



疯狂程

Java讲义

(第2版)

李刚 编著

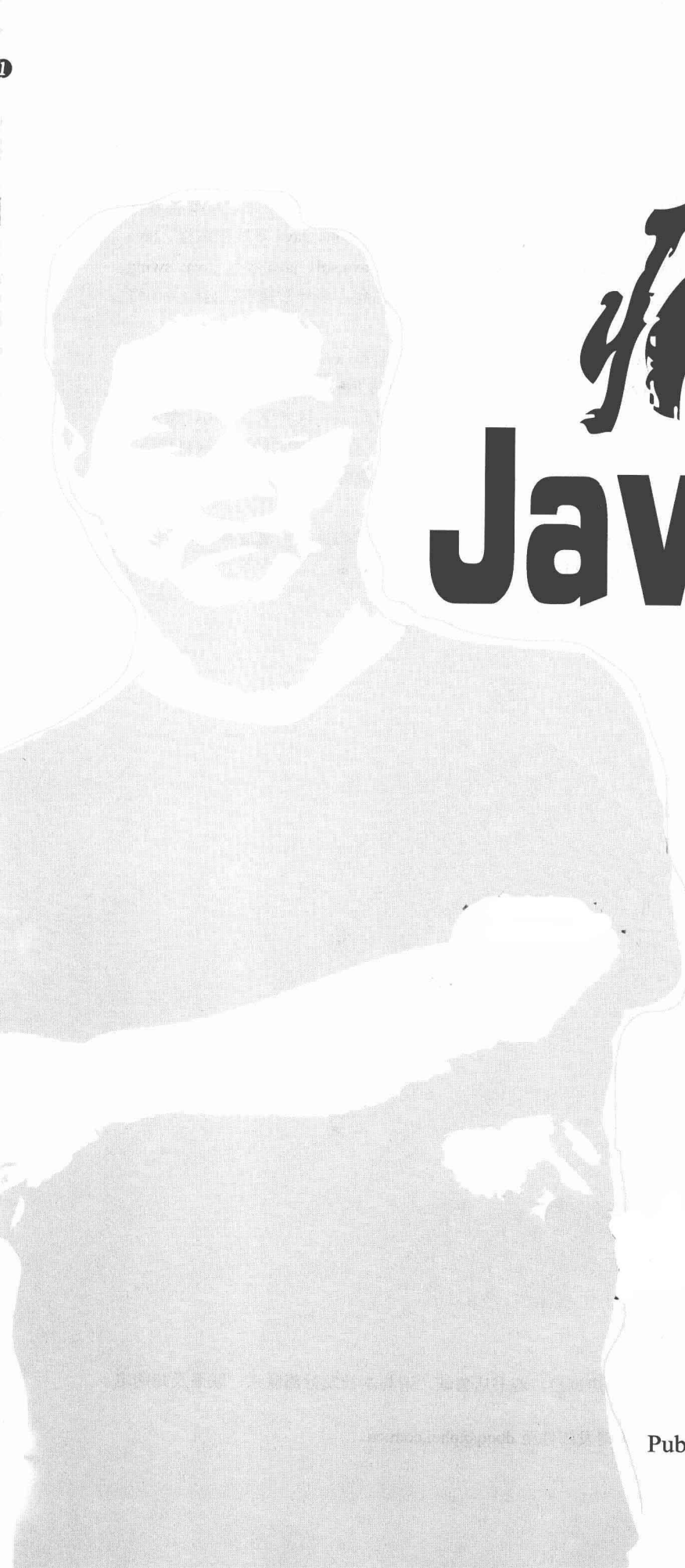
疯狂源自梦想

技术成就辉煌

疯狂源自梦想

技术成就辉煌





疯狂

Java讲义

(第2版)

李刚 编著

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书是《疯狂 Java 讲义》的第 2 版，第 2 版保持了第 1 版系统、全面、讲解浅显、细致的特性，全面介绍了新增的 Java 7 的新特性。

本书深入介绍了 Java 编程的相关方面，全书内容覆盖了 Java 的基本语法结构、Java 的面向对象特征、Java 集合框架体系、Java 泛型、异常处理、Java GUI 编程、JDBC 数据库编程、Java 注释、Java 的 IO 流体系、Java 多线程编程、Java 网络通信编程和 Java 反射机制；覆盖了 java.lang、java.util、java.text、java.io 和 java.nio、java.sql、java.awt、javax.swing 包下绝大部分类和接口。本书全面介绍了 Java 7 的二进制整数、菱形语法、增强 switch 语句、多异常捕获、自动关闭资源的 try 语句、JDBC 4.1 新特性、NIO.2、AIO 等新特性。

与第 1 版类似，本书并不单纯从知识角度来讲解 Java，而是从解决问题的角度来介绍 Java 语言，所以本书中涉及大量的实用案例开发：五子棋游戏、梭哈游戏、仿 QQ 的游戏大厅、MySQL 企业管理器、仿 EditPlus 的文本编辑器、多线程、断点下载工具、Spring 框架的 IoC 容器……这些案例既能让读者巩固每章的知识，又可以让读者学以致用，激发编程自豪感，进而引爆内心的编程激情。本书光盘里包含书中所有示例的代码和《疯狂 Java 实战演义》的所有项目代码，这些项目可以作为本书课后练习的“非标准答案”。如果读者需要获取关于课后习题的解决方法、编程思路，可以登录 <http://www.crazyit.org> 站点与笔者及本书庞大的读者群相互交流。

本书为所有打算深入掌握 Java 编程的读者而编写，适合各种层次的 Java 学习者和工作者阅读，也适合作为大学教育、培训机构的 Java 教材。但如果只是想简单涉猎 Java，则本书过于庞大，不适合阅读。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

图书在版编目 (CIP) 数据

疯狂 Java 讲义 / 李刚编著. —2 版. —北京: 电子工业出版社, 2012.1
ISBN 978-7-121-15578-9

I. ①疯… II. ①李… III. ①JAVA 语言—程序设计 IV. ①TP312

中国版本图书馆 CIP 数据核字 (2011) 第 270649 号

策划编辑: 张月萍

责任编辑: 葛 娜

技术审核: 白 涛

印 刷: 北京京科印刷有限公司

装 订: 三河市皇庄路通装订厂

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: 850×1168 1/16 印张: 54 字数: 1747 千字 彩插: 4

印 次: 2012 年 1 月第 1 次印刷

印 数: 5000 册 定价: 109.00 元 (含光盘 1 张)

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 zlts@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。



如何学习 Java

——谨以此文献给打算以编程为职业、并愿意为之疯狂的人

经常看到有些学生、求职者捧着一本类似 JBuilder 入门、Eclipse 指南之类的图书学习 Java，当他们学会了在这些工具中拖出窗体、安装按钮之后，就觉得自己掌握、甚至精通了 Java；又或是找来一本类似 JSP 动态网站编程之类的图书，学会使用 JSP 脚本编写一些页面后，就自我感觉掌握了 Java 开发。

还有一些学生、求职者听说 J2EE、Spring 或 EJB 很有前途，于是立即跑到书店或图书馆找来一本相关图书。希望立即学会它们，然后进入软件开发业、大显身手。

还有一些学生、求职者非常希望找到一本既速成、又大而全的图书，比如突击 J2EE 开发、一本书精通 J2EE 之类的图书（包括笔者曾出版的《轻量级 J2EE 企业应用实战》一书，据说销量不错），希望这样一本书就可以打通自己的“任督二脉”，一跃成为 J2EE 开发高手。

也有些学生、求职者非常喜欢 J2EE 项目实战、项目大全之类的图书，他们的想法很单纯：我按照书上介绍，按图索骥、依葫芦画瓢，应该很快就可学会 J2EE，很快就能成为一个受人羡慕的 J2EE 程序员了。

.....

凡此种种，不一而足。但最后的结果往往是失败，因为这种学习没有积累、没有根基，学习过程中困难重重，每天都被一些相同、类似的问题所困扰，起初热情十足，经常上论坛询问，按别人的说法解决问题之后很高兴，既不知道为什么错？也不知道为什么对？只是盲目地抄袭别人的说法。最后的结果有两种：

① 久而久之，热情丧失，最后放弃学习。

② 大部分常见问题都问遍了，最后也可以从事一些重复性开发，但一旦遇到新问题，又将束手无策。

第二种情形在普通程序员中占了极大的比例，笔者多次听到、看到（在网络上）有些程序员抱怨：我做了 2 年多 Java 程序员了，工资还是 3000 多点。偶尔笔者会与他们聊聊工作相关内容，他们会告诉笔者：我也用 Spring 了啊，我也用 EJB 了啊……他们感到非常不平衡，为什么我的工资这么低？其实笔者很想告诉他们：你们太浮躁了！你们确实是用了 Spring、Hibernate 又或是 EJB，但你们未想过为什么要用这些技术？用这些技术有什么好处？如果不用这些技术行不行？

很多时候，我们的程序员把 Java 当成一种脚本，而不是一门面向对象的语言。他们习惯了在 JSP 脚本中使用 Java，但从不去想 JSP 如何运行，Web 服务器里的网络通信、多线程机制，为何一个 JSP 页面能同时向多个请求者提供服务？更不会想如何开发 Web 服务器；他们像代码机器一样编写 Spring Bean 代码，但从不去理解 Spring 容器的作用，更不会想如何开发 Spring 容器。

有时候，笔者的学生在编写五子棋、梭哈等作业感到困难时，会向他们的大学师兄、朋友求救，这些程序员告诉他：不用写了，网上有下载的！听到这样回答，笔者不禁感到哑然：网上还有 Windows 下载呢！网上下载和自己编写是两码事。偶尔，笔者会怀念以前黑色屏幕、绿荧荧字符时代，那时候程序员很单纯：当我们想偷懒时，习惯思维是写一个小工具；现在程序员很聪明：当他们想偷懒时，习惯思维是从网上下一个小工具。但是，谁更幸福？

当笔者的学生把他们完成的小作业放上互联网之后，然后就有许多人称他们为“高手”！这个称呼却让他们万分惭愧；惭愧之余，他们也感到万分欣喜，非常有成就感，这就是编程的快乐。编程的过程，与寻宝的过程完全一样：历经辛苦，终于找到心中的梦想，这是何等的快乐？

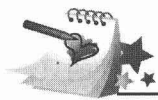
如果真的打算将编程当成职业，那就不应该如此浮躁，而是应该扎扎实实先学好 Java 语言，然后按 Java 本身的学习规律，踏踏实实一步一个脚印地学习，把基本功练扎实了才可获得更大的成功。

实际情况是，有多少程序员真正掌握了 Java 的面向对象？真正掌握了 Java 的多线程、网络通信、反射等内容？有多少 Java 程序员真正理解了类初始化时内存运行过程？又有多少程序员理解 Java 对象从创建到消失的全部细节？有几个程序员真正独立地编写过五子棋、梭哈、桌面弹球这种小游戏？又有几个 Java 程序员敢说：我可以开发 Struts？我可以开发 Spring？我可以开发 Tomcat？很多人又会说：这些都是许多人开发出来的！实际情况是：许多开源框架的核心最初完全是由一个人开发的。现在这些优秀程序已经出来了！你，是否深入研究过它们，是否深入掌握了它们？

如果要真正掌握 Java，包括后期的 Java EE 相关技术（例如 Struts、Spring、Hibernate 和 EJB 等），一定要记住笔者的话：绝不要从 IDE（如 JBuilder、Eclipse 和 NetBeans）工具开始学习！IDE 工具的功能很强大，初学者学起来也很容易上手，但也非常危险：因为 IDE 工具已经为我们做了许多事情，而软件开发者要全部了解软件开发的全部步骤。


A stylized handwritten signature in black ink, consisting of several fluid, connected strokes.

2011 年 12 月 17 日



光盘说明

一、光盘内容

 本光盘是《疯狂 Java 讲义（第 2 版）》一书的配书光盘，书中的代码按章、按节存放，即第 3 章第 1 节所使用的代码放在 codes 文件夹的 03\3.1 文件夹下，依此类推。

另：书中每份源代码也给出与光盘源文件的对应关系，方便读者查找。

 本光盘 codes 目录下有 18 个文件夹，其内容和含义说明如下：

(1) 01~18 个文件夹名对应于《疯狂 Java 讲义（第 2 版）》中的章名，即第 3 章所使用的代码放在 codes 文件夹的 03 文件夹下，依此类推。

(2) 本书所有代码都是 IDE 工具无关的程序，读者既可以在命令行窗口直接编译、运行这些代码，也可以导入 Eclipse、NetBeans 等 IDE 工具来运行它们。

(3) 本书第 12 章第 11 节的 TestTableModel.java 程序，以及第 13 章的绝大部分程序都需要连接数据库，所以读者需要先导入 *.sql 文件中的数据库脚本，并修改 mysql.ini 文件中的数据库连接信息。连接数据库时所用的驱动程序 JAR 文件为 mysql-connector-java-3.1.10-bin.jar 文件。这些需要连接数据库的程序里还提供了一个 *.cmd 文件，该文件是一个批处理文件，运行该文件可以运行相应的 Java 程序，例如 DatabaseMetaDataTest.java 对应的 *.cmd 文件为 runDatabaseMetaDataTest.cmd。

(4) 光盘根目录下提供了一个“Java 设计模式（疯狂 Java 联盟版）.chm”文件，这是一份关于设计模式的电子教材，由疯狂 Java 联盟的杨恩雄亲自编写、制作，他同意广大读者阅读、传播这份开源文档。

(5) 因为本书第 1 版有些读者提出关于《疯狂 Java 讲义》课后习题标准答案的问题，因此本书光盘根目录下包含一个 project_codes 文件夹，该文件夹里包含了疯狂 Java 联盟的杨恩雄编写的《疯狂 Java 实战演义》一书的光盘内容，该光盘中包含了大量实战性很强的项目，这些项目基本覆盖了《疯狂 Java 讲义（第 2 版）》课后习题的要求，读者可以参考相关案例来完成《疯狂 Java 讲义（第 2 版）》的课后习题。

二、运行环境

 本书中的程序在以下环境中调试通过：

(1) 安装 jdk-7-windows-i586.exe，安装完成后，添加 CLASSPATH 环境变量，该环境变量的值为 .;%JAVA_HOME%/lib/tools.jar;%JAVA_HOME%/lib/dt.jar。如果为了可以编译和运行 Java 程序，还应该在 PATH 环境变量中增加 %JAVA_HOME%/bin。其中 JAVA_HOME 代表 JDK（不是 JRE）的安装路径。如何安装上面工具，请参考本书的第 1 章。

(2) 安装 MySQL 5.5 或更高版本，安装 MySQL 时选择 GBK 编码方式（按第 13 章所介绍的方式安装）。

三、注意事项

(1) 书中有大量代码需要连接数据库，读者应修改数据库 URL 以及用户名、密码，让这些代码与

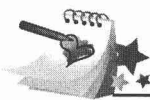
读者运行环境一致。如果项目下有 SQL 脚本，则导入 SQL 脚本即可；如果没有 SQL 脚本，系统将在运行时自动建表，读者只需创建对应的数据库即可。

(2) 在使用本光盘的程序时，请将程序拷贝到硬盘上，并去除文件的只读属性。

技术支持

如果您在使用本光盘中遇到不懂的技术问题，则可以登录如下网站与作者联系：

<http://www.crazyit.org>



前 言

2011年7月28日, Oracle 如约发布了 Java 7 正式版。从 Java 6 到 Java 7, 广大开发者经过了漫长的等待, 新发布的 Java 7 基本满足了广大开发者的渴望: Java 7 加入了不少新特性, 这些新特性进一步增强了 Java 语言的功能。

为了向广大工作者、学习者介绍最新、最前沿的 Java 知识, 在 Java 7 正式发布之前, 笔者已经深入研究过 Java 7 绝大部分可能新增的功能; 当 Java 7 正式发布之后, 笔者在第一时间开始了《疯狂 Java 讲义》的升级: 使用 Java 7 改写了全书所有程序, 全面介绍了 Java 7 的各种新特性。

在以“疯狂 Java 体系”图书为教材的疯狂软件教育中心 (www.fkjava.org) 里, 经常有学生询问: 为什么叫疯狂 Java 这个名字? 也有一些读者通过网络、邮件来询问这个问题。其实这个问题的答案可以在本书第 1 版的前言中找到。疯狂的本质是一种“享受编程”的状态。在一些不了解编程的人看来: 编程的人总面对着电脑、在键盘上敲打, 这种生活实在太枯燥了, 但实际上是因为他们并未真正了解编程, 并未真正走进编程。在外人眼中: 程序员不过是在敲打键盘; 但在程序员心中: 程序员敲出的每个字符, 都是程序的一部分。

程序是什么呢? 程序是对现实世界的数字化模拟。开发一个程序, 实际是创造一个或大或小的“模拟世界”。在这个过程中, 程序员享受着“创造”的乐趣, 程序员沉醉在他所创造的“模拟世界”里: 疯狂地设计、疯狂地编码实现。实现过程不断地遇到问题, 然后解决它; 不断地发现程序的缺陷, 然后重新设计、修复它——这个过程本身就是一种享受。一旦完全沉浸到编程世界里, 程序员是“物我两忘”的, 眼中看到的、心中想到的, 只有他正在创造的“模拟世界”。

在学会享受编程之前, 编程学习者都应该采用“案例驱动”的方式, 学习者需要明白程序的作用是: 解决问题——如果你的程序不能解决你自己的问题, 如何期望你的程序去解决别人的问题呢? 那你的程序的价值何在? ——知道一个知识点能解决什么问题, 才去学这个知识点, 而不是盲目学习! 因此本书强调编程实战, 强调以项目激发编程兴趣。

仅仅是看完这本书, 你不会成为高手! 在编程领域里, 没有所谓的“武林秘籍”, 再好的书一定要配合大量练习, 否则书里的知识依然属于作者, 而读者则仿佛身入宝山而一无所获的笨汉。本书配合了大量高强度的练习, 希望读者强迫自己去完成这些项目。这些习题的答案可以参考本书所附光盘中《疯狂 Java 实战演义》的配套代码。如果需要获得编程思路和交流, 可以登录 <http://www.crazyit.org> 与广大读者和笔者交流。

本书第 1 版面市的近 3 年时间里, 几万读者已经通过本书步入了 Java 编程世界, 这些读者的肯定、赞誉让笔者感到十分欣慰。也有不少读者给予本书一些很好的建议, 因此笔者在创作第 2 版时对部分章节进行了一些小调整, 希望这种调整能让本书日臻完善。

笔者非常欢迎所有热爱编程、愿意推动中国软件业的学习者、工作者对本书提出宝贵的意见, 非常乐意与大家交流。中国软件业还处于发展阶段, 所有热爱编程、愿意推动中国软件业的人应该联合起来, 共同为中国软件行业贡献自己的绵薄之力。

特点



本书并不是一本简单的 Java 入门教材, 也不是一门“闭门造车”式的 Java 读物。本书来自于笔者 6 年多的 Java 培训经历, 凝结了笔者近 6000 小时的授课经验, 总结了上千个 Java 学员学习过程中的典型错误。

因此, 本书具有如下三个特点:

1. 案例驱动，引爆编程激情

本书不再是知识点的铺陈，而是致力于将知识点融入实际项目的开发中，所以本书中涉及了大量 Java 案例：仿 QQ 的游戏大厅、MySQL 企业管理器、仿 EditPlus 的文本编辑器、多线程、断点下载工具……希望读者通过编写这些程序找到编程的乐趣。

2. 再现李刚老师课堂氛围

本书的内容是笔者 6 年多授课经历的总结，知识体系取自疯狂 Java 实战的课程体系。本书力求再现笔者的课堂氛围：以浅显比喻代替乏味的讲解，以疯狂实战代替空洞的理论。书中包含了大量“注意”、“学生提问”部分，这些正是上千个 Java 学员所犯错误的汇总。

3. 注释详细，轻松上手

为了降低读者阅读的难度，书中代码的注释非常详细，几乎每两行代码就有一行注释。不仅如此，本书甚至还把一些简单理论作为注释穿插到代码中，力求让读者能轻松上手。

本书所有程序中关键代码以粗体字标出，也是为了帮助读者能迅速找到这些程序的关键点。

本书写给谁看










如果你仅仅想对 Java 有所涉猎，那么本书并不适合你；如果你想全面掌握 Java 语言，并使用 Java 来解决问题、开发项目，或者希望以 Java 编程作为你的职业，那么本书将非常适合你。希望本书能引爆你内心潜在的编程激情，如果本书能让你产生废寝忘食的感觉，那笔者就非常欣慰了。

2011-12-17


目 录

CONTENTS

第 1 章 Java 语言概述	1	第 2 章 理解面向对象	21
1.1 Java 语言的发展简史	2	2.1 面向对象	22
1.2 Java 的竞争对手及各自优势	4	2.1.1 结构化程序设计简介	22
1.2.1 C#简介和优势	4	2.1.2 程序的三种基本结构	23
1.2.2 Ruby 简介和优势	5	2.1.3 面向对象程序设计简介	25
1.2.3 Python 简介和优势	5	2.1.4 面向对象的基本特征	26
1.3 Java 程序运行机制	6	2.2 UML (统一建模语言) 介绍	27
1.3.1 高级语言的运行机制	6	2.2.1 用例图	29
1.3.2 Java 程序的运行机制和 JVM	6	2.2.2 类图	29
1.4 开发 Java 的准备	7	2.2.3 组件图	31
1.4.1 下载和安装 Java 7 的 JDK	8	2.2.4 部署图	32
学生提问 不是说 JVM 是运行 Java 程序的虚拟机吗? 那 JRE 和 JVM 的关系是怎样的呢?	8	2.2.5 顺序图	32
学生提问 为什么不安装公共 JRE 呢?	9	2.2.6 活动图	33
1.4.2 设置 PATH 环境变量	10	2.2.7 状态机图	34
学生提问 为什么选择用户变量? 用户变量与系统变量有什么区别?	11	2.3 Java 的面向对象特征	35
1.5 第一个 Java 程序	11	2.3.1 一切都是对象	35
1.5.1 编辑 Java 源代码	11	2.3.2 类和对象	35
1.5.2 编译 Java 程序	12	2.4 本章小结	36
学生提问 当我们编译 C 程序时, 不仅需要指定存放目标文件的位置, 也需要指定目标文件的文件名, 这里使用 javac 编译 Java 程序时怎么不需要指定目标文件的文件名呢?	12	第 3 章 数据类型和运算符	37
1.5.3 运行 Java 程序	13	3.1 注释	38
1.5.4 根据 CLASSPATH 环境变量定位类	14	3.1.1 单行注释和多行注释	38
3.1.2 文档注释	39	学生提问 API 文档是什么?	39
3.1.3 为什么要掌握查看 API 文档的方法?	41	3.2 标识符和关键字	45
3.2.1 分隔符	45	3.2.1 分隔符	45
3.2.2 标识符规则	46	3.2.2 标识符规则	46
3.2.3 Java 关键字	46	3.2.3 Java 关键字	46
3.3 数据类型分类	47	3.3 数据类型分类	47
3.4 基本数据类型	47	3.4 基本数据类型	47
3.4.1 整型	48	3.4.1 整型	48
3.4.2 Java 7 新增的二进制整数	49	3.4.2 Java 7 新增的二进制整数	49
3.4.3 字符型	50	3.4.3 字符型	50
3.4.4 浮点型	51	3.4.4 浮点型	51
3.4.5 Java 7 新增的数值中使用下划线分隔	52	3.4.5 Java 7 新增的数值中使用下划线分隔	52
3.4.6 布尔型	52	3.4.6 布尔型	52
3.5 基本类型的类型转换	53	3.5 基本类型的类型转换	53
3.5.1 自动类型转换	53	3.5.1 自动类型转换	53
1.6 Java 程序的基本规则	15		
1.6.1 Java 程序的组织形式	15		
1.6.2 Java 源文件的命名规则	15		
1.6.3 初学者容易犯的错误	16		
1.7 垃圾回收机制	18		
1.8 何时开始使用 IDE 工具	19		
学生提问 我想学习 Java 编程, 到底是学习 Eclipse 好呢, 还是学习 NetBeans 好呢?	20		
1.9 本章小结	20		
本章练习	20		


3.5.2	强制类型转换	54	4.6.2	基本类型数组的初始化	93
3.5.3	表达式类型的自动提升	56	4.6.3	引用类型数组的初始化	94
3.6	直接量	57	4.6.4	没有多维数组	96
3.6.1	直接量的类型	57	 我是否可以	让图 4.13 中灰色覆盖	
3.6.2	直接量的赋值	57	的数组元素再次指向另一个数		
3.7	运算符	58	组? 这样不就可以扩展成三维数		
3.7.1	算术运算符	58	组吗? 甚至扩展成更多维的数		
3.7.2	赋值运算符	61	组?	98	
3.7.3	位运算符	61	4.6.5	操作数组的工具类	99
3.7.4	扩展后的赋值运算符	64	4.6.6	数组的应用举例	101
3.7.5	比较运算符	65	4.7	本章小结	104
3.7.6	逻辑运算符	66	本章练习	104	
3.7.7	三目运算符	66	第 5 章 面向对象 (上)	105	
3.7.8	运算符的结合性和优先级	67	5.1	类和对象	106
3.8	本章小结	68	5.1.1	定义类	106
	本章练习	68	 构造器不是没有返回值吗? 为什		
第 4 章 流程控制与数组		70	么不能用 void 修饰呢?	108	
4.1	顺序结构	71	5.1.2	对象的产生和使用	108
4.2	分支结构	71	5.1.3	对象、引用和指针	109
4.2.1	if 条件语句	71	5.1.4	对象的 this 引用	110
4.2.2	Java 7 的 switch 分支语句	75	5.2	方法详解	114
4.3	循环结构	77	5.2.1	方法的所属性	114
4.3.1	while 循环语句	77	5.2.2	方法的参数传递机制	115
4.3.2	do while 循环语句	78	5.2.3	形参个数可变的方法	118
4.3.3	for 循环	79	5.2.4	递归方法	119
4.3.4	嵌套循环	81	5.2.5	方法重载	121
4.4	控制循环结构	83	 为什么方法的返回值类型不能用		
4.4.1	使用 break 结束循环	83	于区分重载的方法?	121	
4.4.2	使用 continue 结束本次循环	84	5.3	成员变量和局部变量	122
4.4.3	使用 return 结束方法	85	5.3.1	成员变量和局部变量	122
4.5	数组类型	86	5.3.2	成员变量的初始化和内存中的	
4.5.1	理解数组: 数组也是一种类型	86	运行机制	125	
 int[] 是一种类型吗? 怎么使用这种		86	5.3.3	局部变量的初始化和内存中的	
类型呢?		86	运行机制	127	
4.5.2	定义数组	86	5.3.4	变量的使用规则	128
4.5.3	数组的初始化	87	5.4	隐藏和封装	129
 能不能只分配内存空间, 不赋初始		87	5.4.1	理解封装	129
值呢?		87	5.4.2	使用访问控制符	129
4.5.4	使用数组	88	5.4.3	package、import 和 import static	132
 为什么要我记住这些异常信息?		89	5.4.4	Java 的常用包	137
4.5.5	foreach 循环	89	5.5	深入构造器	137
4.6	深入数组	91	5.5.1	使用构造器执行初始化	138
4.6.1	内存中的数组	91	 构造器是创建 Java 对象的途径, 是		
 为什么有栈内存和堆内存之分?		91	不是说构造器完全负责创建 Java		
			对象?	138	

5.5.2	构造器重载	139	6.4.1	final 成员变量	174
学生提问	为什么要用 this 来调用另一个重载的构造器? 我把另一个构造器里的代码复制、粘贴到这个构造器里不就可以了么?	140	6.4.2	final 局部变量	176
5.6	类的继承	140	6.4.3	final 修饰基本类型变量和引用类型变量的区别	177
5.6.1	继承的特点	141	6.4.4	可执行“宏替换”的 final 变量	177
5.6.2	重写父类的方法	142	6.4.5	final 方法	179
5.6.3	super 限定	143	6.4.6	final 类	180
5.6.4	调用父类构造器	146	6.4.7	不可变类	180
学生提问	为什么我创建 Java 对象时从未感觉到 java.lang. Object 类的构造器被调用过?	148	6.4.8	缓存实例的不可变类	183
5.7	多态	148	6.5	抽象类	186
5.7.1	多态性	148	6.5.1	抽象方法和抽象类	186
5.7.2	引用变量的强制类型转换	149	6.5.2	抽象类的作用	189
5.7.3	instanceof 运算符	151	6.6	更彻底的抽象: 接口	190
5.8	继承与组合	151	6.6.1	接口的概念	190
5.8.1	使用继承的注意点	151	6.6.2	接口的定义	191
5.8.2	利用组合实现复用	153	6.6.3	接口的继承	192
学生提问	使用组合关系来实现复用时, 需要创建两个 Animal 对象, 是不是意味着使用组合关系时系统开销更大?	155	6.6.4	使用接口	193
5.9	初始化块	156	6.6.5	接口和抽象类	194
5.9.1	使用初始化块	156	6.6.6	面向接口编程	195
5.9.2	初始化块和构造器	157	6.7	内部类	199
5.9.3	静态初始化块	158	6.7.1	非静态内部类	199
5.10	本章小结	160	学生提问	非静态内部类对象和外部类对象的关系是怎样的?	202
	本章练习	161	6.7.2	静态内部类	203
第 6 章	面向对象 (下)	162	学生提问	为什么静态内部类的实例方法也不能访问外部类的实例属性呢?	204
6.1	Java 7 增强的包装类	163	学生提问	接口里是否能定义内部接口?	205
学生提问	Java 为什么要对这些数据进行缓存呢?	166	6.7.3	使用内部类	205
6.2	处理对象	167	学生提问	既然内部类是外部类的成员, 那么是否可以外部类定义子类, 在子类中再定义一个内部类来重写其父类中的内部类?	207
6.2.1	打印对象和 toString 方法	167	6.7.4	局部内部类	207
6.2.2	= 和 equals 方法	168	6.7.5	匿名内部类	208
学生提问	判断 obj 是否为 Person 类的实例时, 为何不用 obj instanceof Person 来判断呢?	172	6.7.6	闭包 (Closure) 和回调	211
6.3	类成员	172	6.8	枚举类	213
6.3.1	理解类成员	172	6.8.1	手动实现枚举类	213
6.3.2	单例 (Singleton) 类	173	6.8.2	枚举类入门	215
6.4	final 修饰符	174	6.8.3	枚举类的 Field、方法和构造器	216
			6.8.4	实现接口的枚举类	218
			学生提问	枚举类不是用 final 修饰了吗? 怎么还能派生子类呢?	207
			6.8.5	包含抽象方法的枚举类	220
			6.9	对象与垃圾回收	221
			6.9.1	对象在内存中的状态	221

6.9.2	强制垃圾回收	222	7.6.8	使用 SimpleDateFormat 格式化日期	272
6.9.3	finalize 方法	223	7.7	本章小结	273
6.9.4	对象的软、弱和虚引用	225		本章练习	273
6.10	修饰符的适用范围	228	第 8 章	Java 集合	274
6.11	使用 JAR 文件	229	8.1	Java 集合概述	275
6.11.1	jar 命令详解	230	8.2	Collection 和 Iterator 接口	276
6.11.2	创建可执行的 JAR 包	232	8.2.1	使用 Iterator 接口遍历集合元素	278
6.11.3	关于 JAR 包的技巧	232	8.2.2	使用 foreach 循环遍历集合元素	280
6.12	本章小结	233	8.3	Set 集合	280
	本章练习	233	8.3.1	HashSet 类	281
第 7 章	与运行环境交互	234	 hashCode()方法对于 HashSet 是不是十分重要?	283	
7.1	与用户互动	235	8.3.2	LinkedHashSet 类	285
7.1.1	运行 Java 程序的参数	235	8.3.3	TreeSet 类	286
7.1.2	使用 Scanner 获取键盘输入	236	8.3.4	EnumSet 类	292
7.1.3	使用 BufferedReader 获取键盘输入	238	8.3.5	各 Set 实现类的性能分析	293
7.2	系统相关	238	8.4	List 集合	294
7.2.1	System 类	239	8.4.1	List 接口和 ListIterator 接口	294
7.2.2	Runtime 类	240	8.4.2	ArrayList 和 Vector 实现类	297
7.3	常用类	241	8.4.3	固定长度的 List	298
7.3.1	Object 类	241	8.5	Queue 集合	298
7.3.2	Java 7 新增的 Objects 类	243	8.5.1	PriorityQueue 实现类	299
7.3.3	String、StringBuffer 和 StringBuilder 类	244	8.5.2	Deque 接口与 ArrayDeque 实现类	300
7.3.4	Math 类	247	8.5.3	LinkedList 实现类	301
7.3.5	Java 7 的 ThreadLocalRandom 与 Random	248	8.5.4	各种线性表的性能分析	302
7.3.6	BigDecimal 类	250	8.6	Map	303
7.4	处理日期的类	253	8.6.1	HashMap 和 Hashtable 实现类	304
7.4.1	Date 类	253	8.6.2	LinkedHashMap 实现类	308
7.4.2	Calendar 类	254	8.6.3	使用 Properties 读写属性文件	308
7.4.3	TimeZone 类	257	8.6.4	SortedMap 接口和 TreeMap 实现类	309
7.5	正则表达式	258	8.6.5	WeakHashMap 实现类	311
7.5.1	创建正则表达式	258	8.6.6	IdentityHashMap 实现类	312
7.5.2	使用正则表达式	261	8.6.7	EnumMap 实现类	313
7.6	国际化与格式化	264	8.6.8	各 Map 实现类的性能分析	314
7.6.1	Java 国际化的思路	265	8.7	HashSet 和 HashMap 的性能选项	314
7.6.2	Java 支持的语言和国家	265	8.8	操作集合的工具类: Collections	315
7.6.3	完成程序国际化	266	8.8.1	排序操作	315
7.6.4	使用 MessageFormat 处理包含占位符的字符串	267	8.8.2	查找、替换操作	318
7.6.5	使用类文件代替资源文件	268	8.8.3	同步控制	319
7.6.6	使用 NumberFormat 格式化数字	269	8.8.4	设置不可变集合	320
7.6.7	使用 DateFormat 格式化日期、时间	270	8.9	烦琐的接口: Enumeration	320
			8.10	本章小结	321

本章练习	321	10.4.5 异常链	363
第 9 章 泛型	322	10.5 Java 的异常跟踪栈	365
9.1 泛型入门	323	10.6 异常处理规则	366
9.1.1 编译时不检查类型的异常	323	10.6.1 不要过度使用异常	367
9.1.2 手动实现编译时检查类型	324	10.6.2 不要使用过于庞大的 try 块	368
9.1.3 使用泛型	324	10.6.3 避免使用 Catch All 语句	368
9.1.4 Java 7 泛型的“菱形”语法	325	10.6.4 不要忽略捕获到的异常	368
9.2 深入泛型	326	10.7 本章小结	369
9.2.1 定义泛型接口、类	326	本章练习	369
9.2.2 从泛型类派生子类	328	第 11 章 AWT 编程	370
9.2.3 并不存在泛型类	329	11.1 GUI (图形用户界面) 和 AWT	371
9.3 类型通配符	329	11.2 AWT 容器	372
9.3.1 使用类型通配符	331	11.3 布局管理器	375
9.3.2 设定类型通配符的上限	332	11.3.1 FlowLayout 布局管理器	375
9.3.3 设定类型形参的上限	333	11.3.2 BorderLayout 布局管理器	376
9.4 泛型方法	334	学生提问 BorderLayout 最多只能放置 5 个组	
9.4.1 定义泛型方法	334	件吗? 那它也太大不实用了吧?	377
9.4.2 泛型方法和类型通配符的区别	337	11.3.3 GridLayout 布局管理器	378
9.4.3 Java 7 的“菱形”语法与泛型		11.3.4 GridBagLayout 布局管理器	379
构造器	338	11.3.5 CardLayout 布局管理器	381
9.4.4 设定通配符下限	339	11.3.6 绝对定位	383
9.4.5 泛型方法与方法重载	341	11.3.7 BoxLayout 布局管理器	384
9.5 擦除和转换	341	学生提问 图 11.15 和图 11.16 显示的所有按	
9.6 泛型与数组	343	钮都紧挨在一起, 如果希望像	
9.7 本章小结	344	FlowLayout、GridLayout 等布局管	
第 10 章 异常处理	345	理器那样指定组件的间距应该怎	
10.1 异常概述	346	么办?	385
10.2 异常处理机制	347	11.4 AWT 常用组件	386
10.2.1 使用 try...catch 捕获异常	347	11.4.1 基本组件	386
10.2.2 异常类的继承体系	349	11.4.2 对话框 (Dialog)	388
10.2.3 Java 7 提供的多异常捕获	351	11.5 事件处理	390
10.2.4 访问异常信息	352	11.5.1 Java 事件模型的流程	390
10.2.5 使用 finally 回收资源	353	11.5.2 事件和事件监听器	392
10.2.6 异常处理的嵌套	355	11.5.3 事件适配器	396
10.2.7 Java 7 的自动关闭资源的 try		11.5.4 使用内部类实现监听器	397
语句	355	11.5.5 使用外部类实现监听器	397
10.3 Checked 异常和 Runtime		11.5.6 类本身作为事件监听器类	398
异常体系	357	11.5.7 匿名内部类实现监听器	399
10.3.1 使用 throws 声明抛出异常	357	11.6 AWT 菜单	399
10.4 使用 throw 抛出异常	359	11.6.1 菜单条、菜单和菜单项	399
10.4.1 抛出异常	359	11.6.2 右键菜单	401
10.4.2 自定义异常类	360	学生提问 为什么即使我没有给多行文本域	
10.4.3 catch 和 throw 同时使用	361	编写右键菜单, 但当我在多行文本	
10.4.4 Java 7 增强的 throw 语句	362	域上单击右键时也一定会弹出右	
		键菜单?	403

11.7	在 AWT 中绘图	403	12.6.1	创建进度条	482
11.7.1	画图的实现原理	403	12.6.2	创建进度对话框	485
11.7.2	使用 Graphics 类	404	12.7	使用 JSlider 和 BoundedRangeModel 创建滑动条	486
11.8	处理位图	408	12.8	使用 JSpinner 和 SpinnerModel 创建微调控制器	489
11.8.1	Image 抽象类和 BufferedImage 实现类	408	12.9	使用 JList、JcomboBox 创建列表框	492
11.8.2	使用 ImageIO 输入/输出位图	411	12.9.1	简单列表框	493
11.9	剪贴板	414	12.9.2	不强制存储列表项的 ListModel 和 ComboBoxModel	496
11.9.1	数据传递的类和接口	415	12.9.3	强制存储列表项的 DefaultListModel 和 DefaultComboBoxModel	499
11.9.2	传递文本	415	为什么 JComboBox 提供了添加、删 除列表项的方法? 而 JList 没有提供 添加、删除列表项的方法呢?	510	
11.9.3	使用系统剪贴板传递图像	417	12.9.4	使用 ListCellRenderer 改变列表 项外观	501
11.9.4	使用本地剪贴板传递对象引用	421	12.10	使用 JTree 和 TreeModel 创建树	503
11.9.5	通过系统剪贴板传递 Java 对象	423	12.10.1	创建树	504
11.10	拖放功能	426	12.10.2	拖动、编辑树节点	507
11.10.1	拖放目标	427	12.10.3	监听节点事件	511
11.10.2	拖放源	429	12.10.4	使用 DefaultTreeCellRenderer 改变节点外观	512
11.11	本章小结	431	12.10.5	扩展 DefaultTreeCellRenderer 改变节点外观	514
本章练习		431	12.10.6	实现 TreeCellRenderer 改变 节点外观	516
第 12 章	Swing 编程	432	12.11	使用 JTable 和 TableModel 创建表格	518
12.1	Swing 概述	433	12.11.1	创建表格	518
12.2	Swing 基本组件的用法	434	我们指定的表格数据、表格列标题 都是 Object 类型的数组, JTable 如 何显示这些 Object 对象?	519	
12.2.1	Java 7 的 Swing 组件层次	434	12.11.2	TableModel 和监听器	524
12.2.2	AWT 组件的 Swing 实现	435	12.11.3	TableColumnModel 和监听器	528
为什么单击 Swing 多行文本域时不 是弹出像 AWT 多行文本域中的右 键菜单?		441	12.11.4	实现排序	531
12.2.3	为组件设置边框	441	12.11.5	绘制单元格内容	534
12.2.4	Swing 组件的双缓冲和键盘驱动	443	12.11.6	编辑单元格内容	537
12.2.5	使用 JToolBar 创建工具条	444	12.12	使用 JFormattedTextField 和 JTextPane 创建格式文本	541
12.2.6	使用 JFileChooser 和 Java 7 增强的 JColorChooser	446	12.12.1	监听 Document 的变化	541
12.2.7	使用 JOptionPane	454	12.12.2	使用 JPasswordField	543
12.3	Swing 中的特殊容器	459	12.12.3	使用 JFormattedTextField	544
12.3.1	使用 JSplitPane	459	12.12.4	使用 JEditorPane	552
12.3.2	使用 JTabbedPane	461			
12.3.3	使用 JLayeredPane、JDesktopPane 和 JInternalFrame	464			
12.4	Swing 简化的拖放功能	472			
12.5	Java 7 新增的 Swing 功能	473			
12.5.1	使用 JLayer 装饰组件	473			
12.5.2	创建透明、不规则形状窗口	479			
12.6	使用 JProgressBar、ProgressMonitor 和 BoundedRangeModel 创建进度条	481			

12.12.5 使用 JTextPane	552	13.6 Java 7 的 RowSet 1.1	620
12.13 本章小结	559	13.6.1 Java 7 新增的 RowSetFactory 与 RowSet	620
本章练习	559	13.6.2 离线 RowSet	623
第 13 章 MySQL 数据库与 JDBC 编程	560	13.6.3 离线 RowSet 的查询分页	624
13.1 JDBC 基础	561	13.7 事务处理	625
13.1.1 JDBC 简介	561	13.7.1 事务的概念和 MySQL 事务支持	626
13.1.2 JDBC 驱动程序	562	13.7.2 JDBC 的事务支持	627
13.2 SQL 语法	563	13.7.3 批量更新	629
13.2.1 安装数据库	563	13.8 分析数据库信息	630
13.2.2 关系数据库基本概念和 MySQL 基本命令	565	13.8.1 使用 DatabaseMetaData 分析数据库信息	630
13.2.3 SQL 语句基础	567	13.8.2 使用系统表分析数据库信息	632
13.2.4 DDL 语句	568	13.8.3 选择合适的分析方式	633
13.2.5 数据库约束	572	13.9 使用连接池管理连接	633
13.2.6 索引	579	13.9.1 DBCP 数据源	634
13.2.7 视图	579	13.9.2 C3P0 数据源	634
13.2.8 DML 语句语法	580	13.10 本章小结	635
13.2.9 单表查询	583	本章练习	635
13.2.10 数据库函数	587	第 14 章 Annotation (注释)	636
13.2.11 分组和组函数	589	14.1 基本 Annotation	637
13.2.12 多表连接查询	591	14.1.1 限定重写父类方法: @Override	637
13.2.13 子查询	595	14.1.2 标示已过时: @Deprecated	638
13.2.14 集合运算	596	14.1.3 抑制编译器警告: @SuppressWarnings	639
13.3 JDBC 的典型用法	597	14.1.4 Java 7 的“堆污染”警告与 @SafeVarargs	639
13.3.1 JDBC 常用接口和类简介	598	14.2 JDK 的元 Annotation	640
13.3.2 JDBC 编程步骤	599	14.2.1 使用 @Retention	640
 前面给出的仅仅是 MySQL 和 Oracle 两种数据库的驱动,我看不出驱动类 字符串有什么规律啊。如果我希望使 用其他数据库,那怎么找到其他数据 库的驱动类呢?	600	14.2.2 使用 @Target	641
13.4 执行 SQL 语句的方式	602	14.2.3 使用 @Documented	641
13.4.1 使用 executeUpdate 方法执行 DDL 和 DML 语句	602	14.2.4 使用 @Inherited	642
13.4.2 使用 execute 方法执行 SQL 语句	604	14.3 自定义 Annotation	643
13.4.3 使用 PreparedStatement 执行 SQL 语句	605	14.3.1 定义 Annotation	643
13.4.4 使用 CallableStatement 调用 存储过程	609	14.3.2 提取 Annotation 信息	645
13.5 管理结果集	611	14.3.3 使用 Annotation 的示例	646
13.5.1 可滚动、可更新的结果集	611	14.4 编译时处理 Annotation	650
13.5.2 处理 Blob 类型数据	613	14.5 本章小结	654
13.5.3 使用 ResultSetMetaData 分析结果集	617	第 15 章 输入/输出	655
		15.1 File 类	656
		15.1.1 访问文件和目录	656
		15.1.2 文件过滤器	658
		15.2 理解 Java 的 IO 流	659

15.2.1	流的分类	659	16.2.1	继承 Thread 类创建线程类	710
15.2.2	流的概念模型	660	16.2.2	实现 Runnable 接口创建线程类	711
15.3	字节流和字符流	661	16.2.3	使用 Callable 和 Future 创建线程	713
15.3.1	InputStream 和 Reader	661	16.2.4	创建线程的三种方式对比	714
15.3.2	OutputStream 和 Writer	663	16.3	线程的生命周期	715
15.4	输入/输出流体系	665	16.3.1	新建和就绪状态	715
15.4.1	处理流的用法	665	16.3.2	运行和阻塞状态	716
15.4.2	输入/输出流体系	666	16.3.3	线程死亡	717
15.4.3	转换流	668	16.4	控制线程	718
	怎么没有把字符流转换成字节流 的转换流呢?	668	16.4.1	join 线程	719
15.4.4	推回输入流	669	16.4.2	后台线程	720
15.5	重定向标准输入/输出	671	16.4.3	线程睡眠: sleep	721
15.6	Java 虚拟机读写其他 进程的数据	672	16.4.4	线程让步: yield	721
15.7	RandomAccessFile	674	16.4.5	改变线程优先级	722
15.8	对象序列化	677	16.5	线程同步	724
15.8.1	序列化的含义和意义	677	16.5.1	线程安全问题	724
15.8.2	使用对象流实现序列化	678	16.5.2	同步代码块	726
15.8.3	对象引用的序列化	680	16.5.3	同步方法	727
15.8.4	自定义序列化	683	16.5.4	释放同步监视器的锁定	729
15.8.5	另一种自定义序列化机制	688	16.5.5	同步锁 (Lock)	730
15.8.6	版本	690	16.5.6	死锁	732
15.9	NIO	691	16.6	线程通信	733
15.9.1	Java 新 IO 概述	691	16.6.1	传统的线程通信	734
15.9.2	使用 Buffer	691	16.6.2	使用 Condition 控制线程通信	737
15.9.3	使用 Channel	694	16.6.3	使用阻塞队列 (BlockingQueue) 控制线程通信	739
15.9.4	字符集和 Charset	697	16.7	线程组和未处理的异常	742
	二进制序列与字符之间如何对应 呢?	698	16.8	线程池	744
15.9.5	文件锁	699	16.8.1	Java 5 实现的线程池	745
15.10	Java 7 的 NIO.2	701	16.8.2	Java 7 新增的 ForkJoinPool	746
15.10.1	Path、Paths 和 Files 核心 API	701	16.9	线程相关类	750
15.10.2	使用 FileVisitor 遍历 文件和目录	702	16.9.1	ThreadLocal 类	750
15.10.3	使用 WatchService 监控文件 变化	704	16.9.2	包装线程不安全的集合	752
15.10.4	访问文件属性	705	16.9.3	线程安全的集合类	752
15.11	本章小结	706	16.10	本章小结	753
	本章练习	706	第 17 章	网络编程	754
第 16 章	多线程	707	17.1	网络编程的基础知识	755
16.1	线程概述	708	17.1.1	网络基础知识	755
16.1.1	线程和进程	708	17.1.2	IP 地址和端口号	756
16.1.2	多线程的优势	709	17.2	Java 的基本网络支持	757
16.2	线程的创建和启动	710	17.2.1	使用 InetAddress	757
			17.2.2	使用 URLDecoder 和 URLEncoder	758
			17.2.3	使用 URL 和 URLConnection	759