

软件工程院

Java 与 模式



阎宏
飞思科技产品研发中心

编著
监制

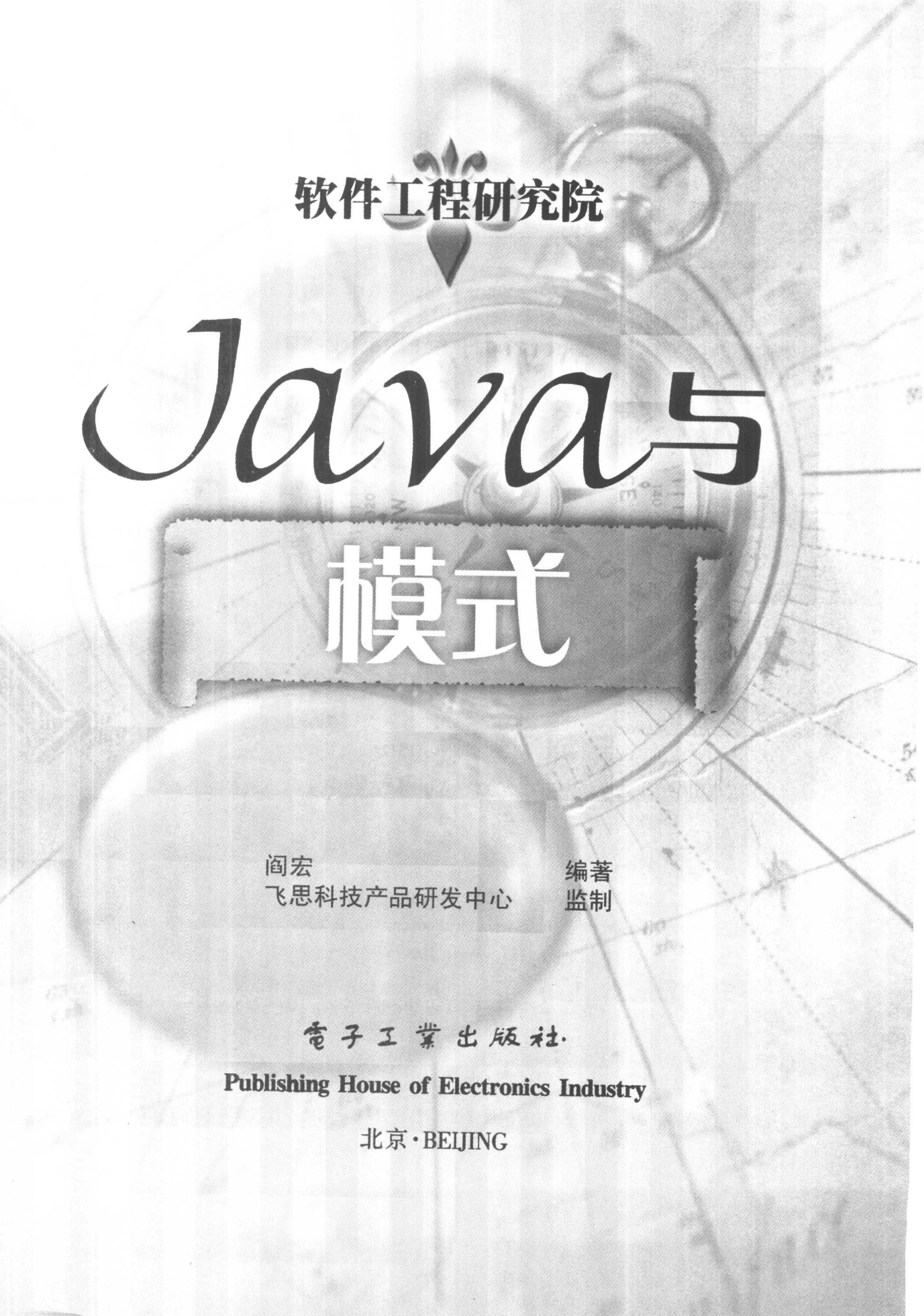
随书光盘附
赠书中实例
源代码



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
<http://www.phei.com.cn>





软件工程院

Java与 模式

阎宏
飞思科技产品研发中心

编著
监制

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING



内 容 简 介

本书是一本讲解设计原则以及最为常见的设计模式的实用教材，目的是为工作繁忙的 Java 系统设计师提供一个快速而准确的设计原则和设计模式的辅导。本书分为 55 章，每一个章节讲解一个编程模式，说明此模式的用意、结构，以及这一模式适合于什么样的情况等。每一个章节都附有多个例子和练习题，研习这些例子、完成这些练习题可以帮助读者更好地理解所讲的内容。大多数的章节都是相对独立的，读者可以打开本书，从任何一章开始读起。为了满足不同程度的读者的需要，本书提供了一些标为“专题”的章节，不阅读这些章节，对阅读本书的标准章节没有影响。读过本书之后，读者可以很有信心地在自己的系统设计工作中使用学过的内容，与同事交流面向对象的设计经验，并在事业发展上有更多的收获。

本书适合于 Java 程序员、架构设计师及设计模式爱好者，是一本实用的设计原则和设计模式的自学教材和使用手册。

未经许可，不得以任何方式复制或抄袭本书的部分或全部内容。

版权所有，侵权必究。

图书在版编目 (CIP) 数据

Java 与模式/阎宏编著. —北京: 电子工业出版社, 2002.10

(软件工程研究院)

ISBN 7-5053-8000-1

I. J... II. 阎... III. Java 设计语言—程序设计 IV.TP312

中国版本图书馆 CIP 数据核字 (2002) 第 068823 号

责任编辑: 郭晶 赵红梅

印 刷: 北京东光印刷厂

出版发行: 电子工业出版社 <http://www.phei.com.cn>

北京海淀区万寿路 173 信箱 邮编: 100036

经 销: 各地新华书店

开 本: 787×1092 1/16 印张: 65.25 字数: 1670.4 千字 附光盘 1 张

版 次: 2002 年 10 月第 1 版 2002 年 10 月第 1 次印刷

印 数: 6 000 册 定价: 88.00 元 (含光盘)

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。联系电话: (010) 68279077



序 1

——请接过这枝花

美国 Goldman Sacks 公司
朱天华

朱天华博士，1987年毕业于中国科技大学近代物理系，1992年和1994年于美国哥伦比亚大学获建筑学硕士学位（M.A.）和物理学硕士学位（M.Ph），1997年获理论天体物理学博士学位（Ph.D）；现任美国 Goldman Sacks 公司利率产品资深策略专家。

在人类所制造的工具中，有的可以将人送上月球，有的可以驾驭原子的火焰，但只有计算机软件，才能透彻地反照出人类的心智。

软件是人类心灵和智慧在虚拟空间中的投射。软件的性能就是人类能力的扩展，它的活动就是人类心智活动的反映。软件直接表达出设计者对目标的理解，对用户的期待，以及对自己的定位。

然而，人类的软件却是问题多多。面对不断变化的性能要求，软件系统往往过于僵硬，过于脆弱，不易复用，很难维护。一个设计师驾驭软件系统的能力，就像他统治一个虚拟王国，或者主宰一个虚拟世界的能力一样；而这些软件设计中的问题，其实就是人类自身心灵和智慧的不足在镜子中的倒影。它反映出的，不仅仅是技术的不足，还包括了科学、艺术、心理和哲学的不足。

在人类思想的百花园里，有一朵奇葩散发出神奇的幽香，这就是由建筑工程学家 Christopher Alexander 发展出的模式理论。它涵盖科学、心理、艺术和哲学，不仅适用于建筑工程学，而且适用于软件工程学以及任何其他工程学。Alexander 试图揭示的，是一条人类头脑与宇宙相互作用的永恒之道，它可以弥补发源于西方的现代科技、艺术和哲学的不足。

说到这里，忘记告诉你，这一散发着奇妙香气的曼妙花朵，嫁接自遥远的中国古代思想的园圃。

阎宏博士，凭借深厚的自然科学和中国传统哲学功底，以及多年来对模式理论的研究，将通过这本书向读者讲解软件设计模式的理论，它的简要历史，以及它与中国道家文化的渊源。这是一本实用的教材和手册，同时又是一本散发着思想火花的作品。当你把它放到书架上的时候，它应当占据一个独特的位置。

请接过这枝花。



序 2

北京科蓝软件系统有限公司

郑仁寰

郑仁寰博士于 1986 年从英国曼彻斯特大学获得计算机科学博士学位，现任北京科蓝软件系统有限公司技术总监。

电子计算机科学是极为年轻的学科，几十年来，新理论、新技术层出不穷。计算处理能力的爆炸式增长使人类的“胃口”越来越大，所研制软件系统的复杂性等问题也与之俱来。20 世纪 60~70 年代，已有了“软件危机”的提法。针对这类问题，各种灵丹妙药应运而生。面向对象法(object-oriented approach)是近十余年的佼佼者。其理论经过 Smalltalk、C++、Java 等编程语言的实现以及现代用户界面的成功普及，可以说是通过了“实践的检验”。

设计模式来源于建筑学。他山之石，可以攻玉。这一理论，在软件行业的成功，恐怕是其发明人也始料不及的。设计模式把面向对象的理念发挥到了极至，对许多重复出现的问题，提出了既优雅又实际的解决方案。如今，作为高级软件设计师不了解设计模式，不能熟练地运用几个主要的模式，说其孤陋寡闻，恐怕也不过分。

Java 编程语言自 1995 年问世以来，其成功好像任何编程语言都无法媲美。生逢其时（互联网的兴起）固然是一方面的原因，而 Java 吸收总结了前人的经验教训，反映了最新技术(the state of the art)，对其受到欢迎和采用，恐怕更为重要。例如，设计模式在 Java 的设计及功能中，有着举足轻重的作用。

十余年来，像西方一样，电脑新书的出版在国内市场也是爆炸式的，这些书出得仓促，寿命亦短（新版频繁，技术更新快）。令人遗憾的是，不少技术书或是语言枯燥，读起来令人味同嚼蜡；或是由于译者对所涉及的技术不甚通晓，加上校对错误，令人不知所云。近来，对中国软件业与印度相比不成气候，多有讨论，其对策往往涉及软件人才的培训、对英语的掌握、新书的翻译等问题。要想使中国软件业脱颖而出，没有一大批技术高超、爱业敬业、与时俱进的软件人才，近乎缘木求鱼。夜郎自大、浅尝辄止，都是大忌。新技术的全面深入引进，至为关键。

阎宏博士有多年软件系统设计、培训的第一手经验，从事软件开发项目管理之余，将其对设计模式的深入理解、实践心得著述成书，以飨中文读者。本书资料翔实、取法新颖、技术前沿、举例生动，既可帮助有志于掌握设计模式的软件设计师做自学、参考书，又适用于高校及专业培训机构选为教材。

前言

设计模式和设计原则已经成为面向对象的编程 (OOP), 以及面向对象的设计 (OOD) 的最新进展。设计模式和设计原则可以帮助 Java 设计师针对日常系统设计工作所遇到的很多设计问题给出结构合理、易于复用、易于维护的示范答案。本书向国内的 Java 程序设计师介绍这一强大的工具。

作者简介



阎宏, 1964 年出生于天津市。1987 年毕业于中国科技大学近代物理系, 1990 年于中科院理论物理所获得硕士学位, 1992 年获博士学位, 翌年赴日本京都大学进行博士后研究工作。

作者曾于美国花旗银行 (Citibank)、汤臣金融 (Thomson Financial)、奥本海默基金 (Oppenheimer) 等处供职, 进行了多年的软件开发、架构设计和技术管理工作。

欢迎读者与作者联系: JavaPatterns@yahoo.com, www.Web-EndsHere.com。

本书为何而写

促使作者写这本书的原因有很多, 最主要的原因有以下几个。

第一, 作者深深感到, 虽然市场上有很多介绍面向对象技术的书籍, 但是这些书籍中, 有一些过于集中在理论层面, 而有一些则过分地集中在具体层面上。

前者脱离软件工业的现状, 对做具体工作的系统设计师和程序员的帮助十分有限。

后者由于拘泥于细节, 很难帮助读者发展出全局的观念和视野。比如能够理解数据抽象化、继承、封装和多态性是非常重要的, 但是仅仅知道这些并不能使设计师设计出具有更好的复用性和易于维护的系统。

通过前几年 UML 知识的普及, 在读者群已经成熟的基础之上, 如果能有一本贯穿理论和实践的教材性书籍, 会对很多读者有益。

第二, 作者深深感到, 市场上没有一本以适合中国人的思维方式, 针对中国读者群讲解软件设计与设计模式的书籍, 这对大批的中国读者来说, 十分不公平。

在西方的思想里面, 分析一个结构直到把它分解为不可分割的单位, 再加以数量化的研究和表述, 是其典型的数理分析的套路。在西方科学文化中, 不能够精确化和数量化的结构不受重视。

而在中国传统的文化里面, 不能够精确化和数量化并不是研究的障碍。中国的阴阳思想可以灵活地描述任何的自然物、人造物和人类思想。

模式的理论有着深厚的中国道家思想渊源，设计模式实际上非常适合中国传统的思考方式。但是，很多中国的模式研究者们却似乎丢掉了自己的传统文化传承，像对待其他的舶来品一样，努力以西方的思维进行“重新”学习。

西方的作者往往由于自己对中国道家文化的理解水平所限，而在讲解模式理论的时候，有意无意地忽略它的道家背景。当这样的书籍充斥中国的模式图书市场的时候，中国的读者就会更加迷失在这一误区里面。

这使得作者感到一种沉重的责任感。

第三，市场上关于软件设计模式的书籍中确实有一部分是针对 Java 语言的，但是这些书籍在有些方面做得还不够。由于计算机图书的生命周期较短，就造成出版周期也很短，这使得一般的作者不愿意进行费时费力的调查研究工作。比如对于设计模式在 Java 语言 API 中的运用，以及设计模式是如何运用到真实的项目中去的，就较少有书籍做详细的介绍。

本书作者集中了很大的精力，对 Java 语言 API 中的设计模式做了详尽的综合、分析，并且从亲身参加的一些项目中抽取一些有代表性的问题，以专题讨论的方式介绍给读者。这可以使读者像亲临实地一样，体会设计模式是怎样应用到 Java 语言 API 中去的，以及是怎样在真实的软件项目中得到运用的。

本书为谁而写

Java 语言是现今最普及的纯粹 OOP 的编程语言，使用 Java 语言编程的程序师的平均素质也相对比较高。这些程序师往往不满足于只是实现程序功能的要求，他们常常想要在代码结构、编程风格，乃至解决问题的思考方式上不断进取和自我完善。设计原则与设计模式就是在大量的实践中总结和理论化之后优选的代码结构、编程风格及解决问题的思考方式。对模式的了解和掌握，是 Java 程序员提高自身素质的一个很好的方向。

这本书是 Java 程序师为 Java 程序员写的一本实用的设计原则和设计模式的自学教材和使用手册。

如果你是一位 Java 程序员，你以此为荣，并且希望能够了解 Java 语言 API 背后的设计思想和设计理念，以便更加了解这一语言，那么这本书就是为你而写的。

如果你是一位架构设计师，你以此为荣，并且希望能够进一步了解设计模式的起源，以及最知名的二十几个设计模式的用意、结构直到代码实现，以便能够改善自己的系统设计方案，那么这本书就是为你而写的。

如果你是一位设计模式的爱好者，你以此为荣，并且希望能够知道设计模式背后的设计原则和背后的哲学思想，那么这本书就是为你而写的。

本书的组织结构

本书的内容可以大体划分成两大部分：第一部分讲述了模式理论的起源、面向对象的设计原则；第二部分内容则是对具体的设计模式的讲解。

在第一部分内容中，本书首先对模式理论的起源做了简单但是比较全面的介绍。本书

特别对 Christopher Alexander 的模式理论与中国道家思想的联系做了哲学上的分析。然后，本书将几个最重要的面向对象的设计原则带给读者，这些原则都会在设计模式中具体体现出来。

在第二部分内容中，本书分别对最著名的几个创建模式、结构模式以及行为模式做了讲解。特别应当指出的是，本书的讲解有三大特点。

第一个特点，本书的讲解结合了具体的系统设计实例，比如单例模式和多例模式的讲解就结合了两个出现在真实的华尔街金融网站系统中的设计实例。读者可以将这些设计细节经过完善后使用到自己的系统设计中去。

第二个特点，本书的讲解结合了中国古代的神话传说、历史故事，内容贴切而生动活泼，可以给人以深刻的印象。

第三个特点，本书的讲解给出了大量的附图和可运行的代码，使读者可以一方面阅读文字，一方面欣赏插图，另一方面又可以将代码运行在 Java 环境中运行，并通过结果研究其结构。

怎样阅读本书

如果读者把本书当做一本手册，那么读者可以从书的任何一个章节开始阅读。本书的大多数章节都是与其他的章节相对独立、自成一体的，在少数依赖于其他章节的情况下，都会提醒读者首先阅读哪些章节。

如果将本书作为自学或教学的教材使用的话，则不一定要沿用模式的逻辑分类顺序。在下面就根据本书所包括的模式的难易、内在逻辑联系给出一个适合于自学成才和教学的顺序。

简单工厂模式（创建模式）：是最简单的模式，有助于引出工厂方法模式。

工厂方法模式（创建模式）：工厂方法模式在后面的许多模式都有应用，它在模式的讨论里屡见不鲜。了解工厂方法模式是了解所有模式的起点。

策略模式（行为模式）：策略模式在本书的讨论里频繁出现。早些理解策略模式有助于理解其他的模式。

装饰模式（结构模式）：装饰模式将是读者第一次接触到的模式的高雅之处。学会区别装饰模式和策略模式，也将是读者第一次学会区别看上去相似而实质不同的模式。

在学习了装饰模式之后，感兴趣的读者可以进一步阅读装饰模式在 Java I/O 中的应用，作为补充内容。

合成模式（结构模式）：合成模式在所有场合都有可能出现。合成模式经常与迭代子模式、责任链模式、解释器模式、访问者模式联合使用，因此，好好掌握合成模式有助于理解后面的有关模式。

迭代子模式（行为模式）：学习迭代子模式可以加深读者对合成模式的理解。本书还提供了关于 Java 语言对迭代子模式的支持的专题分析，供感兴趣的读者阅读。

模版方法模式（行为模式）：学习模版方法模式可以加深读者对策略模式以及简单工厂模式、工厂方法模式的理解。在迭代子模式里，模版方法模式经常被用到。

本书还提供了关于 Java Servlet 构造中的设计模式，主要是模版方法模式的专题研究。

抽象工厂模式（创建模式）：抽象工厂模式是创建模式里的第三个模式。读者回到这个模式，可以复习一下工厂方法模式。

在这一章后面的附录中，本书还给出了 Java 语言 API 中所使用的抽象工厂模式，供感兴趣的读者阅读。

建造模式（创建模式）：读者可以比较一下建造模式与抽象工厂模式的异同。

单例模式（创建模式）：单例模式经常用来实现抽象工厂模式。在单例模式的“相关模式”一节，对这一点有更详尽的描述。

本书为单例模式提供了一个专题研究章节，用来加深读者对这个重要模式在具体系统设计中的意义的了解。

多例模式（创建模式）：多例模式是对单例模式的推广。在学习了单例模式之后，学习多例模式，可以对隐藏在二者背后的设计思想有更深入的了解。

代理模式（结构模式）：读者在这里有机会学习代理模式怎样控制外界对一个对象的引用。代理模式也把读者引导到下一个模式——适配器模式。

本书利用了三章的篇幅，分别以代理模式的基本实现、智能引用代理和虚拟代理等具体种类详细地讲解了代理模式是怎样实现和应用到具体问题中去的。

适配器模式（结构模式）：读者可以学习和比较适配器模式与装饰模式、代理模式，以及后面的桥梁模式的异同。

在学习了适配器模式之后，感兴趣的读者可以进一步阅读适配器模式在 Java I/O 中的应用，作为补充内容。

桥梁模式（结构模式）：现在读者可以比较桥梁模式与适配器模式及代理模式的异同。

缺省适配模式（结构模式）：缺省适配模式是适配器模式的一种特例。在学习了普通的适配器模式之后，再学习缺省适配模式可以更好地体会两者的关系和区别。

调停者模式（行为模式）：学习调停者模式可以作为学习和理解观察者模式，以及 MVC（模型-视图-控制，Model-View-Controller）设计的准备工作。

观察者模式（行为模式）：读者可以在学习中发现调停者模式是怎样在观察者模式里发挥作用，实现 MVC 这一经典设计的。

读者如果使用过 Java 语言的事件处理机制的话，实际上就已经使用过观察者模式，只是没有察觉而已。本书除了使用一章的篇幅讲解观察者模式本身之外，还使用了三章的篇幅讲解观察者模式在几个著名的 Java API 中的应用。

MVC 模式（架构模式）：MVC 模式包含合成模式、策略模式、观察者模式等，可以是这些设计模式的联合运用。

责任链模式（行为模式）：学习责任链模式，读者可以比较责任链模式和观察者模式、调停者模式是怎么处理调用的。

在这一章后面，本书还提供了一个专题章节，介绍使用责任链模式在 Java 语言里对“击鼓传花”游戏进行模拟的内容。

备忘录模式（行为模式）：读者学习备忘录模式，并讨论此模式是怎样实现 undo 和 redo 的，并比较下面要介绍的命令模式的处理方法。

命令模式（行为模式）：命令模式可以有很多种使用方法，其中之一与备忘录模式有关，因此，在备忘录模式后面介绍。

本书还对 Swing 库中命令的撤销做了介绍。

原始模型模式（创建模式）：这可能是创建模式中最复杂的一个。原始模型模式经常与命令模式一起使用，这就是为什么在命令模式后面介绍的原因。

作为补充阅读，本书还对 JavaBean 的串行化和并行化做了介绍。

状态模式（行为模式）：读者可以通过学习状态模式来理解一个对象改变状态的另一种方式。

本书利用状态模式对《聊斋志异》中的崂山道士的故事做了分析。

访问者模式（行为模式）：访问者模式经常与合成模式、迭代子模式一起联合使用。

本书在介绍访问者模式之前，对双重分派的概念做了介绍。

享元模式（结构模式）：这是所有常见的模式中最复杂的一个模式。这一模式的应用在下面的解释器模式里介绍。

解释器模式（行为模式）：这是一个复杂的模式。学习这一模式最好是在学习了享元模式以及访问者模式之后。

门面模式（结构模式）：这是最后学习的一个模式。在学习了解释器模式之后，门面模式就比较好学。因为讲解门面模式的环境常常与讲解解释器模式的环境是一样的。

以上仅供自学和教学者参考。

鸣谢

没有多位朋友的鼎力相助，本书不可能面世。首先，作者的同事 Bo Y. Yu 阅读了本书的大部分文稿，并提出了非常有用的意见。此外，Daniel Yu、David W. Liu、Wilson Liu、Richard Tang、Luo Ding、Tony Feng 也阅读了本书的部分文稿，并提出了宝贵意见，同时蒋涛先生、刘晓融先生也提出了很多好的意见。他们的意见都已经反映到本书的内容之中。

每一本书都是编辑与作者共同创造的成果。电子工业出版社的编辑对本书的面世做了大量细致入微的工作。

作者的爱妻冯君也曾数遍阅读本书，并在本书的写作期间给予了大量的支持。没有她自始至终的支持和帮助，这本书同样是不可能面世的。

在本书出版之际，本人对以上各位的支持致以最诚挚的感谢。

参考文献与进一步阅读

除特别注明的参考文献之外，本书的各个章节还使用了以下的参考文献。这些参考文献都是关于软件设计模式和其他模式理论的著作，可供读者做进一步的阅读。

[GOF95] E.Gamma, R. Helm, R. Johnson, and J. Vlissides. Design Patterns – Elements of Reusable Object-Oriented Software. Addison–Wesley, 1995

[GOF95Z] E.Gamma, R. Helm, R. Johnson, and J. Vlissides. 《设计模式——可复用面向对象软件的基础》。李英军，马晓星，蔡敏，刘建中译。吕建审校。机械工业出版社，2000

[VLISSIDES98] John Vlissides. Pattern Hatching. Addison–Wesley, 1998

[GRAND98] Mark Grand. Patterns in Java. Vol. 1. John Wiley & Sons, Inc., 1998
[GRAND98-2] Mark Grand. Patterns in Java. Vol. 2. John Wiley & Sons, Inc., 1998
[GRAND98-3] Mark Grand. Patterns in Java. Vol. 3. John Wiley & Sons, Inc., 1998
[Cooper00] James W. Cooper. Java Design Patterns: A Tutorial. Addison–Wesley, 2000
[GOVONI99] Darren Govoni. Java Application Frameworks. Wiley Computer Publishing, 1999

[ALPERT98] Sherman R. Alpert, Kyle Brown, and Bobby Woolf. The Design Patterns SamlItalk Companion. Addison–Wesley, 1998

[STELTING02] Stephen Stelting, Olav Maassen. Applied Java Patterns. Prentice Hall, 2002

[J2EEPATT01] Deepak Alur, John Crupi and Dan Malks. Core J2EE Patterns. published by Prentice Hall, 2001

[METSKER02] Steven John Metsker. Design Patterns Java Workbook. published by Addison–Wesley, 2002

[ANTI99] William J. Brown, Hays W. McCormick III, Scott W. Thomas. AntiPatterns and Patterns in Software Configuration Management. Wiley, 1999

[SHALL02] Alan Shalloway and James R. Trott. Design Patterns Explained – A New Perspective on Object–Oriented Design. Addison – Wesley, 2002

[GOLD02] Brandon Goldfedder. The Joy of Patterns – Using Patterns for Enterprise Development. Addison – Wesley, 2002

[KIRK02] Kirk Knoernschild. Java Design – Objects, UML, and Process. Addison–Wesley, 2002

这些书籍都可以作为读者进一步阅读的方向。除了这些书籍之外，本书还向读者推荐以下的这些中文资料网站：

<http://www-900.ibm.com/developerWorks/cn/index.shtml>

<http://www.yesky.com>（程序方舟）

<http://www.chinajavaworld.com>

<http://www.cn-java.com>

<http://www.csdn.net>

<http://www.smiling.com>

<http://www.WebEndsHere.com> 本书的姊妹网站

目录

第一部分

第 1 章	模式的简史和形而上学.....	3
1.1	模式是什么.....	3
1.2	软件模式的简史.....	3
1.3	模式的起源.....	4
1.4	与道家思想的关系.....	5
1.5	软件的永恒之道.....	8
1.6	模式的要素.....	10
1.7	本书讲解模式的格式.....	11
第 2 章	统一建模语言 UML 简介.....	13
2.1	建造世界贸易中心.....	13
2.2	什么是 UML.....	14
2.3	UML 包括什么.....	15
2.4	类图.....	17
2.5	时序图.....	24
2.6	状态图.....	25
2.7	UML 及建模的工具.....	27

第二部分

第 3 章	软件的可维护性与可复用性... 33
3.1	软件系统的可维护性..... 33
3.2	系统的可复用性..... 37
3.3	老子论“不武”..... 40
第 4 章	“开-闭”原则 (OCP) 41
4.1	什么是“开-闭”原则..... 41
4.2	怎样做到“开-闭”原则..... 41
4.3	与其他设计原则的关系..... 43
4.4	策略模式对“开-闭”原则的支持..... 45
4.5	在其他设计模式中的体现..... 46
4.6	一个重构做法的讨论..... 49
第 5 章	专题: Java 语言的接口..... 53
5.1	什么是接口..... 53
5.2	为什么使用接口..... 54
5.3	Java 接口常见的用法..... 56

第 6 章	专题：抽象类.....	61
6.1	什么是抽象类	61
6.2	抽象类的用途	61
6.3	基于抽象类的模式和原则	64
6.4	什么时候才应当使用继承复用	64
第 7 章	里氏代换原则（LSP）	69
7.1	美猴王的智慧	69
7.2	什么是里氏代换原则	70
7.3	里氏代换原则在设计模式中的 体现	71
7.4	墨子论“取譬”	73
7.5	从代码重构的角度理解	74
第 8 章	依赖倒转原则（DIP）	85
8.1	为何而“倒转”	85
8.2	复用与可维护性的“倒转”	86
8.3	依赖倒转原则	86
8.4	怎样做到依赖倒转原则	88
8.5	Java 对抽象类型的支持	90
8.6	一个例子：账号、账号的种类和 账号的状态	92
8.7	墨子论“取周”	95
8.8	依赖倒转原则的优缺点	95
第 9 章	接口隔离原则（ISP）	97
9.1	什么是接口隔离原则	97
9.2	一个角色隔离原则的例子	98
9.3	定制服务的例子	100
第 10 章	合成/聚合复用原则 （CARP）	103
10.1	合成和聚合的区别	103
10.2	复用的基本种类	103
10.3	从代码重构的角度理解	105
第 11 章	迪米特法则（LoD）	109
11.1	迪米特法则的各种表述.....	109
11.2	狭义的迪米特法则	109

11.3	迪米特法则与设计模式.....	114
11.4	广义的迪米特法则.....	116
11.5	广义迪米特法则在类的设计上 的体现.....	117
11.6	广义迪米特法则在代码层次上 的实现.....	120
11.7	老子论“圣人之治”.....	120
第三部分		
第 12 章	简单工厂 (Simple Factory) 模式.....	127
12.1	工厂模式的几种形态.....	127
12.2	简单工厂模式的引进.....	128
12.3	简单工厂模式的结构.....	134
12.4	简单工厂模式的实现.....	136
12.5	简单工厂模式与其他模式的 关系.....	140
12.6	简单工厂模式的优点和缺点.....	142
12.7	简单工厂模式在 Java 中的应用.....	143
12.8	女娲抟土造人.....	146
第 13 章	工厂方法 (Factory Method) 模式.....	151
13.1	引言.....	151
13.2	工厂方法模式的结构.....	153
13.3	工厂方法模式在农场系统中 的实现.....	157
13.4	关于工厂方法模式的实现.....	162
13.5	Java 语言中工厂方法模式的例子...165	
13.6	工厂方法模式与其他模式的关系...168	
13.7	女娲举绳造人.....	170
13.8	其他的例子.....	170
第 14 章	抽象工厂 (Abstract Factory) 模式.....	179
14.1	引言.....	179
14.2	问题.....	181
14.3	抽象工厂模式的结构.....	184
14.4	在什么情形下应当使用抽象 工厂模式.....	189
14.5	抽象工厂模式在农场系统中 的实现.....	191

14.6	抽象工厂模式的另一个例子.....	196
14.7	“开 - 闭”原则.....	197
14.8	相关的模式与模式的实现.....	198
14.9	女娲造万物的故事.....	200
14.10	附录: Java AWT 的 Peer 架构 与抽象工厂模式.....	206
第 15 章	单例 (Singleton) 模式.....	209
15.1	引言.....	209
15.2	单例模式的结构.....	210
15.3	在什么情况下使用单例模式.....	215
15.4	单例类的状态.....	216
15.5	一个实用的例子: 属性管理器.....	217
15.6	Java 语言中的单例模式.....	223
15.7	专题: 不完全的单例类.....	226
15.8	相关模式.....	227
15.9	附录: 双重检查成例的研究.....	230
15.10	给读者的一点建议.....	234
第 16 章	专题: 单例模式与 MX 记录.....	235
16.1	问题与解决方案.....	235
16.2	目录服务与 MX 记录.....	237
16.3	JNDI 架构介绍.....	237
16.4	如何使用 JNDI 编程.....	238
16.5	系统设计.....	241
16.6	讨论.....	246
第 17 章	专题: 多例 (Multiton) 模式 与多语言支持.....	253
17.1	引言.....	253
17.2	多例模式.....	254
17.3	多语言项目的设计.....	258
第 18 章	专题: 序列键生成器与单例及 多例模式.....	267
18.1	问题.....	267
18.2	将单例模式应用到系统设计中.....	269
18.3	将多例模式应用到系统设计中.....	281
18.4	讨论.....	284
第 19 章	建造 (Builder) 模式.....	285
19.1	引言.....	285
19.2	建造模式的结构.....	286

19.3	建造模式的活动序列.....	290
19.4	建造模式的实现.....	291
19.5	众神造人的神话故事.....	295
19.6	JavaMail 中的建造模式.....	296
19.7	一个发送邮件的例子.....	298
19.8	在什么情况下使用建造模式.....	308
19.9	建造模式与其他模式的关系.....	309
第 20 章	原始模型 (Prototype)	
	模式.....	317
20.1	引言.....	317
20.2	变量、对象以及对象的引用.....	318
20.3	Java 对象的复制.....	319
20.4	原始模型模式的结构.....	323
20.5	模式的实现：深复制和浅复制.....	327
20.6	孙大圣的身外身法术.....	329
20.7	在什么情况下使用原始模型 模式.....	340
20.8	原始模型模式的优点和缺点.....	341
20.9	原始模型模式与其他模式 的关系.....	341
第 21 章	专题：JavaBean 的“冷藏” 和“解冻”.....	345
21.1	什么是“冷藏”和“解冻”.....	345
21.2	什么可以“冷藏”.....	346
21.3	Beans.instantiate()方法.....	348
21.4	怎样在 JSP 中使用 Beans.Instantiate()方法.....	353
21.5	与装饰模式的关系.....	354
	第四部分	
第 22 章	适配器 (Adapter) 模式.....	359
22.1	引言.....	359
22.2	类的适配器模式的结构.....	360
22.3	类的适配器模式的效果.....	361
22.4	对象的适配器模式的结构.....	362
22.5	在什么情况下使用适配器模式.....	364
22.6	Iterator 与 Enumeration.....	364
22.7	利用适配器模式指方为圆.....	367
22.8	适配器模式在架构层次上 的应用.....	370

22.9	关于模式实现的讨论.....	373
22.10	适配器模式与相关的模式.....	373
第 23 章	缺省适配 (Default Adapter)	
	模式.....	379
23.1	鲁智深的故事.....	379
23.2	WindowAdapter 是缺省适配 模式.....	381
23.3	模式的结构.....	384
23.4	在什么情况下使用本模式.....	385
23.5	模式的实现.....	385
23.6	J2SE 中的缺省适配模式.....	386
23.7	一个例子.....	387
第 24 章	专题：XMLProperties 与 适配器模式.....	391
24.1	引言.....	391
24.2	SAX2 浏览器.....	393
24.3	开始 XMLProperties 项目.....	398
第 25 章	合成 (Composite) 模式.....	409
25.1	对象的树结构.....	409
25.2	介绍合成模式.....	411
25.3	安全式和透明式的合成模式.....	412
25.4	安全式的合成模式的结构.....	413
25.5	透明式的合成模式的结构.....	416
25.6	合成模式的实现.....	419
25.7	道士的故事.....	420
25.8	一个绘图的例子.....	421
25.9	AWT 库中的例子.....	428
25.10	合成模式与其他模式的关系.....	429
第 26 章	装饰 (Decorator) 模式.....	435
26.1	引言.....	435
26.2	装饰模式的结构.....	435
26.3	装饰模式应当在什么情况 下使用.....	441
26.4	孙大圣的故事.....	441
26.5	使用装饰模式的优点和缺点.....	442
26.6	模式实现的讨论.....	443
26.7	装饰模式与其他模式的关系.....	446
26.8	实例：GrepReader.....	447
26.9	一个例子：发票系统.....	451

26.10	附录：关于适配器模式与装饰模式的对话	464
第 27 章	专题：设计模式在 Java I/O 库中的应用	469
27.1	引言	469
27.2	Java I/O 库的设计原则	471
27.3	装饰模式的应用	472
27.4	半透明的装饰模式	480
27.5	适配器模式的应用	483
27.6	从 byte 流到 char 流的适配	489
第 28 章	代理（Proxy）模式	495
28.1	引言	495
28.2	代理的种类	496
28.3	代理模式的结构	498
28.4	代理模式的时序	501
28.5	Java 2.0 对代理模式的支持	502
28.6	高老庄悟空降八戒	506
28.7	代理模式的优点和缺点	508
28.8	代理模式的实现	509
28.9	代理模式与其他模式的关系	509
第 29 章	专题：智能引用代理	513
29.1	问题	513
29.2	系统设计	513
29.3	系统的时序	516
29.4	系统的源代码	516
第 30 章	专题：虚拟代理的例子	521
30.1	问题	521
30.2	系统的要求	521
30.3	系统的设计	522
30.4	系统的时序	522
30.5	系统的源代码	523
第 31 章	享元模式（Flyweight Pattern）	527
31.1	引言	527
31.2	单纯享元模式的结构	528
31.3	复合享元模式的结构	532
31.4	模式的实现	537
31.5	一个咖啡摊的例子	543
31.6	咖啡屋的例子	548

31.7	享元模式应当在什么情况下使用	553
31.8	从代码重构的角度看享元模式	553
31.9	享元模式的优点和缺点	555
31.10	相关的模式	555
第 32 章	门面（Facade）模式	561
32.1	引言	561
32.2	门面模式的结构	563
32.3	门面模式的实现	563
32.4	在什么情况下使用门面模式	564
32.5	迪米特法则（LoD）	564
32.6	一个例子	565
32.7	使用门面模式的设计	568
32.8	Session 门面模式	572
第 33 章	桥梁（Bridge）模式	575
33.1	引言	575
33.2	桥梁模式的结构	577
33.3	Java 语言中的 Peer 架构	579
33.4	驱动器和 JDBC 驱动器	583
33.5	从重构的角度考察	584
33.6	桥梁模式和其他模式的关系	590
33.7	八戒“转世投胎”的故事	592
33.8	关于桥梁模式的实现	594
33.9	在什么情况下应当使用桥梁模式	595

第五部分

第 34 章	不变（Immutable）模式	603
34.1	引言	603
34.2	不变模式的结构和实现	603
34.3	不变模式在 Java 语言中的应用	605
34.4	不变模式的优点和缺点	606
34.5	不变模式与享元模式的关系	606
34.6	一个例子：复数类	607
第 35 章	策略（Strategy）模式	621
35.1	引言	621
35.2	模式的实现	623
35.3	Java 语言内部的例子	624
35.4	排序策略系统	626

35.5	一个例子：图书折扣的计算.....	627
35.6	在什么情况下应当使用 策略模式.....	632
35.7	策略模式的优点和缺点.....	632
35.8	策略模式与其他模式的关系.....	633
35.9	设计原则的讨论.....	635
35.10	诸葛亮的锦囊妙计.....	635
第 36 章	模版方法 (Template Method) 模式.....	641
36.1	引言.....	641
36.2	模版方法模式的结构.....	641
36.3	“好莱坞原则”.....	643
36.4	一个例子.....	644
36.5	继承作为复用的工具.....	647
36.6	Java 语言里面使用过的 模版方法模式.....	648
36.7	模版方法模式中的方法.....	650
36.8	模版方法模式在代码重构 中的应用.....	651
36.9	重构的原则.....	655
36.10	西天取经的八十一难.....	656
第 37 章	专题：Servlet 技术中 的模式.....	659
37.1	Servlet 技术介绍.....	659
37.2	模版方法模式的使用.....	661
37.3	观察者模式的应用.....	665
第 38 章	观察者 (Observer) 模式.....	671
38.1	引言.....	671
38.2	观察者模式的结构.....	672
38.3	另一种实现方案.....	676
38.4	Java 语言提供的对观察者 模式的支持.....	679
38.5	菩萨的守瓶龟.....	686
38.6	Java 中的 DEM 事件机制.....	687
38.7	观察者模式与其他模式的关系.....	688
第 39 章	专题：观察者模式与 AWT 中的事件处理.....	691
39.1	引言.....	691
39.2	DEM 的结构.....	691

第 40 章	专题：观察者模式与 SAX2 浏览器.....	705
40.1	引言.....	705
40.2	SAX2 是怎么工作的.....	706
40.3	ContentHandler 接口.....	706
40.4	怎样实现 ContentHandler 接口.....	708
40.5	怎样使用 ContentHandler.....	710
40.6	缺省适配模式与 DefaultHandler.....	711
40.7	简单工厂模式的应用.....	716
第 41 章	专题：观察者模式与 Swing 定时器.....	717
41.1	为什么需要定时器.....	717
41.2	Utility 定时器与 Swing 定时器 的区别.....	717
41.3	使用 Swing 定时器的方法.....	719
41.4	观察者模式的应用.....	720
41.5	单例模式的应用.....	724
第 42 章	专题：MVC 模式与用户输入 数据检查.....	729
42.1	什么是 MVC 模式.....	729
42.2	MVC 是架构模式.....	731
42.3	MVC 模式在 J2EE 技术中 的应用.....	733
42.4	从代码重构的角度查看.....	735
42.5	用户输入检查与 MVC 模式.....	736
42.6	SQL 注射.....	738
第 43 章	迭代子 (Iterator) 模式.....	743
43.1	引言.....	743
43.2	迭代子模式的结构.....	745
43.3	迭代子模式的实现.....	755
43.4	迭代子模式的优点和缺点.....	757
43.5	一个例子.....	757
43.6	迭代子模式与其他模式的关系.....	763
第 44 章	专题：Java 对迭代子 模式的支持.....	769
44.1	Java 聚集.....	769
44.2	Java 聚集中的迭代子.....	771
44.3	java.util.Iterator 接口.....	772
44.4	Java 迭代子接口 ListIterator.....	775

44.5	Enumeration 与 Iterator	780	第 49 章	备忘录 (Memento) 模式	869
44.6	其他相关模式	782	49.1	备忘录模式的结构	869
第 45 章	责任链 (Chain of Responsibility) 模式	785	49.2	备忘录模式的白箱实现	870
45.1	从击鼓传花谈起	785	49.3	双重接口及其在 Java 语言中的实现	875
45.2	责任链模式的结构	785	49.4	备忘录模式的黑箱实现	878
45.3	纯的与不纯的责任链模式	788	49.5	负责人角色的增强	883
45.4	Java 1.0 版的 AWT 事件处理机制	789	49.6	备忘录模式与多重检查点	884
45.5	《红楼梦》中击鼓传花的故事	790	49.7	“自述历史”模式	890
45.6	Java 系统的解	792	49.8	“假如”协议模式	894
45.7	责任链模式的实现	798	49.9	备忘录模式与其他模式的关系	895
45.8	对象的树结构	799	49.10	备忘录模式的应用	896
45.9	DHTML 中的事件处理	800	49.11	使用备忘录模式的优点和缺点	898
45.10	责任链模式与其他模式的关系	804	49.12	观世音甘泉活树的故事	899
第 46 章	专题: 定时器与击鼓传花	809	第 50 章	状态 (State) 模式	907
46.1	线程	809	50.1	引言	907
46.2	定时器	810	50.2	状态模式的结构	907
46.3	击鼓传花	812	50.3	曾侯乙编钟的描述	909
第 47 章	命令 (Command) 模式	823	50.4	状态模式的效果	910
47.1	引言	823	50.5	在什么情况下使用状态模式	911
47.2	命令模式的结构	824	50.6	关于模式实现的讨论	911
47.3	玉帝传美猴王上天	827	50.7	一个状态模式的应用例子: TCP	912
47.4	Java 语言内的例子: AWT 的事件处理	828	50.8	状态模式在绘图软件中的应用	914
47.5	一个例子: 创世纪系统	828	50.9	用户登录子系统	915
47.6	一个例子: AudioPlayer 系统	833	50.10	状态模式与策略模式的区别	921
47.7	增加宏命令功能	839	第 51 章	专题: 崂山道士与状态模式	923
47.8	模式的实现	841	51.1	有状态的墙	923
47.9	在什么情况下应当使用命令模式	842	51.2	异常逻辑	923
47.10	使用命令模式的优点和缺点	843	51.3	从代码重构的角度考察	924
47.11	命令模式与其他模式的关系	843	第 52 章	专题: 单分派和多分派	929
第 48 章	专题: Swing 库中的命令撤销和恢复	853	52.1	分派的概念	929
48.1	在视窗系统中使用命令模式	853	52.2	双重分派	933
48.2	Swing 的基本撤销功能	853	52.3	未来的 Java 语言	940
48.3	一休论禅的文字框	861	第 53 章	访问者 (Visitor) 模式	945
			53.1	引言	945
			53.2	访问者模式的结构	947