



畅销书全新升级，第1版广受好评，被誉为设计模式领域最具趣味、最易理解且又讲解极为透彻的一本书，程序员公认的3本经典设计模式著作之一

深刻解读6大设计原则和28种设计模式的准确定义、应用方法和最佳实践，全方位比较各种同类模式之间的异同，详细讲解组合使用不同模式的方法

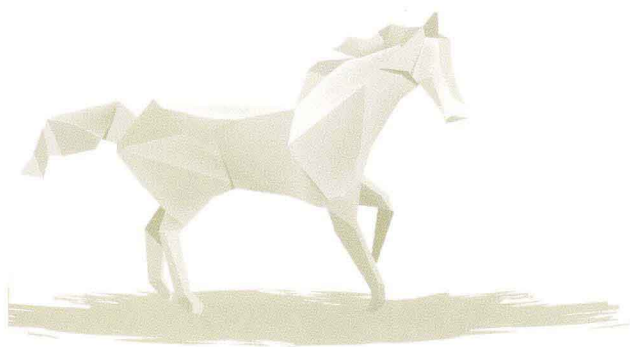


第2版

设计模式之禅

The Zen
of Design Patterns second Edition

秦小波 著



机械工业出版社
China Machine Press

第2版

设计模式之禅

The Zen
of Design Patterns second Edition

秦小波 著



机械工业出版社

图书在版编目(CIP)数据

设计模式之禅/秦小波著. —2版. —北京:机械工业出版社, 2013.10
(华章原创精品)

ISBN 978-7-111-43787-1

I. 设… II. 秦… III. 软件设计 IV. TP311.5

中国版本图书馆CIP数据核字(2013)第201756号

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问 北京市展达律师事务所

本书是设计模式领域公认的3本经典著作之一,“极具趣味,容易理解,但讲解又极为严谨和透彻”是本书的写作风格和方法的最大特点。第1版2010年出版,畅销至今,广受好评,是该领域的里程碑著作。深刻解读6大设计原则和28种设计模式的准确定义、应用方法和最佳实践,全方位比较各种同类模式之间的异同,详细讲解将不同的模式组合使用的方法。第2版在第1版的基础上有两方面的改进,一方面结合读者的意见和建议对原有内容中的瑕疵进行了修正和完善,另一方面增加了4种新的设计模式,希望这一版能为广大程序员们奉上一场更加完美的设计模式盛宴!

全书共38章,分为五部分:第一部分(第1~6章),以一种全新的视角对面向对象程序设计的6大原则进行了深刻解读,旨在让读者能更深刻且准确地理解这些原则,为后面的学习打下基础;第二部分(第7~29章)通过大量生动的案例讲解和分析了23种最常用的设计模式,并进行了扩展讲解,通俗易懂,趣味性极强而又紧扣模式的核心;第三部分(第30~33章)对同类型和相关联的模式进行了深入分析和比较,旨在阐明各种设计模式之间的差别以及它们的理想应用场景;第四部分(第34~36章)探讨了如何在实际开发中将各种设计模式混合起来使用,以发挥设计模式的最大效用;第五部分(第37~38章)是本书的扩展篇,首先从实现的角度对MVC框架的原理进行了深入分析,然后讲解了5种新的设计模式的原理、意图和最佳实践。本书最后附有一份精美的设计模式彩图,可以裁剪,便于参考。

机械工业出版社(北京市西城区百万庄大街22号 邮政编码 100037)

责任编辑:陈佳媛 迟振春 刘立卿 白 宇

蕻城市京瑞印刷有限公司印刷

2014年1月第2版第1次印刷

186mm×240mm • 36.25印张(含0.5印张彩插)

标准书号:ISBN 978-7-111-43787-1

定 价:89.00元

凡购本书,如有缺页、倒页、脱页,由本社发行部调换

客服热线:(010) 88378991 88361066

投稿热线:(010) 88379604

购书热线:(010) 68326294 88379649 68995259

读者信箱:hzsj@hzbook.com

前言



为什么写这本书

2009年5月份，我在JavaEye上发了一个帖子，其中提到自己已经工作9年了，总觉得这9年不应该就这么荒废了，应该给自己这9年的工作写一个总结，总结的初稿就是这本书。

在谈为什么写这本书之前，先抖抖自己前9年的职业生涯吧。大学时我是学习机械的，当时计算机刚刚热起来，自己也喜欢玩一些新奇的东西，记得最清楚的是用VB写了一个自由落体的小程序，模拟小球从桌面掉到地板上，然后计算反弹趋势，很有成就感。于是2000年毕业时，我削尖了脑袋进入了IT行业，成为了一名真正的IT男，干着起得比鸡早、睡得比狗晚的程序员工作，IT男的辛酸有谁知晓！

坦白地说，我的性格比较沉闷，属于典型的程序员型闷骚，比较适合做技术研究。在这9年里，项目管理做过，系统分析做过，小兵当过，团队领导人也当过，但至今还是一个做技术的。要总结这9年技术生涯，总得写点什么吧，最好是还能对其他人有点儿用的。那写什么好呢？Spring、Struts等工具框架类的书太多太多，很难再写出花样来，经过一番思考，最后选择了一个每一位技术人员都需要掌握的、但普及程度还不是非常高的、又稍微有点难度的主题——设计模式（Design Pattern，DP）。

中国人有不破不立的思维，远的如秦始皇焚书坑儒、项羽火烧阿房宫，近的如破“四旧”。正是由于有了这样的思想，于是乎能改的就改，不能改的就推翻重写，没有一个持续开发蓝图。为什么要破才能立呢？为什么不能持续地发展？你说这是谁的错呢？是你架构师的错，你不能持续地拥抱变化，这是一个系统最失败的地方。那怎样才能实现拥抱变化的理想呢？设计模式！

设计模式是什么？它是一套理论，由软件界的先辈们（The Gang of Four：包括Erich Gamma、Richard Helm、Ralph Johnson、John Vlissides）总结出的一套可以反复使用的经验，它可以提高代码的可重用性，增强系统的可维护性，以及解决一系列的复杂问题。做软件的人都知道需求是最难把握的，我们可以分析现有的需求，预测可能发生的变更，但是我们不能控制需求的变更。问题来了，既然需求的变更是不可控的，那如何拥抱变化呢？幸运的是，设计模式给了我们指导，专家们首先提出了6大设计原则，但这6大设计原则仅仅是一系列“口号”，

真正付诸实施还需要有详尽的指导方法，于是23种设计模式出现了。

设计模式已经诞生20年了，其间出版了很多关于它的经典著作，相信大家都能如数家珍。尽管有这么多书，工作5年了还不知道什么是策略模式、状态模式、责任链模式的程序员大有人在。不信？你找个机会去“虚心”地请教一下你的同事，看看他对设计模式有多少了解。不要告诉我翻书才明白！设计模式不是工具，它是软件开发的哲学，它能指导你如何去设计一个优秀的架构、编写一段健壮的代码、解决一个复杂的需求。

因为它是软件行业的经验总结，因此它具有更广泛的适应性，不管你使用什么编程语言，不管你遇到什么业务类型，设计模式都可以自由地“侵入”。

因为它不是工具，所以它没有一个可以具体测量的标尺，完全以你自己的理解为准，你认为自己多了解它，你就有可能产生多少的优秀代码和设计。

因为它是指导思想，你可以在此基础上自由发挥，甚至是自己设计出一套设计模式。

世界上最难的事有两件：一是让人心甘情愿地把钱掏出来给你，二是把自己的思想灌输到别人的脑子里。设计模式就属于第二种，它不是一种具体的技术，不像Struts、Spring、Hibernate等框架。一个工具用久了可以熟能生巧，就像砌墙的工人一样，长年累月地砌墙，他也知道如何把墙砌整齐，如何多快好省地干活，这是一个人的本能。我们把Struts用得很溜，把Spring用得很顺手，这非常好，但这只是一个合格的程序员应该具备的基本能力！于是我们被冠以代码工人（Code Worker）——软件行业的体力劳动者。

如果你通晓了这23种设计模式就不同了，你可以站在一个更高的层次去赏析程序代码、软件设计、架构，完成从代码工人到架构师的蜕变。注意，我说的是“通晓”，别告诉我你把23种设计模式的含义、适应性、优缺点都搞清楚了就是通晓。错了！没有工作经验的积累是不可能真正理解设计模式的，这就像大家小时候一直不明白为什么爸爸妈妈要工作而不能每天陪自己玩一样。

据说有的大学已经开了设计模式这门课，如果仅仅是几堂课，让学生对设计模式有一个初步的了解，我觉得并无不妥，但如果是专门的一门课程，我建议取消它！因为对一个尚无项目开发经验的学生来说，理解设计模式不是一般困难，而是非常非常困难！之前没有任何的实战经验，之后也没有可以立即付诸实践的场景，这样能理解设计模式吗？

在编写本书之前，23种设计模式我都用过，而且还算比较熟练，但是当真正要写到书中时，感觉心里没谱儿了。这个定义是这样的吗？是需要用抽象类还是应该用接口？为什么在这里不能抽取抽象呢？为什么在实际项目中这个模式要如此蜕化？这类小问题有时候很纠结，需要花费大量的精力和时间去分析和确认。所以，在写作的过程中我有过很多忧虑，担心书中会有太多瑕疵，这种忧虑现在仍然存在。遇到挫折的时候也气馁过，但是我坚信一句话：“开弓没有回头箭，回头即是空”，既然已经开始，就一定要圆满完成。

第2版与第1版的区别

本书是第2版，在写作中吸取了读者对上一版的许多意见和建议，修订了一些代码的变量、类、方法名称，以更加符合自然语言；删除了部分有争议的内容（如单例模式的垃圾回收问题）；修改了一些常用的名词，确保与编程人员的习惯相匹配。希望通过这些改进，给读者提供一个更完美的设计模式盛宴，弥补上一版中的诸多不足。

第2版第38章中新增了4种新的设计模式：对象池模式、雇工模式、黑板模式、空指针模式。这些模式是我们在实际工作中经常遇到，或者在开源代码时常看到的，但是我们却没有升级到“模式”这一理性高度。特别是像空指针模式，我们在编码中经常会遇到空值判断问题，但我们没有去想一想是否可以有更好的方式解决。第2版对空指针模式进行了讲解，虽然简单，但相信对你提升编码质量有很大的帮助。

本书的特色

简单、通俗、易懂，但又不肤浅，这是本书的最大特色。自己看过的技术书还算比较多，很痛恨那种大块头的巨著，搁家里当枕头都觉得太硬。如果要是再晦涩难懂点，那根本没法看，看起来实在是太累。设计模式原本就是理论性的知识，讲解的难度比较大，但我相信这本书能够把你对设计模式的恐惧一扫而光。不信？挑几页先看看！

我的理念是：**像看小说一样阅读本书。**我尽量用浅显通俗的语言讲解，尽量让你有继续看下去的欲望，尽量努力让你有兴趣进入设计模式的世界，兴趣是第一老师嘛！虽然我尽量让这本书浅显、通俗、易懂，但并不代表我的讲解就很肤浅。每个设计模式讲解完毕之后，我都附加了两个非常精华的部分：设计模式扩展和最佳实践，这是俺压箱底的技能了，为了博君一看，没招了，抖出来吧！**尤为值得一提的是，本书还有设计模式PK和混编设计模式两部分内容教你如何自如地去运用这些设计模式，这是当前所有设计模式类的图书都不具备的，连最权威的那本书也不例外。**

我很讨厌技术文章中夹杂着的那些晦涩难懂的文字，特别是一堆又一堆的名词堆砌，让人看着就反胃。但是为了学习技术，为了生存，还是必须看下去。国内的技术文档，基本上都是板着一副冷面孔讲技术，为什么要把技术弄得这么生硬呢？技术也有它幽默、柔情的一面，只是被我们的“孔夫子们”掩盖了，能用萝卜、白菜这种寻常人都熟悉的知识来讲解原子弹理论的人，那是牛人，我佩服这样的人。记住，用一堆名词把你忽悠晕的人很可能什么都不懂！

本书想告诉你的是，技术也可以很有趣，也可以让你不用皱着眉头思考，等待你的只是静静地看，慢慢地思考，本书的内容会润物细无声地融入你的思维中。

本书面向的读者

热爱技术并且讨厌枯燥乏味技术文章的读者都可以看本书；

你是**程序员**，没问题，本书能够让你写出更加高效、优雅的代码；

你是**架构师**，那更好，设计模式可让你设计出健壮、稳定、高效的系统，并且自动地预防未来业务变化可能对系统带来的影响；

你是**项目经理**，也OK，设计模式可以让你的工期大大缩短，让你的项目团队成员快速地了解你的意图，最终的成果就是优质的项目：高可靠性、高稳定性、高效率和低维护成本。

如何阅读本书

首先声明，本书中所有的例子都是用Java语言来实现的，但是你可以随手翻翻看，基本上能保证每三条语句一个注释，可以说是在用咱们的母语讲解设计模式。即使你不懂Java语言，也没有关系，只要知道在Java中双斜杠（//）代表注释就足够了，况且Java如此强大和盛行，多了解一点没有坏处。类图看不懂？没关系，不影响你理解设计模式，多看看就懂了！

如果你还没有编程经验，我建议你把它当做小说来看，懂行的看门道，不懂行的看热闹，这里的热闹足够多，够你看一壶的了。你现在能看懂多少是多少，不懂没有关系，你要知道，经验不是像青春痘一样，说长就长出来了，它是需要时间积累的，需要你用心去感受，然后才能明白为什么要如此设计。

如果你已经对编程有感觉了（至少两年开发经验），我相信你都能看懂，但能“懂”到什么程度，就很难说了，看你的水平了。但是，我可以保证，这里的设计模式都是你能看懂的，没有你看不懂的！我建议你通读这本书，然后挑门你最得意的编程语言，动手写吧！给自己制定一个计划，每天编写一段代码，不需要太多，200行足够，时不时地把设计模式融入你的代码中。甭管是什么代码，比如你想编写一个识别美女图片的程序，好呀，抓紧时间去写吧，写好了就不用到处看美女了，程序一跑就把网上的美女图片都抓过来了，牛呀（记住，程序写好了要分享给我）。看吧，坚持下去，一年以后你再跟你的同侪比较一下，那差距肯定不是一般的大。

如果你是资深工程师、架构师、技术顾问等高等级的技术人员，那我告诉你，你找对这本书了。系统架构没有思路？没有问题，看看扩展部分，它会开阔你的思路。系统的维护成本居高不下？看看本书，设计模式也许能帮你省点银子。开发资源无法保证？设计模式能让你用有限的资源（软硬件资源和人力资源）设计出一个优秀的系统。项目质量参差不齐，缺陷一大堆？多用设计模式，它会给你意想不到的效果。给人讲课没有素材？没问题，本书中的素材足以让你赢得阵阵掌声！

编程是一门艺术活，我有一个同事，能把类图画成一个小乌龟的形状，天才呀！作为一位

技术人员，最基本的品质就是诚实，“知之为知之，不知为不知，是知也”，自己不懂没有关系，去学，学无止境，但是千万不要贪多，这抓一点，那挖一点，好像什么都懂，其实什么都不懂。中国一直推崇复合型人才，我不是很赞成，因为这对年轻人来说是一个误导。先精一项技术，然后再发散学习，先点后面才是正道。

记得《武林外传》中有这样一段对话：

刑捕头：手中无刀，心中有刀。

老白：错了，最高境界是手中无刀，心中也无刀。

体验一下吧，我们的设计模式就是一把刀，极致的境界就是心中无设计模式，代码亦无设计模式——设计模式随处可见，俯拾皆是，已经融入软件设计的灵魂中，这才是高手中的高手，简称高高手。

哦，最重要的忘记说了，请把附录中的“23种设计模式附图”撕下来，贴在你的办公桌上，时不时地看看，也让老板看看，咱是多么地用心！

关于书名

乍一看，书名和内容貌似不相符呀，其实不然！

在我们的常规思维中，“禅”应该是很高深的东西，只可意会，不可言传。没错，禅宗也是如此说。禅是得道者的“悟”，是不能用言语来表达的，但是得道者为了能让更多的人“悟”，就必须用最容易让人理解的文字把自己的体会表达出来。本书的“禅”是作者对设计模式的“悟”，本书的“形”就是你现在看到的这些极其简单、通俗、易懂的文字。

至此，大家应该不会再对书名有疑虑了吧，嘿嘿。

致谢

本书第1版的写作耗时7个月，第2版的更新又花了4个月，可以说是榨干了海绵里所有的水——基本上能用的时间都用上了。在公交车上打腹稿，干过！在马桶上查资料，干过！在睡梦中思考案例，也有过！就差没有走火入魔了！

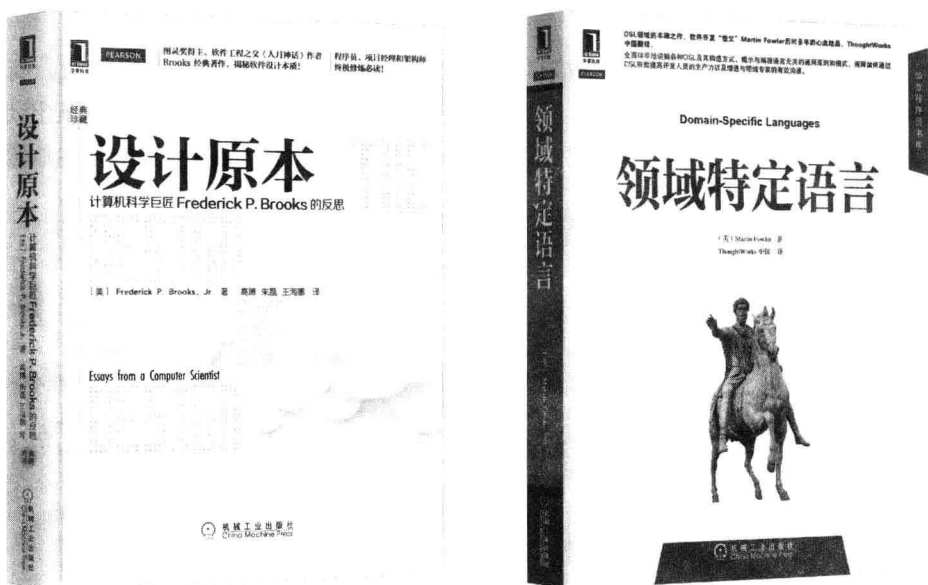
首先，感谢杨福川编辑，没有他的慧眼，这本书不可能出版。其次，感谢妻子和儿子，每天下班回到家，一按门铃，儿子就在里面叫：“我来开门，我来开门。”儿子三岁，太调皮了，他不睡觉我基本上是不能开写的，我一旦开始写东西，他就跑过来问：“爸爸，你在干什么呀”，紧接着下一句就是“爸爸，你陪我玩”，基本都是拿我当玩具，别的小朋友都是把父亲当马骑，他却不，他把我当摩托车骑，还要加油门，发动……小家伙脚太重了，再骑摩托，非被他踩死不可！

还要感谢我的朋友王骢，周末只要小家伙在家，我只有找地方写书的份儿，王骢非常爽快地把钥匙给我，让我有一个安静的地方写书。一个人沉浸在自己喜欢的世界里也是一件非常幸福的事。

当然，还要感谢JavaEye上所有顶帖的网友，没有你们的支持我就没有写作的动力，就像希腊神话中的巨人安泰失去了大地的力量一样，是你们的回帖让我觉得不孤单，让我知道我不是一个人在战斗！

最后，再次对本书中可能出现的错误表示歉意，真诚地接受大家轰炸！如果你在阅读本书时发现错误或有问题想讨论，请发邮件给我。

推荐阅读



设计原本（精装本）

如果说《人月神话》是近40年来所有软件开发工程师和项目经理们必读的一本书，那么本书将会是未来数十年内从事软件行业的程序员、项目经理和架构师必读的一本书。它是《人月神话》作者、著名计算机科学家、软件工程教父、美国两院院士、图灵奖和IEEE计算机先驱奖得主Brooks在计算机软硬件架构与设计、建筑和组织机构的架构与设计等领域毕生经验的结晶，是计算机图书领域的又一史诗级著作。

领域特定语言

本书是DSL领域的丰碑之作，由世界级软件开发大师和软件开发“教父”Martin Fowler历时多年写作而成。全面详尽地讲解了各种DSL及其构造方式，揭示了与编程语言无关的通用原则和模式，阐释了如何通过DSL有效提高开发人员的生产力以及增进与领域专家的有效沟通，能为开发人员选择和使用DSL提供有效的决策依据和指导方法。

目 录



前 言

第一部分 大旗不挥，谁敢冲锋——6大设计原则全新解读

第1章 单一职责原则2

- 1.1 我是“牛”类，我可以担任多职吗.....2
- 1.2 绝杀技，打破你的传统思维3
- 1.3 我单纯，所以我快乐6
- 1.4 最佳实践7

第2章 里氏替换原则8

- 2.1 爱恨纠葛的父子关系8
- 2.2 纠纷不断，规则压制9
- 2.3 最佳实践18

第3章 依赖倒置原则19

- 3.1 依赖倒置原则的定义19
- 3.2 言而无信，你太需要契约20
- 3.3 依赖的三种写法25
- 3.4 最佳实践26

第4章 接口隔离原则28

- 4.1 接口隔离原则的定义28
- 4.2 美女何其多，观点各不同29
- 4.3 保证接口的纯洁性33

4.4 最佳实践35

第5章 迪米特法则36

- 5.1 迪米特法则的定义36
- 5.2 我的知识你知道得越少越好36
- 5.3 最佳实践43

第6章 开闭原则44

- 6.1 开闭原则的定义44
- 6.2 开闭原则的庐山真面目44
- 6.3 为什么要采用开闭原则49
- 6.4 如何使用开闭原则51
- 6.5 最佳实践55

第二部分 真刀实枪——23种设计模式完美演绎

第7章 单例模式58

- 7.1 我是皇帝我独苗58
- 7.2 单例模式的定义59
- 7.3 单例模式的应用60
 - 7.3.1 单例模式的优点60
 - 7.3.2 单例模式的缺点60
 - 7.3.3 单例模式的使用场景61
 - 7.3.4 单例模式的注意事项61
- 7.4 单例模式的扩展62

7.5 最佳实践	64	11.3 建造者模式的应用	111
第8章 工厂方法模式	65	11.3.1 建造者模式的优点	111
8.1 女娲造人的故事	65	11.3.2 建造者模式的使用场景	111
8.2 工厂方法模式的定义	69	11.3.3 建造者模式的注意事项	111
8.3 工厂方法模式的应用	70	11.4 建造者模式的扩展	111
8.3.1 工厂方法模式的优点	70	11.5 最佳实践	112
8.3.2 工厂方法模式的使用场景	71	第12章 代理模式	113
8.4 工厂方法模式的扩展	71	12.1 我是游戏至尊	113
8.5 最佳实践	77	12.2 代理模式的定义	116
第9章 抽象工厂模式	78	12.3 代理模式的应用	118
9.1 女娲的失误	78	12.3.1 代理模式的优点	118
9.2 抽象工厂模式的定义	83	12.3.2 代理模式的使用场景	119
9.3 抽象工厂模式的应用	86	12.4 代理模式的扩展	119
9.3.1 抽象工厂模式的优点	86	12.4.1 普通代理	119
9.3.2 抽象工厂模式的缺点	86	12.4.2 强制代理	121
9.3.3 抽象工厂模式的使用场景	86	12.4.3 代理是有个性的	126
9.3.4 抽象工厂模式的注意事项	86	12.4.4 动态代理	128
9.4 最佳实践	87	12.5 最佳实践	134
第10章 模板方法模式	88	第13章 原型模式	135
10.1 辉煌工程——制造悍马	88	13.1 个性化电子账单	135
10.2 模板方法模式的定义	93	13.2 原型模式的定义	141
10.3 模板方法模式的应用	94	13.3 原型模式的应用	142
10.3.1 模板方法模式的优点	94	13.3.1 原型模式的优点	142
10.3.2 模板方法模式的缺点	95	13.3.2 原型模式的使用场景	142
10.3.3 模板方法模式的使用场景	95	13.4 原型模式的注意事项	143
10.4 模板方法模式的扩展	95	13.4.1 构造函数不会被执行	143
10.5 最佳实践	99	13.4.2 浅拷贝和深拷贝	144
第11章 建造者模式	100	13.4.3 clone与final两个冤家	146
11.1 变化是永恒的	100	13.5 最佳实践	146
11.2 建造者模式的定义	109		

第14章 中介者模式	147	第17章 装饰模式	192
14.1 进销存管理是这个样子的吗	147	17.1 罪恶的成绩单	192
14.2 中介者模式的定义	156	17.2 装饰模式的定义	198
14.3 中介者模式的应用	159	17.3 装饰模式应用	201
14.3.1 中介者模式的优点	159	17.3.1 装饰模式的优点	201
14.3.2 中介者模式的缺点	159	17.3.2 装饰模式的缺点	201
14.3.3 中介者模式的使用场景	159	17.3.3 装饰模式的使用场景	201
14.4 中介者模式的实际应用	160	17.4 最佳实践	201
14.5 最佳实践	161	第18章 策略模式	203
第15章 命令模式	162	18.1 刘备江东娶妻，赵云他容易吗	203
15.1 项目经理也难当	162	18.2 策略模式的定义	206
15.2 命令模式的定义	170	18.3 策略模式的应用	208
15.3 命令模式的应用	173	18.3.1 策略模式的优点	208
15.3.1 命令模式的优点	173	18.3.2 策略模式的缺点	208
15.3.2 命令模式的缺点	173	18.3.3 策略模式的使用场景	209
15.3.3 命令模式的使用场景	173	18.3.4 策略模式的注意事项	209
15.4 命令模式的扩展	173	18.4 策略模式的扩展	209
15.4.1 未讲完的故事	173	18.5 最佳实践	214
15.4.2 反悔问题	174	第19章 适配器模式	215
15.5 最佳实践	175	19.1 业务发展——上帝才能控制	215
第16章 责任链模式	178	19.2 适配器模式的定义	221
16.1 古代妇女的枷锁—— “三从四德”	178	19.3 适配器模式的应用	223
16.2 责任链模式的定义	186	19.3.1 适配器模式的优点	223
16.3 责任链模式的应用	189	19.3.2 适配器模式的使用场景	224
16.3.1 责任链模式的优点	189	19.3.3 适配器模式的注意事项	224
16.3.2 责任链模式的缺点	190	19.4 适配器模式的扩展	224
16.3.3 责任链模式的注意事项	190	19.5 最佳实践	229
16.4 最佳实践	190	第20章 迭代器模式	230
		20.1 整理项目信息——苦差事	230
		20.2 迭代器模式的定义	236

20.3 迭代器模式的应用	239	23.3 门面模式的应用	284
20.4 最佳实践	239	23.3.1 门面模式的优点	284
第21章 组合模式	240	23.3.2 门面模式的缺点	285
21.1 公司的人事架构是这样的吗	240	23.3.3 门面模式的使用场景	285
21.2 组合模式的定义	253	23.4 门面模式的注意事项	285
21.3 组合模式的应用	255	23.4.1 一个子系统可以有多个 门面	285
21.3.1 组合模式的优点	255	23.4.2 门面不参与子系统内的 业务逻辑	286
21.3.2 组合模式的缺点	256	23.5 最佳实践	288
21.3.3 组合模式的使用场景	256	第24章 备忘录模式	289
21.3.4 组合模式的注意事项	256	24.1 如此追女孩子，你还不乐	289
21.4 组合模式的扩展	256	24.2 备忘录模式的定义	294
21.4.1 真实的组合模式	256	24.3 备忘录模式的应用	297
21.4.2 透明的组合模式	257	24.3.1 备忘录模式的使用场景	297
21.4.3 组合模式的遍历	259	24.3.2 备忘录模式的注意事项	297
21.5 最佳实践	260	24.4 备忘录模式的扩展	297
第22章 观察者模式	262	24.4.1 clone方式的备忘录	297
22.1 韩非子身边的卧底是谁派来的	262	24.4.2 多状态的备忘录模式	300
22.2 观察者模式的定义	271	24.4.3 多备份的备忘录	304
22.3 观察者模式的应用	273	24.4.4 封装得更好一点	305
22.3.1 观察者模式的优点	273	24.5 最佳实践	307
22.3.2 观察者模式的缺点	274	第25章 访问者模式	308
22.3.3 观察者模式的使用场景	274	25.1 员工的隐私何在	308
22.3.4 观察者模式的注意事项	274	25.2 访问者模式的定义	316
22.4 观察者模式的扩展	275	25.3 访问者模式的应用	320
22.4.1 Java世界中的观察者模式	275	25.3.1 访问者模式的优点	320
22.4.2 项目中真实的观察者模式	276	25.3.2 访问者模式的缺点	320
22.4.3 订阅发布模型	277	25.3.3 访问者模式的使用场景	320
22.5 最佳实践	277	25.4 访问者模式的扩展	321
第23章 门面模式	278	25.4.1 统计功能	321
23.1 我要投递信件	278	25.4.2 多个访问者	323
23.2 门面模式的定义	283		

25.4.3 双分派	326	第29章 桥梁模式	371
25.5 最佳实践	328	29.1 我有一个梦想	371
第26章 状态模式	329	29.2 桥梁模式的定义	379
26.1 城市的纵向发展功臣——电梯	329	29.3 桥梁模式的应用	381
26.2 状态模式的定义	341	29.3.1 桥梁模式的优点	381
26.3 状态模式的应用	343	29.3.2 桥梁模式的使用场景	382
26.3.1 状态模式的优点	343	29.3.3 桥梁模式的注意事项	382
26.3.2 状态模式的缺点	344	29.4 最佳实践	382
26.3.3 状态模式的使用场景	344		
26.3.4 状态模式的注意事项	344	第三部分 谁的地盘谁做主—— 设计模式PK	
26.4 最佳实践	344		
第27章 解释器模式	346	第30章 创建类模式大PK	384
27.1 四则运算你会吗	346	30.1 工厂方法模式VS建造者模式	384
27.2 解释器模式的定义	352	30.1.1 按工厂方法建造超人	384
27.3 解释器模式的应用	354	30.1.2 按建造者模式建造超人	386
27.3.1 解释器模式的优点	354	30.1.3 最佳实践	389
27.3.2 解释器模式的缺点	354	30.2 抽象工厂模式VS建造者模式	390
27.3.3 解释器模式使用的场景	355	30.2.1 按抽象工厂模式生产车辆	390
27.3.4 解释器模式的注意事项	355	30.2.2 按建造者模式生产车辆	394
27.4 最佳实践	355	30.2.3 最佳实践	399
第28章 享元模式	356	第31章 结构类模式大PK	400
28.1 内存溢出，司空见惯	356	31.1 代理模式VS装饰模式	400
28.2 享元模式的定义	361	31.1.1 代理模式	400
28.3 享元模式的应用	364	31.1.2 装饰模式	402
28.3.1 享元模式的优点和缺点	364	31.1.3 最佳实践	403
28.3.2 享元模式的使用场景	364	31.2 装饰模式VS适配器模式	404
28.4 享元模式的扩展	365	31.2.1 用装饰模式描述丑小鸭	404
28.4.1 线程安全的问题	365	31.2.2 用适配器模式实现丑小鸭	407
28.4.2 性能平衡	366	31.2.3 最佳实践	410
28.5 最佳实践	369		

第32章 行为类模式大PK411

32.1 命令模式VS策略模式411

32.1.1 策略模式实现压缩算法411

32.1.2 命令模式实现压缩算法414

32.1.3 小结419

32.2 策略模式VS状态模式420

32.2.1 策略模式实现人生420

32.2.2 状态模式实现人生423

32.2.3 小结425

32.3 观察者模式VS责任链模式426

32.3.1 责任链模式实现DNS 解析过程427

32.3.2 触发链模式实现DNS 解析过程432

32.3.3 小结437

第33章 跨战区PK438

33.1 策略模式VS桥梁模式438

33.1.1 策略模式实现邮件发送439

33.1.2 桥梁模式实现邮件发送442

33.1.3 最佳实践445

33.2 门面模式VS中介者模式446

33.2.1 中介者模式实现工资计算446

33.2.2 门面模式实现工资计算451

33.2.3 最佳实践454

33.3 包装模式群PK455

33.3.1 代理模式455

33.3.2 装饰模式457

33.3.3 适配器模式459

33.3.4 桥梁模式461

33.3.5 最佳实践464

第四部分 完美世界—— 设计模式混编

第34章 命令模式+责任链模式466

34.1 搬移UNIX的命令466

34.2 混编小结481

第35章 工厂方法模式+策略模式483

35.1 迷你版的交易系统483

35.2 混编小结493

第36章 观察者模式+中介者模式495

36.1 事件触发器的开发495

36.2 混编小结508

第五部分 扩展篇

第37章 MVC框架510

37.1 MVC框架的实现510

37.1.1 MVC的系统架构512

37.1.2 模型管理器518

37.1.3 值栈522

37.1.4 视图管理器522

37.1.5 工具类526

37.2 最佳实践528

第38章 新模式530

38.1 规格模式530

38.1.1 规格模式的实现530

38.1.2 最佳实践543

38.2 对象池模式	546	38.4 黑板模式	552
38.2.1 正确的池化	546	38.4.1 黑板模式的意图	552
38.2.2 对象池模式的意图	547	38.4.2 黑板模式的实现方法	553
38.2.3 最佳实践	549	38.5 空对象模式	554
38.3 雇工模式	549	38.5.1 空对象模式的例子	554
38.3.1 雇工合作	549	38.5.2 最佳实践	555
38.3.2 雇工模式的意图	551		
38.3.3 最佳实践	552		

附录 23种设计模式彩图