即开始启动,各位委员根据国内外高等院校计算机科学及相关专业的现有课程体系,并结合各个专业的培养方向,从 Pearson 出版的计算机系列教材中精心挑选针对性强的题材,以保证该套教材的优秀性和领先性,避免出现“低质重复引进”或“高质消化不良”的现象。

为了保证出版质量,我们为这套教材配备了一批经验丰富的编辑、排版、校对人员,制定了更加严格的出版流程。本套教材的译者,全部来自于对应专业的高校教师或拥有相关经验的 IT 专家。每本教材的责编在翻译伊始,就定期不间断地与该书的译者进行交流与反馈。为了尽可能地保留与发扬教材原著的精华,在经过翻译、排版和传统的三审三校之后,我们还请编审委员或相关的专家教授对文稿进行审读,以最大程度地弥补和修正在前面一系列加工过程中对教材造成的误差和瑕疵。

由于时间紧迫和受全体制作人员自身能力所限,该套教材在出版过程中很可能还存在一些遗憾,欢迎广大师生来电来信批评指正。同时,也欢迎读者朋友积极向我们推荐各类优秀的国外计算机教材,共同为我国高等院校计算机教育事业贡献力量。

清华大学出版社

国外经典教材·计算机科学与技术

编审委员会

主任委员:

孙家广

清华大学教授

副主任委员:

周立柱

清华大学教授

委员(按姓氏笔画排序):

王成山

天津大学教授

王 珊

中国人民大学教授

冯少荣

厦门大学教授

冯全源

西南交通大学教授

刘乐善

华中科技大学教授

刘腾红

中南财经政法大学教授

吉根林

南京师范大学教授

孙吉贵

吉林大学教授

阮秋琦

北京交通大学教授

何 晨

上海交通大学教授

吴百锋

复旦大学教授

李 彤

云南大学教授

杨宗源

华东师范大学教授

沈钧毅

西安交通大学教授

邵志清

华东理工大学教授

陈 纯

浙江大学教授

陈 钟

北京大学教授

陈道蓄

南京大学教授

周伯生

北京航空航天大学教授

孟祥旭

山东大学教授

姚淑珍

北京航空航天大学教授

徐佩霞

中国科学技术大学教授

徐晓飞

哈尔滨工业大学教授

秦小麟

南京航空航天大学教授

钱培德

苏州大学教授

曹元大

北京理工大学教授

龚声蓉

苏州大学教授

谢希仁

中国人民解放军理工大学教授

序

鱼从烂头起。

——谚语

Java 正成为 Internet 上的通用语言，同时也是讲授编程时最常用的语言。Lyon 博士为 Java 所做的贡献，就像 Strunk 先生和 White 先生为英语所做的贡献一样多。他告诉人们怎样来写精练而有效的 Java 程序。和英语一样，Java 也在不断地发展和完善。就像英语中依然还存在一些陈词滥调或性别歧视的代词而令人感到遗憾一样，Lyon 博士偶尔也会为 Java 的一些“饱受争议”的特性而叹息，但他还是为新版本的 Java 的强大功能和表现力感到高兴和愉悦。

注意此书并非简单的语言说明，因为 Lyon 在用 Java 思考问题，甚至做梦都是 Java。Lyon 仿佛是为面向对象而生的，而随着 Java 的出现，他与计算机的姻缘也上升到新的高度。Lyon 行文或优雅大方，或充满激情，或异想天开，始终不变的是清晰与简洁。本书如实地反映了 Java 的全貌，目前，Java 的 API 包含 2 000 多个类、20 000 多个成员（本书以首字母缩略词的形式罗列出来了）。

我了解很多编程语言，也曾教过书，现在又是 Java 的初学者。我以读者的身份从头至尾地拜读了这本极富教学性的书。苏格拉底式的对话，清晰透彻地阐述了如何依据 Java 规范成功地设计决策。富有启发性的正反示例阐明了深奥的 Java 特性。另外，在本书的各章后面均有习题，习题的难度逐渐提高，旨在强化实践能力。

前面几章简要介绍了 Java 的核心，并提供了一些实例，这些实例是我见过的有关面向对象设计方面最清楚、最具说服力的。Lyon 也强调内部文档和外部文档的重要性，因此本书安排了一些章节，透彻地讲解了软件设计工具和 JavaDoc，JavaDoc 可以根据源代码的注释自动生成 HTML 页面。

作为程序员中的程序员，Lyon 尤为擅长有关类型、继承、嵌套类和接口、映射以及静态和动态代理委派等方面的设计。他带领读者进入操作系统（线程）、图形界面（利用 Swing 和 AWT 库生成的图标、字体和图）、数据库链接（SQL 的 JDBC）、网络（浏览器、XML 和电子邮件）和 Web service（servlet）等 Java 微观世界。而大多数其他有关 Java 的书，并没有深入讨论这些问题，而且也没有探讨有关服务器端服务和 JSP 的内容。

经典巨著，无须赘言。请继续读下去，愉快地阅读本书，轻松地编写程序！

George Nagy 教授
仁斯里尔理工学院 美国纽约州特洛伊市
2002 年 10 月 12 日

前 言

莫用斧头从朋友的额前赶苍蝇。

——中国谚语

欢迎阅读《Java 程序员指南》。本书的目标读者是了解编程且对 Java 感兴趣的人。因此，本书只是简明讲述 Java，要求读者对非面向对象编程相当熟悉。

Java 的世界包括缓慢发展的 Java 语言和迅速壮大的库集合以及大量的支持工具。随着库的发展，不断有新的元素加入，而旧元素不断被更新。这些变化会给已有代码带来难以预料的影响。于是，程序员不得不学习保守性编程技术。保守编程意味着需要创建自定义的、稳定的库，可以调用正在变化的库的功能。

大量的工具和库为软件工程解决方案带来了巨大的市场。兴奋的管理者和新手，纷纷涌向书店，购买有关 Java 技术的图书，但结果却往往只是发现他们并不真正了解这些技术（或者根本就不会使用这些技术）。

初学者往往喜欢全面地阅读，因为他们希望学到很多知识，却不要求深入研究。因此，本书介绍了很多 Java 技术，但只是点到为止。要深入介绍几项 Java 技术，就需要整整一本书。本书将这些主题分为几小块，以便于讲述和理解。

《Java 程序员指南》的特色

与其他 Java 图书相比，本书具有以下独特之处。

- 面向具有编程经验的编程人员。很多书往往花大量篇幅来介绍诸如“什么是计算机？”以及“什么是程序？”之类的入门知识。本书则假定读者已了解了基本的概念。
- 全面介绍客户端的和服务端端的 Java。本书把程序员从基本的 Java 编程技术逐步引入高级服务器端编程。
- 不需要有 Java 编程经验，本书也并不要求读者了解 Java。
- 全面讲解模块技术。将语言从库中分离出来单独阐述。
- GST/AFA（Graduate Student Tested/Adjunct Faculty Approved）级别（研究生入学测试水平/准助教级）。
- 面向双重市场：学校和业界。
- 在每章开头介绍学习目标和新的保留词。

一般来说，讲述 Sun API 对 Java 书籍而言已经足够了。然而，《Java 程序员指南》并非如此。《Java 程序员指南》讲述了开源的 DocJava API，这也是本书的创新之处。例如，在 DocJava API 中，你可以通过 Swing API 或 AWT API 打开文件对话框。甚至可以设置

API 来获取使用 Wireless（例如手机）API 的文件，即 MIDP。

Sun 经常淘汰 Java 的公共接口，这会使相关书籍很快就要过时；但我们可以通过创建一个 API 使程序员不受这些变化的干扰，我们可以确信以下内容。

1. 保持本书中 API 的稳定性。
2. 保持本书的热销。
3. 着眼于可靠的软件工程原理，而不是 Sun API 本身。
4. 告诉程序员如何创建自己的工具包，这样，他们就可以不用完全依赖别人。

本书为谁而写？

如果我比别人看得更远，那是因为我站在巨人的肩上。

—艾萨克·牛顿（哲学家、数学家）(1642—1727)

本书专门为已有编程经验的程序员编写。在我看来，学习第一门语言往往是最难的，接下来学习其他语言就容易多了。因此，本书认为 Java 至少是读者学习的第二门计算机语言。

本书讲述 Java 的方式很适合作为课堂教材或作为职业程序员的快速指南。目前，它正用于研究生课程（45 周、总共 135 个课时）。数百名研究生已经学过本书，其内容足够讲授数百个课时。

教学方法

《Java 程序员指南》提供了很多实例、框架、习题和项目。本书讨论了设计模式，但只在必要时才应用它们。比起那些专门的设计模式教程，这种“即时”教学方法可以减轻学生学习设计模式的负担。

延后讲述 API

《Java 程序员指南》是一本精心组织的教材，介绍的概念在逻辑上彼此紧密相联。因此，首先讨论 Java 语言和面向对象设计，然后在必要时介绍了 API。这种模块化的风格为教师的课程安排提供了更大的灵活性。而且，本书并没有像有些书那样错误地首先讲述 applet。

本书清晰、简洁地概述了 Java 编程的方方面面（如类的设计、线程和事件处理）。每一章都涵盖了读者所必须了解的全部主题，并提供了很多实例。

与一般的 Java 书籍不同，本书并非只讲述相关 Java 的主题。相反，本书涉及到很多其他主题，其中大多数是比较超前的。本书提供了很多有用的提示和窍门。

很多书籍都只着眼于一个主题，全篇讲的都是些无关紧要的东西。相比之下，本书则抓住了很多 Java 程序员必须了解的主题。对于大部分主题而言，本书阐述的深度已经足以让读者理解并使用我们所讲述的 Java 技术了。本书还有大量例子和代码。在某些情况下，这些代码甚至可以直接应用于某些程序，有时还会提供一些实用程序，以帮助读者更好地理解 and 运用 Java 技术。

最后，本书还提供了各种 API，使程序员能够更深入地探索 Java 的高级前沿技术。

关于图标

本书全篇使用了几种图标来强调重点。



表示提供有关 Java 语言的语法摘要。



警告

表示可能损害程序或威胁系统安全的情况。



注意

表示特殊情况。



表示有关设计模式的内容。



风格提示

表示如何正确处理 Java 代码的格式和用法。

致 谢

世间万物莫不变幻无常。

——赫拉克利特（公元前 540 年—公元前 480 年）

参与编写这本书的人太多，所以我无法一一列举。如果你的名字被漏掉了，还望你能原谅我的疏忽。

感谢 Hak Kywn Roh 为本书提供了 `ReplaceString` 类，感谢 James Linn 为 `AtomicClock` 类做出的巨大努力。感谢 Job Shen 为自动桥式设计模式代码所做出的杰出贡献。感谢 Naoki Chigai 准许本书使用 `HtmlGenerator`，感谢 Glenn Seseske 准许我们复制 `ComponentEditor`。还要感谢我的许多学生，他们在整理 beta 版代码和手稿上做出了很大的努力。我希望这不会给他们的学习成绩带来太大的影响。

感谢 Allison McHenry 为本书第 6 章所做的评论和建议，还要感谢所有编辑人员，尤其是 Mary Ellen Buschman、Vipul Jagadevsinh Chavda、Maynard Marquis 和 Carl Weiman。

最后，我还要感谢为这本书提出很多宝贵意见的匿名评论家。

这本书耗费了我太多的心血。我现在需要的只是彻底地放松。

关于撰稿人

Douglas Lyon 是 Fairfield University 计算机工程系主任兼 DocJava 公司总裁。Lyon 曾写过 3 本 Java 方面的著作，并在 Jet Propulsion Laboratory 的 Artificial Intelligence Research and Development department（人工智能研发部）和 AT&T Bell Laboratories 工作。Lyon 还是 RAYTEL 公司的首席科学家。自 1984 年以来，Lyon 一直致力于面向对象编程。Lyon 博士的通信地址是：Computer Engineering Department, Fairfield University, North Benson Road, Fairfield, CT 06430-5195 USA。电子邮箱的地址是：lyon@docjava.com；Web 地址是：<http://www.docjava.com>。

Diane Asmus，自 1994 年以来，一直担任 Executone 和 NOTARA 公司的 Visual Basic 和 Java 软件工程师。通信地址是：Fairfield University, North Benson Road, Fairfield, CT 06430-5195 USA；电子邮箱的地址是：DianeJava@Yahoo.com。

Frances Grodzinsky 博士是美国康涅狄格州 Fairfield 的 Sacred Heart University Computer Science and Information Technology（计算机科学与信息技术）教授，她与人合著了 *The Anatomy of Programming Languages*，她还是 *The Networking and Data Communications Laboratory Manual* 的编辑，还发表过几篇有关计算机道德规范的文章。Frances Grodzinsky 博士的电子信箱地址是：grodzinskyf@sacredheart.edu。

James Linn 硕士曾先后供职于 Executone、IPC、Neu Vis 和 Hartford Technology Service 公司。自 1997 年以来，一直从事面向对象的构架和 C++及 Java 的开发工作。他的通信地址是：Computer Engineering Department, Fairfield University, North Benson Road, Fairfield, CT 06430-5195 USA。他的电子邮箱地址是：jim@raptureart.com。

Maynard L. Marquis，理科硕士，Fairfield University 的 School of Engineering（工程研究生院）的教师，为 DocJava 公司提供商业培训，有多年航空航天行业的工程经验。他的通信地址是：Computer Engineering Department, Fairfield University, North Benson Road, Fairfield, CT 06430-5195 USA。他的电子邮箱地址是：maynard_marquis@juno.com。

Allison McHenry，Fairfield University 大学 Software Engineering（软件工程）硕士，也是 INT Media Group 的一名开发人员，自 1996 年以来一直从事与 Internet 相关的技术工作。她的通信地址是：Computer Engineering Department, Fairfield University, North Benson Road, Fairfield, CT 06430-5195 USA。她的电子邮箱地址是：amchenry_fagan@techie.com。

Rodrigo A. Obando 博士是 Fairfield University 的 Charles F. Dolan School of Business 的 Information System and Operation Management Department（信息系统与操作管理系）的助理教授，曾先后供职于 NASA 兰利研究中心和喷气推进实验室，同时还在几家咨询公司工作。现在，他专注于信息可视化和数据挖掘方面的研究。Obando 从 1988 年开始研究面向对象编程。他的通信地址是：Assistant Professor of Information Systems and Operations Management, Charles F. Dolan School of Business, Fairfield University, North Benson Road, Fairfield,

CT 06430-5195 USA。他的电话是：(203)254-4000 转 2830；他的电子邮箱地址是：Robando@fairl.fairfield.edu。

Thomas F. Rowland, Fairfield University 软件工程硕士并获电子工程学士学位。自 1987 年以来, Rowland 一直从事软件和硬件开发。在过去的 5 年中, 一直专注于面向对象软件开发。现在是 DocJava 公司的一名技术人员。他的通信地址是: Computer Engineering Department, Fairfield University, North Benson Road, Fairfield, CT 06430-5195 USA; 他的电子邮箱地址是: tomrowland@yahoo.com。

目 录

第 1 章	Java 基本概念	1
1.1	Java 的历史	1
1.2	什么是 Java	2
1.3	Java 模型与 HTML 模型的比较	3
1.4	小结	4
1.5	练习	4
第 2 章	原始数据类型	5
2.1	Java 中的 8 种原始数据类型	5
2.2	有符号定点数据类型	7
2.3	简单算术表达式	7
2.4	原始数据类型的基本实例	8
2.5	原始数据类型转换	10
2.6	一些数学算法	10
2.7	小结	11
2.8	练习	11
第 3 章	运算符	13
3.1	优先级、结合性以及单目/二目运算符	13
3.2	具有最高优先级的运算符	20
3.3	增量与减量运算符	21
3.4	算术运算符	22
3.5	赋值运算符	23
3.6	运算符的句法规则	24
3.7	自测题	24
3.8	练习	30
第 4 章	基数换算	31
4.1	计数系统	31
4.2	通用计算机计数系统	32
4.3	从二进制、八进制和十六进制换算至十进制	33
4.4	由二进制换算至八进制和十六进制	33
4.5	由十进制换算为二进制、八进制和十六进制	33

4.6 小结	35
4.7 练习	36
第 5 章 控制结构	37
5.1 关系运算符	38
5.2 等于运算符	38
5.3 逻辑运算符	39
5.4 条件运算符	39
5.5 <i>if</i> 语句	40
5.6 <i>while</i> 和 <i>do</i> 语句	43
5.7 <i>for</i> 语句	44
5.8 <i>continue</i> 语句	45
5.9 <i>break</i> 语句	46
5.10 <i>switch</i> 语句	48
5.11 <i>return</i> 语句	50
5.12 练习	51
第 6 章 引用数据类型	53
6.1 类的概念	53
6.2 构造函数	56
6.3 <i>getter</i> 和 <i>setter</i> 方法	57
6.4 关键词 <i>null</i>	58
6.5 子类 and 超类	59
6.6 <i>casting</i>	63
6.7 关键词	64
6.8 接口	65
6.9 以数据为中心同时面向对象的设计	70
6.10 脆弱的基类问题	71
6.11 小结	71
6.12 练习	72
第 7 章 静态、抽象及 <i>final</i> 修饰符	75
7.1 静态方法	75
7.2 静态变量	76
7.3 抽象类和方法	80
7.4 <i>final</i> 修饰符和命名常量	83
7.5 <i>final</i> 类和方法	84
7.6 单件模式	84
7.7 小结	85

7.8 练习	86
第 8 章 数组和向量	88
8.1 数组	88
8.2 向量	98
8.3 构建一辆购物手推车	101
8.4 堆栈	104
8.5 列表	105
8.6 队列	106
8.7 小结	107
8.8 练习	107
第 9 章 <i>wrapper</i> 类和字符串	110
9.1 <i>wrapper</i> 类	110
9.1.1 <i>Boolean</i>	111
9.1.2 <i>Character</i>	111
9.1.3 数值 <i>wrapper</i> 类	112
9.2 <i>String</i>	114
9.3 <i>ReplaceString</i>	117
9.4 <i>StringTokenizer</i>	118
9.5 <i>CsvParser</i>	119
9.6 迷宫程序 (Maze Solver)	121
9.7 小结	124
9.8 练习	124
第 10 章 异常	127
10.1 第 1 个异常	127
10.2 监视输入或捕捉错误	128
10.3 定义自己的异常	129
10.4 检验异常与非检验异常	130
10.5 <i>try</i> 语句的句法	131
10.6 <i>throw</i> 语句	134
10.7 <i>throws</i> 修饰符	135
10.8 <i>System.err</i>	136
10.9 公共安全和异常	136
10.10 小结	137
10.11 练习	138
第 11 章 包、导入以及可见性	139

11.1	包	139
11.2	导入语句	141
11.3	可见性修饰符	143
11.4	小结	145
11.5	练习	145
第 12 章	嵌套类和嵌套接口	146
12.1	成员内部类	147
12.2	<i>LocalInnerClass</i>	149
12.3	匿名内部类	150
12.4	嵌套静态类	154
12.5	嵌套接口	155
12.6	<i>This</i> 关键词	157
12.7	小结	158
12.8	练习	159
第 13 章	线程	160
13.1	并发、多任务和多线程	160
13.2	建立新线程	163
13.3	线程状态	165
13.4	同步化方法和超量的啤酒	166
13.5	线程组	168
13.6	线程管理器	169
13.7	<i>ThreadUtil</i>	171
13.8	<i>ThreadQueue</i>	175
13.9	运行数以千计的线程	177
13.10	<i>Job</i> 线程	178
13.11	小结	179
13.12	练习	180
第 14 章	文件	182
14.1	什么是文件	182
14.2	获取用户文件	183
14.2.1	<i>Futil.getDirFile</i>	183
14.2.2	<i>Futil.getReadFile</i>	183
14.2.3	<i>Futil.getWriteFile</i>	184
14.3	<i>File</i> 类	184
14.3.1	<i>File</i> 类概要	185
14.3.2	<i>File</i> 类的使用	185

14.4	<i>java.io.FilenameFilter</i> 接口	187
14.4.1	<i>FilenameFilter</i> 接口概要	187
14.4.2	<i>FilenameFilter</i> 接口的使用	188
14.4.3	<i>DirFilter</i>	188
14.4.4	<i>FileFilter</i> 类	188
14.4.5	<i>WildFilter</i> 类	190
14.5	<i>Ls</i> 类	191
14.5.1	<i>Ls.getWildNames</i>	191
14.5.2	<i>Ls.wildToConsole</i>	192
14.5.3	<i>Ls.getDirNames</i>	192
14.5.4	<i>Ls.deleteWildFiles</i>	193
14.5.5	<i>Ls.WordPrintMerge</i>	193
14.6	<i>DirList</i>	194
14.7	小结	197
14.8	练习	197
第 15 章	流	198
15.1	<i>FileInputStream</i> 类	199
15.1.1	<i>FileInputStream</i> 类概要	199
15.1.2	<i>FileInputStream</i> 类的使用	200
15.1.3	<i>Futil.getFileInputStream</i>	201
15.1.4	<i>Futil.available</i>	202
15.2	<i>FileOutputStream</i> 类	203
15.2.1	<i>FileOutputStream</i> 类概要	203
15.2.2	<i>FileOutputStream</i> 类的使用	203
15.2.3	<i>Futil.getFileOutputStream</i>	206
15.2.4	<i>Futil.close (OutputStream)</i>	206
15.3	<i>DataInputstream</i> 类	207
15.3.1	<i>DataInputStream</i> 类概要	207
15.3.2	<i>DataInputStream</i> 类的使用	208
15.4	<i>DataOutputStream</i> 类	210
15.4.1	<i>DataOutputStream</i> 类概要	210
15.4.2	<i>DataOutputStream</i> 类的使用	211
15.5	<i>StreamSniffer</i> 类	213
15.5.1	<i>StreamSniffer</i> 类概要	218
15.5.2	<i>StreamSniffer</i> 类的使用	219
15.6	序列化	220
15.7	读取和写入 Float 的 GZIPed 文件	223
15.8	小结	225