

一本值得反复研读的书……

研  
磨

设计模式

陈臣  
王斌  
著

GOF 设计模式细细研磨中……

清华大学出版社

## 作者简介



陈 臣

10年Java/JavaEE开发经验，高级系统架构师，功力深厚，技术精湛，精通Java/JavaEE相关技术和多种开源框架，尤其擅长系统分析和架构设计。从事过专业的中间件研发，包括基于组件的Web页面框架、基于WFMC的工作流中间件、类似于Hibernate的ORM框架等；参与或主持了多个大中型的企业级应用项目，拥有多年项目经理、技术部经理的管理经验。

个人博客：

<http://chjavach.javaeye.com>



王 斌

从事Java/JavaEE开发5年，系统架构师，精通EJB、Struts、Spring、Hibernate、iBatis等框架技术，擅长设计模式和Eclipse插件开发。作为架构小组骨干，参与了国旅电子商务平台、南王酒庄等多个项目的开发，开发并维护有constance4j、myxstream、SimpleMapping等多个公司内部开源框架，深得多个项目组好评。

# 研磨设计模式

陈臣 王斌 著

清华大学出版社

北京

## 内 容 简 介

本书完整覆盖 GoF 讲述的 23 个设计模式并加以细细研磨。初级内容从基本讲起，包括每个模式的定义、功能、思路、结构、基本实现、运行调用顺序、基本应用示例等，让读者能系统、完整、准确地掌握每个模式，培养正确的“设计观”；中高级内容则深入探讨如何理解这些模式，包括模式中蕴涵什么样的设计思想，模式的本质是什么，模式如何结合实际应用，模式的优缺点以及与其他模式的关系等，以期让读者尽量去理解和掌握每个设计模式的精髓所在。

本书在内容上深入、技术上实用，和实际开发结合程度很高，书中大部分的示例程序都是从实际项目中简化而来，因此很多例子都可以直接拿到实际项目中使用。如果你想要深入透彻地理解和掌握设计模式，并期望能真正把设计模式应用到项目中去，那么这是你不可错过的一本好书。

本书难度为初级到中级，适合于所有开发人员、设计人员或者即将成为开发人员的朋友。也可以作为高校学生深入学习设计模式的参考读物。

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。

版权所有，侵权必究。侵权举报电话：010-62782989 13701121933

### 图书在版编目（CIP）数据

研磨设计模式 / 陈臣，王斌 著. —北京：清华大学出版社，2011.1

ISBN 978-7-302-23923-9

I. ①研… II. ①陈… ②王… III. ①Java 语言—程序设计 IV. ①TP312

中国版本图书馆 CIP 数据核字(2010)第 192952 号

责任编辑：栾大成

责任校对：

责任印制：徐俊伟

出版发行：清华大学出版社

地 址：北京清华大学学研大厦 A 座

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

邮 编：100084

社 总 机：010-6279975

邮 购：010-62786544

投稿与读者服务：010-62795954, [jsjic@tup.tsinghua.edu.cn](mailto:jsjic@tup.tsinghua.edu.cn)

质量反馈：010-62772015, [zhiliang@tup.tsinghua.edu.cn](mailto:zhiliang@tup.tsinghua.edu.cn)

印 刷 者：

装 订 者：

经 销：全国新华书店

开 本：188 × 260

印 张：50

字 数：1248 千字

版 次：2011 年 1 月第 1 版

印 次：2011 年 1 月第 1 次印刷

印 数：1~5000

定 价：89.00 元

---

产品编号：039306-01

# 前言

## 创作背景

软件开发越来越复杂，对软件设计的要求也越来越高，而软件设计和架构的入门功夫就是深入理解和掌握设计模式。因此，设计模式的重要性不言而喻。

很多朋友认识到了设计模式的重要性，也看了很多的书籍和资料，但是，常听到这样的抱怨：“设计模式的书我看了不少，觉得都看懂了，就是不知道在实际开发中怎么运用这些设计模式”，从而认为设计模式是“看上去很美的花拳绣腿”。

其实不然，造成这种情况的原因就在于：这些朋友对设计模式的理解不到位，自己感觉懂了，其实还差很远，并没有“真正”理解和掌握设计模式。

市面上有不少设计模式方面的书籍，但对一般的学习者而言，要么是太深，看得云里雾里的，比如 GoF 的著作《设计模式——可复用面向对象软件的基础》，很经典，但是能吃透的人少；要么就是太浅，看了跟没看差不多，也就是介绍一下每个设计模式，告诉你这就是某某设计模式，虽然语言很生动但是实在没货，看完也不知道怎么运用，就像是带领大家摸到了设计模式的大门口，却不告诉你怎么进去一样，其根本原因还是讲得太浅，跟实际的应用有太大的差距。

对于所有想要真正理解和掌握设计模式的朋友，其实需要这样的书籍：

- 理论全面、准确，难度适中；
- 讲解深入浅出、浅显易懂；
- 理论联系实际应用，对于晦涩的理论，应有相应的示例；
- 示例最好来自实际应用，而不是来自虚拟的场景；
- 示例最好相对完整，而不是片段代码，以利于学习和应用。

这也是本书写作的目的，希望能够帮助更多的朋友早日修成设计模式的正果。

经过多年的准备和一年的写作，以及各层次读者的多轮试读意见和建议汇总，最终成书，我们可以这样说：这是一本诚意十足的书，敬请您的评鉴！

本书的试读人员包括：从还没有参加工作的学生，一直到工作 7 年的人员；职务覆盖普通的程序员、项目经理、高级系统架构师、技术部的经理；两位作者本身从事开发工作的年限，一位超过 10 年，一位超过 5 年。

试读的结果：工作经验在 1 年以下的朋友，能正常理解和掌握初级部分的内容，能部分理解中高级部分的内容；工作经验在 1~2 年的朋友，基本上能全面理解，但是领悟尚有不足；工作经验在 2~5 年的朋友，能够正常理解和掌握，基本达到本书写作的意图；工作经验在 5 年以上的朋友，主要是弥补以前较少用到的部分，使知识更加系统化和全

面化，另外把本书当作一本工具参考书，案头必备。

## 本书内容

本书完整覆盖 GoF 的著作《设计模式——可复用面向对象软件的基础》一书所讲述的 23 个设计模式。

- 初级内容：从基本讲起，包括每个模式的定义、功能、思路、结构、基本实现、运行调用顺序、基本应用示例等，让读者能系统、完整、准确地掌握每个模式，培养正确的“设计观”。
- 中高级内容：深入探讨如何理解这些模式、模式中蕴涵什么样的设计思想、模式的本质是什么、模式如何结合实际应用、模式的优缺点，以及和其他模式的关系等，以期让读者尽量去理解和掌握每个设计模式的精髓所在。

本书在内容上深入、技术上实用，和实际开发结合程度很高，书中大部分的示例程序都是从实际项目中简化而来，因此很多例子都可以直接拿到实际项目中使用。如果你想要深入透彻地理解和掌握设计模式，并期望能真正把设计模式应用到项目中去，那么这是你不可错过的一本好书。

## 本书特色

- 本书有很多独到的见解和精辟的总结，能写出一些人所不敢写、不能写的内容，是一本“真正有货”的书。
- 本书大部分示例程序都来自真实的项目应用，让你真正理解和掌握设计模式，尽量做到“从实际项目中来，再应用到实际项目中去”。
  - ◆ 本书涉及的实际应用，包括但不限于：
    - ◆ 代码生成的应用工具（独立应用）；
    - ◆ 日志管理平台（来自于基础平台）；
    - ◆ 缓存管理（来自于基础平台）；
    - ◆ 订单处理（来自于 CRM 系统）；
    - ◆ 导出数据的应用框架（来自于 SCM）；
    - ◆ 组织机构管理（来自于基础平台）；
    - ◆ 大数据量访问（很多系统都有）；
    - ◆ 水质监测系统（独立应用）；
    - ◆ 工资管理（来自于 HRM 系统）；
    - ◆ 商品管理（来自于电子商务系统）；
    - ◆ 登录控制（来自于 OA 系统）；
    - ◆ 报价管理（来自于 CRM 系统）；
    - ◆ 在线投票系统（来自于 OA 系统）；

- ◆ 仿真系统（来自于 Workflow 系统）；
- ◆ 权限管理（来自于基础平台）；
- ◆ 配置文件管理（来自于基础平台）；
- ◆ 奖金核算系统（来自于 HRM 系统）；
- ◆ 费用报销管理（来自于 OA 系统）；
- ◆ 客户管理（来自于 CRM 系统）。

说明：OA（Office Automation）办公自动化

CRM（Customer Relationship Management）客户关系管理

HRM（Human Resource Management）人力资源管理

SCM（Supply Chain Management）供应链管理

Workflow 工作流

本书探讨了很多应用设计模式来解决实际项目中的问题

本书涉及的实际问题，包括但不限于：

- ◆ 如何实现可配置；
- ◆ 如何实现同时支持数据库和文件存储的日志管理；
- ◆ 如何实现缓存以及缓存的管理；
- ◆ 如何实现用缓存来控制多实例的创建；
- ◆ 如何处理平行功能；
- ◆ 如何实现参数化工厂；
- ◆ 如何应用工厂实现 DAO；
- ◆ 如何实现可扩展工厂；
- ◆ 如何实现原型管理器；
- ◆ 如何实现 Java 的静态代理和动态代理；
- ◆ 如何实现多线程处理队列请求；
- ◆ 如何实现命令的参数化配置、可撤销的操作、宏命令、队列请求和日志请求；
- ◆ 如何实现双向迭代；
- ◆ 如何实现带策略的迭代器；
- ◆ 如何实现翻页迭代；
- ◆ 如何实现树状结构和父组件引用；
- ◆ 如何检测环状结构；
- ◆ 如何实现通用的增删改查；
- ◆ 如何实现容错恢复机制；
- ◆ 如何模拟工作流来处理流程；

- ◆ 如何实现对象实例池；
  - ◆ 如何实现自定义语言的解析；
  - ◆ 如何实现既简单又通用的 XML 读取；
  - ◆ 如何实现功能链，实现类似于 Web 开发中 Filter 的功能；
  - ◆ 如何实现模拟 AOP 的功能；
  - ◆ 如何为系统加入权限控制；
  - ◆ 如何自定义 I/O 装饰器；
  - ◆ 如何实现通用请求处理框架。
- 本书的示例程序基本上都是带着客户端测试代码的，可直接运行，不是片段代码，更有利于大家整体学习和理解。

## 读者定位

本书难度为初级到中级，适合于所有开发人员、设计人员或者即将成为开发人员的朋友；也可以作为高校学生深入学习设计模式的参考读物。

我们强烈建议您认真阅读和学习本书的内容，全面、准确、深入、实用的内容定会有助于您凤凰涅槃般地实现技术升华，请相信。

## 阅读指南

本书假定您懂一些基本的 Java 知识，并具备一定的开发经验。

### 1. 对于初学设计模式的朋友

如果对常见面向对象的设计原则不太熟悉的话，请先参看附录 A。

如果对 UML 不太熟悉的话，请先参看附录 B。

然后开始看第 1 章，学习设计模式的一些基础知识，了解本书的整体大纲。

接下来就可以从前到后，循序渐进地学习每个设计模式。对每个模式建议先认真学习场景问题和解决方案两个部分，切实掌握每个模式标准的结构、实现和基本的应用。对于模式讲解中简单的内容也可以先看，但是对于后面较为复杂的内容，可以先不看，等到技术和经验积累到一定程度的时候，再循序渐进地向后学习。

### 2. 对于已有一定的开发经验和设计经验的朋友

首先应该从场景问题和解决方案看起，对于其中已会的内容权当复习，对于不会的内容，相当于是在查漏补缺，先把基础部分夯得全面、扎实。

然后再认真学习模式讲解部分，并结合实际的开发经验来思考，看看如何应用模式来解决实际问题、如何把模式应用到实际的项目中去，再深入地思考模式的本质和设计思想，掌握模式的精髓，这样才能真正做到在实际开发中自如地应用设计模式。

### 3. 对所有的朋友

这不是一本随便看看，读完一遍就可以扔掉的书籍，需要反复研读。因此，第一次阅读本书时，如果发现有些不理解的内容也不要紧，可以在今后的学习和工作中，反复参阅本书，以加深对设计模式的理解，获取设计灵感，并把设计模式切实应用到实际项目中去。

### 4. 善意提醒

在实际开发和设计中，要遵循简单设计的原则，不要为了模式而模式，不要过度设计，要在合适的地方应用合适的设计模式来解决问题。

这对于初学者尤其要注意，因为刚学会一个东西，总是跃跃欲试，急于一显身手，往往容易造成设计模式的误用。

## 本书约定

### 1. 本书的知识边界

由于关于设计的知识过于博大精深，因此本书“集中火力”，重点讲述 GoF 著作中涉及的 23 个设计模式本身，以及和这些设计模式相关的应用内容。

没有过多涉及：面向对象设计原则、重构、系统架构设计、JavaEE（原 J2EE，也有简写成 JEE）设计模式或是其他分类的设计模式（如 EJB 设计模式）等内容，原因可以参见附录 A。也没有过多讲述 UML，有需要的朋友请参看附录 B。

对于每章涉及的实际应用，描述也非常简略，只抽取讲述模式需要的一点东西。因为这些实际应用的东西，对于有相应开发经验的朋友多说无益，一提就明白；对于没有相应经验的朋友，多讲一点也未见得能多明白多少，反而冲淡了设计模式这个主题。

### 2. 本书的示例和代码

本书的示例虽然大都来自实际应用，但是经过相当的删除简化和重新组合；另外一点，为了突出设计模式这个主题，因此代码并不是按照实际应用那样来严格要求，很多例外处理、数据检测等都没有做，逻辑也未见得那么严密；还有一点，在实际的开发中，很可能是多个模式组合来实现某个功能，但是本书为了示例某个模式，让重点突出而避免读者迷惑，会选择重点示例某个模式的用法，而简化或去掉其他模式。

如果要把这些示例代码在实际应用中使用，还需要对这些代码进行加工，使其更加严谨，才能达到工业级的要求。

## 真诚致谢

首先要感谢清华大学出版社的员工，他们给予本书很多中肯的意见和建议，对本书从选题到出版的各个环节，都给予了大量的指导和帮助。

其次要感谢张开涛先生，他对本书的内容提出了很多有用的意见和建议。

然后要感谢魏源先生和蔡抒杨先生，他们对本书内容的完善也给出了很好的建议。

接下来，按照惯例，应该感谢家人、感谢朋友、感谢北京的漫天风沙和明媚阳光，

以及那可爱的阳台和小巧的书桌，总之，感谢一切。

最后，提前感谢购买本书的朋友们，你们的信任和赏识是我们继续前进的动力，对于本书有任何意见或建议，可以直接与我们联系，联系邮件：[sjms\\_2010@yahoo.cn](mailto:sjms_2010@yahoo.cn)，我们也很乐意与各位朋友交流设计模式或是其他相关的技术内容。

# 目 录

第 1 章 设计模式基础.....	1	2.3.6 思考简单工厂.....	27
1.1 设计模式是什么.....	2	2.3.7 相关模式.....	27
1.1.1 什么是模式.....	2	第 3 章 外观模式.....	29
1.1.2 设计模式的概念.....	2	3.1 场景问题.....	30
1.1.3 设计模式的理解.....	3	3.1.1 生活中的示例.....	30
1.1.4 设计模式的历史.....	4	3.1.2 代码生成的应用.....	31
1.2 设计模式有什么.....	4	3.1.3 不用模式的解决方案.....	31
1.2.1 设计模式的组成.....	4	3.1.4 有何问题.....	35
1.2.2 设计模式的分类.....	4	3.2 解决方案.....	35
1.3 设计模式的学习.....	5	3.2.1 使用外观模式来解决 问题.....	35
1.3.1 为什么要学习设计模式.....	5	3.2.2 外观模式的结构和说明.....	36
1.3.2 学习设计模式的层次.....	5	3.2.3 外观模式示例代码.....	36
1.3.3 如何学习设计模式.....	6	3.2.4 使用外观模式重写示例.....	39
1.4 本书的组织方式.....	7	3.3 模式讲解.....	40
1.4.1 本书所讲述的设计模式 的提纲.....	7	3.3.1 认识外观模式.....	40
1.4.2 每个模式的讲述结构.....	9	3.3.2 外观模式的实现.....	41
第 2 章 简单工厂.....	11	3.3.3 外观模式的优缺点.....	44
2.1 场景问题.....	12	3.3.4 思考外观模式.....	44
2.1.1 接口回顾.....	12	3.3.5 相关模式.....	45
2.1.2 面向接口编程.....	12	第 4 章 适配器模式 (Adapter).....	47
2.1.3 不用模式的解决方案.....	14	4.1 场景问题.....	48
2.1.4 有何问题.....	15	4.1.1 装配电脑的例子.....	48
2.2 解决方案.....	16	4.1.2 同时支持数据库和文件 的日志管理.....	49
2.2.1 使用简单工厂来解决问 题.....	16	4.1.3 有何问题.....	54
2.2.2 简单工厂的结构和说明.....	16	4.2 解决方案.....	55
2.2.3 简单工厂示例代码.....	17	4.2.1 使适配器模式来解决问 题.....	55
2.2.4 使用简单工厂重写示例.....	19	4.2.2 适配器模式的结构和说 明.....	55
2.3 模式讲解.....	20	4.2.3 适配器模式示例代码.....	56
2.3.1 典型疑问.....	20	4.2.4 使用适配器模式来实现 示例.....	58
2.3.2 认识简单工厂.....	21	4.3 模式讲解.....	61
2.3.3 简单工厂中方法的写法.....	22	4.3.1 认识适配器模式.....	61
2.3.4 可配置的简单工厂.....	24		
2.3.5 简单工厂的优缺点.....	26		

4.3.2	适配器模式的实现 .....	62	6.2.1	使用工厂方法模式来 解决问题 .....	103
4.3.3	双向适配器 .....	62	6.2.2	工厂方法模式的结构 和说明 .....	104
4.3.4	对象适配器和类适配器 .....	66	6.2.3	工厂方法模式示例代码 .....	104
4.3.5	适配器模式的优缺点 .....	69	6.2.4	使用工厂方法模式来 实现示例 .....	105
4.3.6	思考适配器模式 .....	70	6.3	模式讲解 .....	108
4.3.7	相关模式 .....	70	6.3.1	认识工厂方法模式 .....	108
<b>第 5 章</b>	<b>单例模式 (Singleton) .....</b>	<b>73</b>	6.3.2	工厂方法模式 与 IoC/DI .....	112
5.1	场景问题 .....	74	6.3.3	平行的类层次结构 .....	115
5.1.1	读取配置文件的内容 .....	74	6.3.4	参数化工厂方法 .....	117
5.1.2	不用模式的解决方案 .....	74	6.3.5	工厂方法模式的优缺点 .....	120
5.1.3	有何问题 .....	76	6.3.6	思考工厂方法模式 .....	121
5.2	解决方案 .....	76	6.3.7	相关模式 .....	123
5.2.1	使用单例模式来解决问 题 .....	76	<b>第 7 章</b>	<b>抽象工厂模式</b>	
5.2.2	单例模式的结构和说明 .....	77		<b>(Abstract Factory) .....</b>	<b>125</b>
5.2.3	单例模式示例代码 .....	77	7.1	场景问题 .....	126
5.2.4	使用单例模式重写示例 .....	80	7.1.1	选择组装电脑的配件 .....	126
5.3	模式讲解 .....	82	7.1.2	不用模式的解决方案 .....	126
5.3.1	认识单例模式 .....	82	7.1.3	有何问题 .....	132
5.3.2	懒汉式和饿汉式实现 .....	83	7.2	解决方案 .....	132
5.3.3	延迟加载的思想 .....	86	7.2.1	使用抽象工厂模式来 解决问题 .....	132
5.3.4	缓存的思想 .....	87	7.2.2	抽象工厂模式的结构 和说明 .....	133
5.3.5	Java 中缓存的基本实现 .....	88	7.2.3	抽象工厂模式示例 代码 .....	134
5.3.6	利用缓存来实现单例模 式 .....	89	7.2.4	使用抽象工厂模式重写 示例 .....	136
5.3.7	单例模式的优缺点 .....	90	7.3	模式讲解 .....	140
5.3.8	在 Java 中一种更好的单 例实现方式 .....	93	7.3.1	认识抽象工厂模式 .....	140
5.3.9	单例和枚举 .....	94	7.3.2	定义可扩展的工厂 .....	141
5.3.10	思考单例模式 .....	95	7.3.3	抽象工厂模式和 DAO .....	146
5.3.11	相关模式 .....	97	7.3.4	抽象工厂模式的 优缺点 .....	151
<b>第 6 章</b>	<b>工厂方法模式</b>		7.3.5	思考抽象工厂模式 .....	151
	<b>(Factory Method) .....</b>	<b>99</b>	7.3.6	相关模式 .....	152
6.1	场景问题 .....	100			
6.1.1	导出数据的应用框架 .....	100			
6.1.2	框架的基础知识 .....	100			
6.1.3	有何问题 .....	102			
6.2	解决方案 .....	103			

<b>第 8 章 生成器模式 (Builder)</b> .....	153		
8.1 场景问题 .....	154	9.3.4 原型管理器 .....	211
8.1.1 继续导出数据的应用 框架 .....	154	9.3.5 原型模式的优缺点 .....	214
8.1.2 不用模式的解决方案 ..	154	9.3.6 思考原型模式 .....	215
8.1.3 有何问题 .....	161	9.3.7 相关模式 .....	215
8.2 解决方案 .....	161	<b>第 10 章 中介者模式 (Mediator)</b> ...	217
8.2.1 使用生成器模式来解决 问题 .....	161	10.1 场景问题 .....	218
8.2.2 生成器模式的结构和 说明 .....	162	10.1.1 如果没有主板 .....	218
8.2.3 生成器模式示例代码 ..	162	10.1.2 有何问题 .....	218
8.2.4 使用生成器模式重写 示例 .....	164	10.1.3 使用电脑来看电影 ...	219
8.3 模式讲解 .....	170	10.2 解决方案 .....	219
8.3.1 认识生成器模式 .....	170	10.2.1 使用中介者模式来解决 问题 .....	219
8.3.2 生成器模式的实现 .....	171	10.2.2 中介者模式的结构和 说明 .....	220
8.3.3 使用生成器模式构建 复杂对象 .....	172	10.2.3 中介者模式示例代码 ..	220
8.3.4 生成器模式的优点 .....	182	10.2.4 使用中介者模式来实现 示例 .....	223
8.3.5 思考生成器模式 .....	182	10.3 模式讲解 .....	230
8.3.6 相关模式 .....	183	10.3.1 认识中介者模式 .....	230
<b>第 9 章 原型模式 (Prototype)</b> ..	185	10.3.2 广义中介者 .....	232
9.1 场景问题 .....	186	10.3.3 中介者模式的优缺点 ..	242
9.1.1 订单处理系统 .....	186	10.3.4 思考中介者模式 .....	243
9.1.2 不用模式的解决方案 ..	186	10.3.5 相关模式 .....	243
9.1.3 有何问题 .....	192	<b>第 11 章 代理模式 (Proxy)</b> .....	245
9.2 解决方案 .....	193	11.1 场景问题 .....	246
9.2.1 使用原型模式来解决 问题 .....	193	11.1.1 访问多条数据 .....	246
9.2.2 原型模式的结构和 说明 .....	194	11.1.2 不用模式的解决方案 ..	246
9.2.3 原型模式示例代码 .....	194	11.1.3 有何问题 .....	250
9.2.4 使用原型模式重写 示例 .....	196	11.2 解决方案 .....	250
9.3 模式讲解 .....	200	11.2.1 使用代理模式来解决 问题 .....	250
9.3.1 认识原型模式 .....	200	11.2.2 代理模式的结构和 说明 .....	251
9.3.2 Java 中的克隆方法 .....	202	11.2.3 代理模式示例代码 .....	252
9.3.3 浅度克隆和深度克隆 ..	204	11.2.4 使用代理模式重写 示例 .....	253
		11.3 模式讲解 .....	259
		11.3.1 认识代理模式 .....	259

11.3.2	保护代理.....	261	13.2.3	命令模式示例代码 ...	304
11.3.3	Java 中的代理.....	266	13.2.4	使用命令模式来实现 示例 .....	307
11.3.4	代理模式的特点.....	269	13.3	模式讲解.....	312
11.3.5	思考代理模式.....	269	13.3.1	认识命令模式 .....	312
11.3.6	相关模式.....	272	13.3.2	参数化配置 .....	313
<b>第 12 章</b>	<b>观察者模式 (Observer)</b> .....	<b>273</b>	13.3.3	可撤销的操作 .....	317
12.1	场景问题 .....	274	13.3.4	宏命令 .....	327
12.1.1	订阅报纸的过程.....	274	13.3.5	队列请求 .....	333
12.1.2	订阅报纸的问题.....	274	13.3.6	日志请求 .....	341
12.2	解决方案 .....	275	13.3.7	命令模式的优点 .....	346
12.2.1	使用观察者模式来解决 问题.....	275	13.3.8	思考命令模式 .....	347
12.2.2	观察者模式的结构和 说明.....	276	13.3.9	退化的命令模式 .....	347
12.2.3	观察者模式示例代码 .....	277	13.3.10	相关模式 .....	351
12.2.4	使用观察者模式实现 示例.....	279	<b>第 14 章</b>	<b>迭代器模式 (Iterator)</b> ..	<b>353</b>
12.3	模式讲解 .....	283	14.1	场景问题.....	354
12.3.1	认识观察者模式 .....	283	14.1.1	工资表数据的整合 ...	354
12.3.2	推模型和拉模型.....	285	14.1.2	有何问题 .....	354
12.3.3	Java 中的观察者模式 .....	289	14.2	解决方案.....	354
12.3.4	观察者模式的优缺点 .....	292	14.2.1	使用迭代器模式来解决 问题.....	354
12.3.5	思考观察者模式.....	293	14.2.2	迭代器模式的结构和 说明 .....	355
12.3.6	Swing 中的观察者 模式.....	293	14.2.3	迭代器模式示例代码 .....	355
12.3.7	简单变形示例——区别 对待观察者.....	294	14.2.4	使用迭代器模式来实现 示例 .....	359
12.3.8	相关模式.....	299	14.3	模式讲解.....	368
<b>第 13 章</b>	<b>命令模式 (Command)</b> .....	<b>301</b>	14.3.1	认识迭代器模式 .....	368
13.1	场景问题 .....	302	14.3.2	使用 Java 的迭代器 ..	370
13.1.1	如何开机.....	302	14.3.3	带迭代策略的迭代器 .....	373
13.1.2	与我何干 .....	302	14.3.4	双向迭代器 .....	376
13.1.3	有何问题.....	302	14.3.5	迭代器模式的优点 ...	379
13.2	解决方案 .....	303	14.3.6	思考迭代器模式 .....	380
13.2.1	使用命令模式来解决 问题.....	303	14.3.7	翻页迭代 .....	381
13.2.2	命令模式的结构和 说明.....	304	14.3.8	相关模式 .....	389
			<b>第 15 章</b>	<b>组合模式 (Composite)</b> .....	<b>391</b>
			15.1	场景问题.....	392
			15.1.1	商品类别树 .....	392

15.1.2	不用模式的解决方案	392	16.3.5	实现通用的增删改查	449
15.1.3	有何问题 .....	395	16.3.6	模板方法模式的 优缺点 .....	463
15.2	解决方案 .....	396	16.3.7	思考模板方法模式 .....	463
15.2.1	使用组合模式来解决 问题 .....	396	16.3.8	相关模式 .....	464
15.2.2	组合模式的结构和 说明 .....	396	<b>第 17 章 策略模式 (Strategy)</b>		465
15.2.3	组合模式示例代码 .....	397	17.1	场景问题 .....	466
15.2.4	使用组合模式重写 示例 .....	400	17.1.1	报价管理 .....	466
15.3	模式讲解 .....	405	17.1.2	不用模式的解决方案	466
15.3.1	认识组合模式 .....	405	17.1.3	有何问题 .....	467
15.3.2	安全性和透明性 .....	407	17.2	解决方案 .....	469
15.3.3	父组件引用 .....	409	17.2.1	使用策略模式来解决 问题 .....	469
15.3.4	环状引用 .....	414	17.2.2	策略模式的结构和 说明 .....	470
15.3.5	组合模式的优缺点 .....	418	17.2.3	策略模式示例代码 .....	470
15.3.6	思考组合模式 .....	419	17.2.4	使用策略模式重写 示例 .....	472
15.3.7	相关模式 .....	419	17.3	模式讲解 .....	475
<b>第 16 章 模板方法模式</b>			17.3.1	认识策略模式 .....	475
(Template Method)		421	17.3.2	Context 和 Strategy 的 关系 .....	477
16.1	场景问题 .....	422	17.3.3	容错恢复机制 .....	484
16.1.1	登录控制 .....	422	17.3.4	策略模式结合模板 方法模式 .....	487
16.1.2	不用模式的解决方案	422	17.3.5	策略模式的优缺点 .....	490
16.1.3	有何问题 .....	428	17.3.6	思考策略模式 .....	492
16.2	解决方案 .....	428	17.3.7	相关模式 .....	493
16.2.1	使用模板方法模式来 解决问题 .....	428	<b>第 18 章 状态模式 (State)</b>		495
16.2.2	模板方法模式的结构 和说明 .....	429	18.1	场景问题 .....	496
16.2.3	模板方法模式示例 代码 .....	429	18.1.1	实现在线投票 .....	496
16.2.4	使用模板方法模式重 写示例 .....	430	18.1.2	不用模式的解决方案	496
16.3	模式讲解 .....	434	18.1.3	有何问题 .....	498
16.3.1	认识模板方法模式 .....	434	18.2	解决方案 .....	498
16.3.2	模板的写法 .....	438	18.2.1	使用状态模式来解决 问题 .....	498
16.3.3	Java 回调与模板方法 模式 .....	441	18.2.2	状态模式的结构和 说明 .....	499
16.3.4	典型应用: 排序 .....	445			

18.2.3	状态模式示例代码	499			
18.2.4	使用状态模式重写 示例	501			
18.3	模式讲解	505			
18.3.1	认识状态模式	505			
18.3.2	状态的维护和转换 控制	509			
18.3.3	使用数据库来维护 状态	514			
18.3.4	模拟工作流	516			
18.3.5	状态模式的优缺点	527			
18.3.6	思考状态模式	527			
18.3.7	相关模式	528			
<b>第 19 章</b>	<b>备忘录模式 (Memento)</b>	<b>529</b>			
19.1	场景问题	530			
19.1.1	开发仿真系统	530			
19.1.2	不用模式的解决方案	530			
19.1.3	有何问题	533			
19.2	解决方案	533			
19.2.1	使用备忘录模式来解决 问题	533			
19.2.2	备忘录模式的结构和 说明	534			
19.2.3	备忘录模式示例代码	535			
19.2.4	使用备忘录模式重写 示例	537			
19.3	模式讲解	541			
19.3.1	认识备忘录模式	541			
19.3.2	结合原型模式	544			
19.3.3	离线存储	546			
19.3.4	再次实现可撤销操作	549			
19.3.5	备忘录模式的优缺点	558			
19.3.6	思考备忘录模式	558			
19.3.7	相关模式	559			
<b>第 20 章</b>	<b>享元模式 (Flyweight)</b>	<b>561</b>			
20.1	场景问题	562			
20.1.1	加入权限控制	562			
20.1.2	不使用模式的解决 方案	563			
20.1.3	有何问题	568			
20.2	解决方案	569			
20.2.1	使用享元模式来解决 问题	569			
20.2.2	享元模式的结构和 说明	570			
20.2.3	享元模式示例代码	570			
20.2.4	使用享元模式重写 示例	573			
20.3	模式讲解	578			
20.3.1	认识享元模式	578			
20.3.2	不需要共享的享元 实现	580			
20.3.3	对享元对象的管理	587			
20.3.4	享元模式的优缺点	596			
20.3.5	思考享元模式	597			
20.3.6	相关模式	597			
<b>第 21 章</b>	<b>解释器模式 (Interpreter)</b>	<b>599</b>			
21.1	场景问题	600			
21.1.1	读取配置文件	600			
21.1.2	不用模式的解决方案	600			
21.1.3	有何问题	602			
21.2	解决方案	604			
21.2.1	使用解释器模式来解 决问题	604			
21.2.2	解释器模式的结构和 说明	605			
21.2.3	解释器模式示例代码	605			
21.2.4	使用解释器模式重写 示例	607			
21.3	模式讲解	615			
21.3.1	认识解释器模式	615			
21.3.2	读取多个元素或属性 的值	617			
21.3.3	解析器	625			
21.3.4	解释器模式的优缺点	633			
21.3.5	思考解释器模式	633			

21.3.6 相关模式 .....	634	23.3 模式讲解 .....	679
<b>第 22 章 装饰模式 (Decorator) .....</b>	<b>635</b>	23.3.1 认识职责链模式 .....	679
22.1 场景问题 .....	636	23.3.2 处理多种请求 .....	680
22.1.1 复杂的奖金计算 .....	636	23.3.3 功能链 .....	691
22.1.2 简化后的奖金计算 体系 .....	636	23.3.4 职责链模式的优缺点 .....	697
22.1.3 不用模式的解决方案 .....	636	23.3.5 思考职责链模式 .....	697
22.1.4 有何问题 .....	639	23.3.6 相关模式 .....	698
22.2 解决方案 .....	640	<b>第 24 章 桥接模式 (Bridge) .....</b>	<b>701</b>
22.2.1 使用装饰模式来解决 问题 .....	640	24.1 场景问题 .....	702
22.2.2 装饰模式的结构和 说明 .....	641	24.1.1 发送提示消息 .....	702
22.2.3 装饰模式示例代码 .....	642	24.1.2 不用模式的解决方案 .....	702
22.2.4 使用装饰模式重写 示例 .....	644	24.1.3 有何问题 .....	705
22.3 模式讲解 .....	650	24.2 解决方案 .....	707
22.3.1 认识装饰模式 .....	650	24.2.1 使用桥接模式来解决 问题 .....	707
22.3.2 Java 中的装饰模式 应用 .....	653	24.2.2 桥接模式的结构和 说明 .....	708
22.3.3 装饰模式和 AOP .....	657	24.2.3 桥接模式示例代码 .....	709
22.3.4 装饰模式的优缺点 .....	664	24.2.4 使用桥接模式重写 示例 .....	710
22.3.5 思考装饰模式 .....	664	24.3 模式讲解 .....	715
22.3.6 相关模式 .....	665	24.3.1 认识桥接模式 .....	715
<b>第 23 章 职责链模式 (Chain of Responsibility) .....</b>	<b>667</b>	24.3.2 谁来桥接 .....	718
23.1 场景问题 .....	668	24.3.3 典型例子——JDBC .....	721
23.1.1 申请聚餐费用 .....	668	24.3.4 广义桥接——Java 中 无处不桥接 .....	723
23.1.2 不用模式的解决方案 .....	668	24.3.5 桥接模式的优点 .....	726
23.1.3 有何问题 .....	671	24.3.6 思考桥接模式 .....	727
23.2 解决方案 .....	671	24.3.7 相关模式 .....	728
23.2.1 使用职责链模式来解决 问题 .....	671	<b>第 25 章 访问者模式 (Visitor) .....</b>	<b>731</b>
23.2.2 职责链模式的结构和 说明 .....	672	25.1 场景问题 .....	732
23.2.3 职责链模式示例代码 .....	673	25.1.1 扩展客户管理的功能 .....	732
23.2.4 使用职责链模式重写 示例 .....	674	25.1.2 不用模式的解决方案 .....	734
		25.1.3 有何问题 .....	738
		25.2 解决方案 .....	739
		25.2.1 使用访问者模式来解决 问题 .....	739