

高等学校教材·计算机科学与技术

可赠送课件

dingl@tup.tsinghua.edu.cn

Java面向对象程序设计教程

李发致 编著



清华大学出版社

高等学校教材·计算机科学与技术

Java 面向对象程序设计教程

李发致 编著

清华大学出版社

北 京

内 容 简 介

本书围绕着面向对象技术和面向对象编程语言这一主题展开,注重对学习 Java 语言过程中的主干问题进行深入分析,使学生能搞清问题的来龙去脉。

书中对虚拟机机制、应用程序主方法的使用、对象初始化、对象使用、类设计以及面向对象编程规范等问题进行了比较深入的梳理与分析;对知识点溯本求源,力求给出合理的解释,做到言之有物,绝不含糊其辞。书中选用的例子是典型的但绝不是繁琐的,例子的设计也尽可能精致,做到有的放矢,避免粗制滥造。书中所述都是作者思考探索过的,绝不存在凑数式的资料堆砌。

本书适用于高等学校计算机及相关专业的师生,也适用于 Java 初、中级编程人员以及对从事面向对象编程感兴趣的读者。

版权所有,翻印必究。举报电话:010-62782989 13901104297 13801310933

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

图书在版编目(CIP)数据

Java 面向对象程序设计教程/李发致编著. —北京:清华大学出版社,2004.9
(高等学校教材·计算机科学与技术)

ISBN 7-302-09130-7

I. J… II. 李… III. JAVA 语言—程序设计—高等学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字(2004)第 075365 号

出 版 者:清华大学出版社

<http://www.tup.com.cn>

社总机:010-62770175

地 址:北京清华大学学研大厦

邮 编:100084

客户服务:010-62776969

组稿编辑:闫红梅

文稿编辑:陶萃渊

印 刷 者:清华大学印刷厂

装 订 者:三河市金元装订厂

发 行 者:新华书店总店北京发行所

开 本:185×260 印张:29.75 字数:734 千字

版 次:2004 年 9 月第 1 版 2004 年 9 月第 1 次印刷

书 号:ISBN 7-302-09130-7/TP·6438

印 数:0001~5000

定 价:40.00 元

本书如存在文字不清、漏印以及缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。联系电话:(010)62770175-3103 或(010)62795704

清华大学出版社计算机教材

编审委员会成员

(按地区排序)

清华大学	周立柱 教授	北京大学	杨冬青 教授
	覃 征 教授		陈 钟 教授
	王建民 教授		陈立军 副教授
	刘 强 副教授	中国人民大学	王 珊 教授
	冯建华 副教授		孟小峰 教授
北京航空航天大学	马殿富 教授		陈 红 教授
	吴超英 副教授	北京信息工程学院	孟庆昌 教授
	姚淑珍 教授	石油大学(北京)	陈 明 教授
北京交通大学	阮秋琦 教授	天津大学	艾德才 教授
北京科技大学	杨炳儒 教授		
南开大学	吴功宜 教授		
复旦大学	吴立德 教授	上海交通大学	傅育熙 教授
	吴百锋 教授		蒋建伟 副教授
	杨卫东 副教授	华东师范大学	杨宗源 教授
华东理工大学	邵志清 教授		应吉康 教授
东华大学	乐嘉锦 教授	上海第二工业大学	蒋川群 教授
浙江大学	吴朝晖 教授	南京大学	骆 斌 教授
	李善平 教授	南京理工大学	张功萱 教授
南京航空航天大学	秦小麟 教授	苏州大学	龚声蓉 教授
南京邮电学院	朱秀昌 教授		
江苏大学	宋余庆 教授		
武汉大学	何炎祥 教授	华中科技大学	刘乐善 教授
中南财经政法大学	刘腾红 教授		朱定华 教授
武汉理工大学	李中年 教授	华中师范大学	魏开平 教授
			王林平 副教授

国防科技大学	赵克佳 教授	中南大学	陈松乔 教授
	肖 依 副教授		
湖南大学	林亚平 教授		
	邹北骥 教授		
西安交通大学	沈钧毅 教授	西北大学	周明全 教授
	齐 勇 教授	西安石油学院	方 明 教授
长安大学	巨永峰 教授		
西安邮电学院	陈莉君 副教授		
哈尔滨工业大学	郭茂祖 教授	吉林大学	何 桥 教授
长春工程学院	沙胜贤 教授		徐一平 教授
			毕 强 教授
山东大学	孟祥旭 教授	山东科技大学	郑永果 教授
	郝兴伟 教授		
中山大学	潘小轰 教授	厦门大学	冯少荣 副教授
福州大学	林世平 副教授		
云南大学	刘惟一 教授	重庆邮电学院	王国胤 教授
西南交通大学	杨 燕 副教授		

出版说明

改革开放以来,特别是党的十五大以来,我国教育事业取得了举世瞩目的辉煌成就,高等教育实现了历史性的跨越,已由精英教育阶段进入国际公认的大众化教育阶段。在质量不断提高的基础上,高等教育规模取得如此快速的发展,创造了世界教育发展史上的奇迹。当前,教育工作既面临着千载难逢的良好机遇,同时也面临着前所未有的严峻挑战。社会不断增长的高等教育需求同教育供给特别是优质教育供给不足的矛盾,是现阶段教育发展面临的基本矛盾。

教育部一直十分重视高等教育质量工作。2001年8月,教育部下发了《关于加强高等学校本科教学工作,提高教学质量的若干意见》,提出了十二条加强本科教学工作提高教学质量的措施和意见。2003年6月和2004年2月,教育部分别下发了《关于启动高等学校教学质量与教学改革工程精品课程建设工作的通知》和《教育部实施精品课程建设提高高校教学质量和人才培养质量》文件,指出“高等学校教学质量和教学改革工程”,是教育部正在制订的《2003—2007年教育振兴行动计划》的重要组成部分,精品课程建设是“质量工程”的重要内容之一,教育部计划用五年时间(2003—2007年)建设1500门国家级精品课程,利用现代化的教育信息技术手段将精品课程的相关内容上网并免费开放,以实现优质教学资源共享,提高高等学校教学质量和人才培养质量。

为了深入贯彻落实教育部《关于加强高等学校本科教学工作,提高教学质量的若干意见》精神,紧密配合教育部已经启动的“高等学校教学质量与教学改革工程精品课程建设工作”,在有关专家、教授的倡议和有关部门的大力支持下,我们组织并成立了“清华大学出版社教材编审委员会”(以下简称“编委会”),旨在配合教育部制定精品课程教材的出版规划,讨论并实施精品课程教材的编写与出版工作。“编委会”成员皆来自全国各类高等学校教学与科研第一线的骨干教师,其中许多教师为各校相关院、系主管教学的院长或系主任。

按照教育部的要求,“编委会”一致认为,精品课程的建设工作从开始就要坚持高标准、严要求,处于一个比较高的起点上;精品课程教材应该能够反映各高校教学改革与课程建设的需要,要有特色风格、有创新性(新体系、新内容、新手段、新思路,教材的内容体系有较高的科学创新、技术创新和理念创新的含量)、先进性(对原有的学科体系有实质性的改革和发展,顺应并符合新世纪教学发展的规律,代表并引领课程发展的趋势和方向)、示范性(教材所体现的课程体系具有较广泛的辐射性和示范性)和一定的前瞻性。教材由个人申报或各校推荐(通过所在高校的“编委会”成员推荐),经“编委会”认真评审,最后由清华大学出版社审定出版。

目前,针对计算机类和电子信息类相关专业成立了两个“编委会”,即“清华大学出版社计算机教材编审委员会”和“清华大学出版社电子信息教材编审委员会”。首批推出的特色精品教材包括以下三个系列:

(1) 高等学校教材·计算机应用——高等学校各类专业,特别是非计算机专业的计算

机应用类教材。

(2) 高等学校教材·计算机科学与技术——高等学校计算机相关专业的教材。

(3) 高等学校教材·电子信息——高等学校电子信息相关专业的教材。

清华大学出版社经过近二十年的努力，在教材尤其是计算机和电子信息类专业教材出版方面树立了权威品牌，为我国的高等教育事业做出了重要贡献。清华版教材经过二十多年的精雕细刻，形成了技术准确、内容严谨的独特风格，这种风格将延续并反映在特色精品教材的建设中。

总策划 李家强

策 划 卢先和 丁 岭

清华大学出版社教材编审委员会

E-mail: dingl@ tup. tsinghua. edu. cn

luxh@ tup. tsinghua. edu. cn

前 言

为什么写这本书

我们都在演绎 Java，都试图找出一条更好的学习 Java 语言的途径。

本书的构思是围绕着面向对象技术和面向对象编程语言这一主题展开的，因为这是学习 Java 时无法回避的问题。事实上，对象对于初学者是一个比较难以逾越的坎：对象怎么声明、创建、使用和清除？类如何设计？怎样编写出符合面向对象编程规范的 Java 程序？……像这样一些问题，真正深究起来的确有许多值得思考的地方。

作者结合自己的教学经验，把学习 Java 的过程分成三个阶段，即用 Java 语言写程序、写 Java 程序及写 OO（Object Oriented，面向对象）程序。

怎么讲？我们来看下面一个很简单的命题：用 Java 语言设计一个类，包含一个 int 型数值，类似于 int 类型的包装类 Integer；然后创建两个该类的实例，比较其 int 型数值是否相等。

在用 Java 语言写程序阶段，只要基本掌握类的概念及 Java 语言的基本语法即可写出如下程序段：

```
//用 Java 语言写程序阶段的版本
public class NewInteger{
    //声明变量
    int value;

    public static void main(String[] args){
        NewInteger obj1 = new NewInteger();
        obj1.value = 100;

        NewInteger obj2 = new NewInteger();
        obj2.value = 60;

        System.out.println(obj1.value == obj2.value);
    }
}
```

在用 Java 语言写 Java 程序阶段，把着眼点放在如何编写符合 Java 编程规范的程序。那么，我们可能会阅读 JDK 的源代码（在安装 JDK 目录下的 src.zip 中）以及其他一些专家编写的 Java 程序，模仿他们的编程风格，直到写出让别人一眼就能看出这就是地道的 Java 程序——“简直和 JDK 的源代码一模一样”应该是一种很高的评价吧！这时，前述的程序段或许要改写成如下的程序段：

```
//用 Java 语言写 Java 程序阶段的版本
final public class NewInteger{
    //将变量设为私有
```

```
private int value;

//利用构造方法参数对变量赋值
public NewInteger(int value){
    this.value=value;
}

//利用 intValue 获取值
public int intValue(){
    return value;
}

//直接仿照 JDK 中 Integer 类的 equals 方法设计
public boolean equals(Object obj){
    if (obj instanceof NewInteger){
        return value==((NewInteger)obj).intValue();
    }
    return false;
}

//用作调试, 发布时去掉
public static void main(String[] args){
    NewInteger obj1 = new NewInteger(100);
    NewInteger obj2 = new NewInteger(60);

    System.out.println(obj1.equals(obj2));
}
}
```

在用 Java 语言写 OO 程序阶段, 我们把着眼点放在如何编写符合 OO 规范的程序。那么, 在写程序之前应当熟悉架构、模式、UML 等许多概念, 并且有一种强烈的欲望要把这些概念真正落实到自己的编程实践中去。这时, 除了考虑 Java 程序的规则外, 我们可能还要关心代码的 OO 特性、效率、可复用性、成熟度等。例如, 我们可以找到很好的理由把前述例子的第二版再改写成如下的程序段:

```
//用 Java 语言写 OO 程序阶段的版本
final public class NewInteger{
    //将变量设为私有
    private int value;

    //利用构造方法参数对变量赋值
    public NewInteger(int value){
        this.value=value;
    }

    //利用 intValue 获取值
    public int intValue(){
        return value;
    }
}
```

```
//适合于同类型（需要转型）比较和不同类型比较
public boolean equals(Object obj){
    //巧妙利用 try-catch 语句，效率更高
    try{
        //做同类型比较，直接转型
        return value==(NewInteger)obj.intValue();
    }catch(ClassCastException e){
        //而利用 catch 子句对类型不符的情况进行处理
        return false;
    }
}

/* 设计重载 equals 方法：在大多数情况下为同类型（不需要转型）
   比较，增加此方法以减少开销*/
public boolean equals(NewInteger obj){
    return value==obj.intValue();
}

//用作调试，发布时去掉
public static void main(String[] args){
    NewInteger obj1 = new NewInteger(100);
    NewInteger obj2 = new NewInteger(60);

    System.out.println(obj1.equals(obj2));
}
}
```

相信读者可以看出这三个版本程序段在质量上的差别。

此外，笔者还分析了学习 Java 的一般规律，认为对学生的指导应该是从框架式的概念介绍到主干问题的深入理解再到细节问题的详尽掌握。

根据这一思路，在书稿的取材和策划过程中精心衡量主干与细节的关系，既防浮光掠影、浅尝辄止，又防事无巨细、只见树木不见森林。在内容组织上紧扣面向对象编程，并强调编程规范的重要性，对知识点溯本求源，力求给出合理的解释，做到言之有物，绝不含糊其辞，以达到使读者能“学什么像什么”的目的。在内容编排上尽可能做到由浅入深，即围绕着一个主题，先从基础谈起，让读者能基本掌握知识点，然后逐渐把问题引向深入，给学有余力者提供一个发展的空间。书中选用的例子是典型的但绝不是繁琐的，例子的设计也尽可能精致，做到有的放矢，避免粗制滥造。还有，书中所述都是笔者思考探索过的，绝不存在凑数式的资料堆砌。这些都是本书的特点，也是笔者写书的观点。

毋庸置疑，有关 Java 的好书已经很多。当然 Java 阵营也非常庞大，只要技术还在发展，或者心得有之，书还将继续写下去。本书的另一个作用，就是记录了自己从困惑到解惑的学习历程，把些许学习心得总结出来呈现给读者，与人分享知识本身就是一种乐趣。笔者把自己所思所想的东西基本整理出来，虽然不至于词不达意，但意犹未尽的恐怕有之——留点遗憾吧，或许这就是学习和研究的动力。

内容的组成

本书共分 9 章。

第 1~5 章均围绕对象展开论述, 建议按顺序阅读: 第 1~3 章作为 Java 体系的知识储备, 总是应该学会的, 但仅仅是基础而已, 笔者把它们作为学会用 Java 语言写程序阶段的教学内容; 第 4 章主要论述类及对象的使用, 第 5 章主要论述类的设计, 这两章是本书的重点, 它们将作为学会用 Java 语言写 Java 程序乃至写 OO 程序阶段的教学内容。

第 6~9 章的内容每章相对独立, 如果具备了 Java 基本语法基础即可按需阅读: 第 6 章讨论 Java 的线程, 第 7 章讨论 Java 的输入/输出, 第 8 章讨论 Java 的集合框架, 第 9 章讨论 Java 的图形用户界面。

本书约定

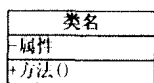
1. 正文斜体字约定

斜体 *type* 代表各种具体的类型的总称, 是具体类型的抽象, 如 *type*[] 中的 *type* 可以表示 *int* 型、*String* 型等, 而不是 *type* 型。类似的斜体有 *name*、*value* 等。

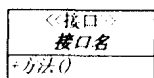
斜体加括号 (*default*) 表示默认访问权限, 实际上并没有这个关键字。

代码中的斜体字部分用来强调与论述有关的语句。

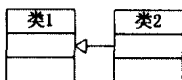
2. 图例符号约定



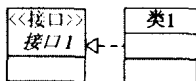
类图。一般使用正体, 表示具体类及其具体的属性、方法等。斜体类名表示抽象类, 如 *Shape*; 斜体方法名表示抽象方法, 如 *draw()*。“-”表示 *private* 访问权限, “+”表示 *public* 访问权限。



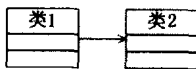
接口图。接口名用斜体表示, 如 *Refuellable*; 接口中的方法均为抽象方法, 用斜体表示, 如 *refuel()*。“+”表示 *public* 访问权限。



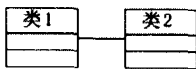
空心三角加实线表示继承关系。图中所示为类 2 继承 (或称扩展) 自类 1。接口与接口之间的继承关系也用此符号表示。



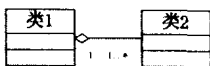
空心三角加虚线表示某个类实现了某个接口。图中所示为类 1 实现了接口 1。



箭头加实线表示单向关联。图中表示创建类 1 时与类 2 相关, 而创建类 2 时与类 1 无关。



实线表示双向关联。图中表示创建类 1 时与类 2 有关, 反之亦然。



空心菱形加实线表示聚合关系。图中表示一个 (1 的含义) 类 1 的对象拥有一个或多个 (1.. * 的含义) 类 2 的对象。



实心菱形加实线表示复合关系。图中表示一个 (1 的含义) 类 1 的对象拥有多个 (* 的含义) 类 2 的对象, 并且类 2 只有与类 1 在一起才有存在的意义。



标签, 起解释作用。

“aa” ⇨ “a” 空心箭头指某一种具体的操作。如图中表示为重新赋值: 把 *String* 类

型的值 aa 改为 a。

除此以外，图中画法有含义的均在图上给出了注释；图中没有注释而有些差别的画法，可能只是为了生动好看而已，没有其他特殊含义。

主要的阅读对象

本书的主要阅读对象是计算机及相关专业的大专院校学生，Java 初、中级编程人员以及对从事面向对象编程感兴趣者。相信他们都能从书中找到适合自己学习的内容，特别是喜欢边学习边思考的读者，相信在阅读时更容易与作者产生共鸣。

首先感谢您阅读了这本书，无论您有什么意见或建议，如果愿意，请不吝与笔者交流。您可以发电子邮件到 glob3d@163.com，笔者在此先谢了！

给学生的建议

教授本书的目标是培养学生达到学会用 Java 语言写 Java 程序的层次，善于思考和提问的学生将进入学会用 Java 语言写 OO 程序的阶段——当然，这些学生可能还要自学不少知识。

对于把本书作为教材学习的学生，建议能仔细阅读源程序中的注释，笔者花了大量精力为源程序添加了注释，其中包括对一个问题的多种解决方案、编译或运行时可能出现的异常以及程序段之间的关联等。如果把它们抽取出来单独描述将费力不讨好，所以到最后，笔者还是决定把它们留在源代码中论述。不过，笔者也有责任提醒学生不要忽略这些内容——有时，把一段程序中的注释连起来阅读，可能更容易发现笔者想要表达的意图。

还有一点个人的经验：笔者发现从计算机屏幕上阅读源代码比从书本上阅读源代码更加顺畅，即程序更容易看懂。如果您也有同感，不妨到出版社的网站上 (<http://www.tupwq.net>) 下载本书的源代码，学习的效率会更高。

关于选用编写源代码的平台，笔者建议初学时应选用最一般的文本编辑器，如 Windows 下的记事本工具，这种没有任何提示的工具对于培养您严谨的作风以及犀利的“法眼”将大有好处。

如果您读写程序已经没有太大问题，编程效率可能成为您最关心的问题，这时选用一个比较好的 IDE (Integrated Development Environment, 集成开发环境) 是比较明智的。笔者原来使用的是 JCreator Pro，觉得还不错。有学生推荐使用 Eclipse 平台——一个开放的可扩展的 IDE，据说很好，笔者正准备使用。不过，学语言时使用 JBuilder 似乎有些过分，容易导致花费在学平台的时间比花费在学语言本身的时间更多的现象，但这么说也不是绝对的。

给教师的建议

本书的授课时间建议安排 72 学时，如果能再加 36 学时辅导完成一个大作业则效果更佳。课内 72 学时安排如下：

- 第 1 章 2 学时
- 第 2 章 4 学时
- 第 3 章 6 学时
- 第 4 章 24 学时
- 第 5 章 16 学时

第 6 章 6 学时

第 7 章 3 学时

第 8 章 3 学时

第 9 章 8 学时

如果您选用书中的源代码在课堂上讲解，建议在备课时去掉源代码中的注释。因为如果不去掉的话，可能学生先关注答案，而不会真正用心思考问题，这样的效果不会太好。

如果您还有更好的案例和教学心得，同样欢迎来信交流。

致谢

本书由李发致主要编写，金佳和潘晓隽收集了书中的部分资料，陈建华和霍焱老师完成所有书稿的录入工作，郑建伟、张耿和沈钦完成所有书稿的校对工作。

数月的辛劳终于要告一段落——在搁笔之前，笔者非常衷心地对支持和鼓励过自己编写工作的人道声谢！

感谢所有曾经上过我的课的同学，你们的提问让我的大脑没有太早地退化萎缩。

感谢 SMIC 208——这是离我最近的软件开发俱乐部，这里的小兄弟们让我激情倍增。

感谢陈建华和霍焱老师，你们为本书的录入花费了大量的精力。

感谢清华大学出版社所有支持和合作出版本书的同志。

感谢上海交通大学技术学院的领导和同仁的大力支持。

小女宸宸，刚满三岁，在我十分疲倦的时候跑过来对我说：“爸爸，爸爸，别再做干活的机器啦！陪我玩一会儿吧。”——童言无忌，令人忍俊不禁。我愿意在此记下你的“大”话，愿你长大以后也有这么一份敬业精神，也愿我能拥有一份如你一般的闲情。

祝愿所有辛劳的人都有一个温馨的家！

李发致写于海上兼诗斋

2004 年 6 月

目 录

第 1 章 面向对象概述	1
1.1 面向过程与面向对象的方法论比较	1
1.1.1 系统建模方法	1
1.1.2 程序设计方法	3
1.1.3 面向过程方法的优势	5
1.1.4 面向对象方法的优势	6
1.1.5 面向过程与面向对象方法的应用优势	6
1.1.6 历史反思	6
1.2 面向对象技术的基本原则	7
1.3 面向对象技术的基本要素	8
1.3.1 封装性	9
1.3.2 继承性	10
1.3.3 多态性	10
1.3.4 面向对象概念和术语	11
1.4 面向对象的软件开发体系	13
1.4.1 面向对象框架下软件开发生命周期	13
1.4.2 面向对象相关技术的基本概念	16
1.4.3 面向对象程序设计的重要性	19
1.4.4 面向对象分析与设计的重要性	19
1.4.5 Java 的地位	19
1.5 思考与练习	20
第 2 章 Java 导论	21
2.1 Java 语言的特点	21
2.1.1 简单的面向对象程序设计语言	21
2.1.2 健壮安全的分布式语言	22
2.1.3 结构中立、可移植性强的解释型语言	22
2.1.4 高效能	22
2.1.5 支持多线程的动态语言	23
2.1.6 Java 与几种常用语言的比较	23
2.2 Java 执行环境	24
2.2.1 JVM 机制——Java 平台的基础	24
2.2.2 自动垃圾收集	25
2.2.3 保护域机制与沙箱模型	25

2.2.4	类加载器——启动引擎的点火器	28
2.3	JDK 的组成与安装设置	30
2.3.1	Java 平台的组成结构	30
2.3.2	SET PATH 和 SET CLASSPATH 的作用	31
2.4	典型例子及常见问题	33
2.4.1	编译器、解释器、小程序观察器和反编译器	33
2.4.2	简单的 Hello World! 例子	37
2.4.3	应用程序的主方法	43
2.4.4	小程序的运行	46
2.5	思考与练习	49
第 3 章	Java 的基本语法	50
3.1	类型、值与基本变量	50
3.1.1	基本概念	50
3.1.2	基本类型	52
3.1.3	引用类型	62
3.1.4	类型转换	71
3.2	运算符与表达式	76
3.2.1	算术运算符与表达式	76
3.2.2	关系运算符与表达式	77
3.2.3	逻辑运算符与表达式	77
3.2.4	位运算符与表达式	78
3.2.5	条件运算符与表达式	81
3.2.6	赋值运算符与表达式	81
3.2.7	instanceof 运算符	81
3.2.8	运算符优先级与赋值顺序的理解	81
3.2.9	基本类型所支持的操作运算	86
3.3	语句	87
3.3.1	编组语句	87
3.3.2	表达式语句	88
3.3.3	选择语句	89
3.3.4	循环语句	91
3.3.5	迭代与递归	93
3.3.6	控制转移语句	95
3.3.7	注释语句	98
3.4	思考与练习	100
第 4 章	Java 对象的生命周期	104
4.1	对象的声明与创建	104

4.1.1	基本术语	104
4.1.2	对象的声明	110
4.1.3	对象的创建	111
4.1.4	数组对象的声明和创建	114
4.2	对象的初始化	121
4.2.1	成员变量的默认初始化过程	121
4.2.2	类成员变量的初始化与静态代码块	121
4.2.3	实例成员变量的初始化与构造方法	127
4.2.4	程序健壮性的代价和编码启示	132
4.2.5	类成员变量与实例成员变量初始化的区别	133
4.2.6	继承链上相关类的加载顺序以及构造方法链的调用	139
4.3	对象的使用	144
4.3.1	多态的使用	144
4.3.2	对象的比较	153
4.3.3	对象的传递	160
4.3.4	内部类的使用	168
4.4	对象的清除	173
4.4.1	垃圾自动回收机制	173
4.4.2	垃圾回收概念模型	174
4.4.3	与垃圾回收有关的一些常用方法	175
4.5	思考与练习	178
第 5 章	对象设计的 Java 规范	185
5.1	好程序的基本条件	185
5.2	Java 语言的 Object 类及标准包	187
5.2.1	Object 类	187
5.2.2	java.lang 包	195
5.2.3	Java 标准包	211
5.3	Java 的异常处理机制	213
5.3.1	异常处理	213
5.3.2	核心代码与异常处理分离机制	215
5.3.3	引发异常及对异常处理的两条途径	217
5.3.4	Java 的内置异常类	225
5.3.5	设计异常类	226
5.3.6	断言语句	227
5.3.7	对异常不作为与作为的区别	230
5.4	包、接口、类与方法设计	233
5.4.1	包设计	233
5.4.2	制作 JAR 文件包	245

5.4.3	接口设计	247
5.4.4	抽象类设计	254
5.4.5	类设计	256
5.4.6	方法设计	272
5.4.7	设计模式	277
5.5	Java 编码的其他规范	278
5.5.1	命名规范	278
5.5.2	文件组织样式	279
5.5.3	增加程序可读性的一些建议	280
5.5.4	完整的例子	283
5.6	思考与练习	284
第 6 章	Java 的线程	288
6.1	线程的概念与 POSIX 标准	288
6.1.1	线程的概念	288
6.1.2	线程的 POSIX 标准	292
6.2	多线程的 Java 实现	293
6.2.1	线程的创建方法	293
6.2.2	线程的状态	306
6.3	互斥线程间的同步机制	314
6.3.1	多线程带来的冲突问题	314
6.3.2	共享资源合理使用的实现	320
6.3.3	按同步协调程度划分的线程间的关系	325
6.4	思考与练习	339
第 7 章	Java 的输入/输出	345
7.1	流输入/输出类的层次结构	345
7.1.1	流的概念	345
7.1.2	字节流	345
7.1.3	字符流	349
7.1.4	转换流	351
7.1.5	输入/输出类的分类	352
7.1.6	标准流	354
7.1.7	IOException 及其子类	354
7.2	输入/输出流类的应用	355
7.2.1	输入/输出流类的一般例子	355
7.2.2	典型输入/输出流类的组合应用	357
7.2.3	对象串行化	361
7.3	文件类的应用	364