# <<重构与模式>>

## 图书基本信息

书名：<<重构与模式>>

13位ISBN编号：9787115297259

10位ISBN编号：7115297258

出版时间：2013-1

出版时间：人民邮电出版社

作者：Joshua Kerievsky

页数：295

字数：444000

# <<重构与模式>>

## 内容概要

《重构与模式(修订版)》开创性地深入揭示了重构与模式这两种软件开发关键技术之间的联系，说明了通过重构实现模式改善既有的设计，往往优于在新的设计早期使用模式。
《重构与模式(修订版)》不仅展示了一种应用模式和重构的创新方法，而且有助于读者透过实战深入理解重构和模式。
书中讲述了27种重构方式。

《重构与模式(修订版)》适于面向对象软件开发人员阅读，也可作为高等学校计算机专业、软件工程专业师生的参考读物。

## 作者简介

Joshua Kerievsky
最具人气的年轻一代软件开发专家之一，极限编程先驱、敏捷项目管理的思想领袖、敏捷eLearning的改革者。
软件开发公司Industrial
Logic的创始人。
他曾为许多专业杂志撰稿，并多次在世界级技术会议上担任讲师，并撰写了大量的论文。
除本书外，他还参与撰写了Extreme
Programming Explored和Extreme Programming Perspectives两本书。

# <<重构与模式>>

## 书籍目录

## 章节摘录



本书中讲述的几乎所有重构都是复合重构。
我们从一段待修改的代码着手，然后渐进地应用各种低层次重构，直至完成所需的修改。
在应用各个低层次重构之间，需要运行测试确保修改后的代码仍能如愿运行。
因此，测试也是复合重构不可分割的一部分。
如果不运行测试，你很难充满自信地应用低层次重构。

测试在重构中还扮演着一个完全不同的角色：它可以用来重新编写、代替老代码。
测试驱动的重构（test—driven refactoring）包括应用测试驱动开发得到替换代码，然后将老代码替换为新代码（同时保留并重新运行老代码的测试）。

与测试驱动的重构相比，复合重构的使用率要高得多，因为大量重构工作只是改变原有代码的位置。
当这样无法改善设计时，采用测试驱动的重构能够帮助我们安全而且有效地得到更佳的设计。

替换算法[F]重构是最适合使用测试驱动重构方式来实现重构的绝佳例子。
它实际上是彻底地将原有算法替换为另一个更简单、更清晰的算法。
你应该怎样得到新算法呢？
通过将老算法转换为新算法是不行的，因为新算法的逻辑与之完全不同。
可以先编写好新算法，用它替换老算法，然后看测试能否通过。
如果测试无法通过，你很可能要花上很长时间进行调试。
编写算法的更好方式是使用测试驱动开发。
这种方式能够产生简单的代码，而且还能产生测试，从而使你或者其他人能够在此后充满自信地应用各种低层次重构或者复合重构。

用Builder封装Composite（6.5节）重构是测试驱动的重构的另一个例子。
这种重构的目的是通过简化构建过程使客户代码能够更容易地构建Composite。
设计中使用Builder提供构建Composite的简化方式。
如果设计与原有设计差距很大，就可能无法使用多个低层次重构或者复合重构得到新的设计。
同样，使用测试驱动开发能够更加有效地重新实现、替换老代码。

用Composite替换隐含树（7.5节）重构既是一个复合重构又是一个测试驱动重构。
选择如何实现这个重构，取决于所遇到的代码的性质。
一般说来，如果该代码很难实现提炼类[F]重构，那么测试驱动重构方式可能更容易。
用Composite替换隐含树重构中有一个使用测试驱动重构的例子。

将装饰功能搬移Decorator（7.3节）重构不是测试驱动的重构；但是，这个重构的示例却说明了如何用测试驱动的重构将框架外的行为搬移到框架内。
这个示例所涉及的是搬移代码，因此似乎用复合重构实现应该更方便。
但事实上，修改要涉及更新许多类，最终还是用测试驱动开发进行设计转换更容易。

在实际重构中，可能大多数时间都在使用低层次重构和复合重构。
只需要记住，通过测试驱动的重构完成的“重新实现和替换”技术，也是重构的一种有用方式即可。
在设计一种新的算法或者机制时这种方式最有效，而且这种方式比应用低层次重构或者复合重构更容易。

## 媒体关注与评论

“重构必须付诸实践，才能体现出其真正价值，而非仅仅作为一种抽象的智力练习。
模式则记录了具有公认良好属性的程序结构。
本书将两者完美地结合起来。
如果想真正实践重构，我推荐你阅读本书并活学活用。
”——Kent Beck，软件开发方法学的泰斗，极限编程创始人，模式先驱“在《设计模式》一书中，我们曾经提到，设计模式是重构的目标。
本书终于证实我们所言不虚。
除此之外，本书还能够加深读者对设计模式和重构两方面的领悟。
”——Erich Gamma，IBM公司Eclipse Java开发工具负责人，《设计模式》四作者之一，模式先驱“现在，软件模式和敏捷开发之间的联系终于被人道破。
”——Ward Cunningham，极限编程创始人，模式先驱，Wiki发明者“本书展示了一种应用模式的创新方法，将自上而下地使用设计模式与自下而上地揭示迭代式开发和持续重构结合起来。
任何职业软件开发人员都应该使用这种方法，去寻找使用模式改进代码的新的可能。
”——Bobby Woolf，IBM公司WebSphere软件服务部门IT咨询专家，Enterprise Integration Patterns和The Design Patterns Smalltalk Companion作者之一“Joshua Kerievsky通过一系列独树一帜的设计级重构，将重构提升到全新的层次。
本书向开发人员展示了如何对设计进行改进，从而简化日常工作。
本书是重构实践的珍贵参考书。
”——Sven Gorts，重构与敏捷开发布道者，比利时refactoring.be网站创始人“本书是对《设计模式》一书的重构，可能意义还不仅限于此。
在此之前，设计模式这一主题一直是作为静态和僵化的过程来阐述的，本书则将其看做是动态和灵活的，使模式的学习变成了一种试验、出错然后改正的人性化过程，从中读者能够理解到，优秀的设计并非一蹴而就——它们都经历了艰难和反思。
Kerievsky还重构了阐述方式本身，使其更加清晰，更容易接受。
实际上，他解决了我在写作Thinking in Patterns一书中遇到的许多组织问题。
本书透彻地介绍并结合了测试、重构和设计模式诸多方面，字里行间洋溢着叙述的轻松、良好的技术感觉和难得的真知灼见。
”——Bruce Eckel，Mindview公司总裁，《Java编程思想》和《C++编程思想》的作者“我第一次见到Joshua，就对他在理解、应用和教授设计模式上表现出来的热情留下了深刻印象。
伟大的教师对自己教授的内容和如何与人分享都有这样的热情。
我想Joshua不愧是一位伟大的教师，一位伟大的开发者，我们都从他的深刻洞察中获益良多。
”——Craig Larman，Valtech首席科学家，《UML和模式应用》和《敏捷迭代开发》作者

## 编辑推荐

讲述重构与设计模式两大热门技术，将两者有机结合，极具实战价值，业界专家学习和教授重构与模式亲身经历的结晶，《设计模式》作者Erich Gamma、Ralph Johnson和《重构》作者Martin Fowler联合推荐。

## 名人推荐

重构必须付诸实践，才能体现出其真正价值，而非仅仅作为一种抽象的智力练习。
模式则记录了具有公认的良好属性的程序结构。
本书将两者完美地结合起来。
如果想真正实践重构，我推荐你阅读本书并活学活用。
——Kent Beck，软件开发方法学的泰斗，极限编程创始人，模式先驱 在《设计模式》一书中，我们曾经提到，设计模式是重构的目标。
本书终于证实我们所言不虚。
除此之外，本书还能够加深读者对设计模式和重构两方面的领悟。
——Erich Gamma，《设计模式》作者之一，模式先驱 现在，软件模式和敏捷开发之间的联系终于被人道破。
——Ward Cunningham，极限编程创始人，模式先驱，Wiki发明者 Joshua Kerievsky通过一系列独树一帜的设计级重构，将重构提升到全新的层次。
本书向开发人员展示了如何对设计进行改进，从而简化日常工作。
本书是重构实践的珍贵参考书。
——Sven Gorts，重构与敏捷开发布道者，比利时refactoring.be网站创始人 我第一次见到Joshua就对他在理解、应用和教授设计模式上表现出来的热情留下了深刻印象。
伟大的教师对自己教授的内容和如何与人分享都有这样的热情。
我想Joshua不愧是一位伟大的教师，一位伟大的开发者，我们都从他的深刻洞察中获益良多。
——Craig Larman，Valtech首席科学家，《UML和模式应用》和《敏捷迭代开发》作者 掌握一门手艺不仅仅要获得正确的工具，还需要学会高效地使用工具。
本书阐释了如何将工业级的设计工具与艺术家的技巧熔于一炉。
——Russ Rufer，硅谷模式讨论组创始人 本书是对《设计模式》一书的重构，可能意义还不仅限于此。
在此之前，设计模式这一主题一直是作为静态和僵化的过程来阐述的，本书则将其看做是动态和灵活的，使模式的学习变成了一种试验、出错然后改正的人性化过程，从中读者能够理解到，优秀的设计并非一蹴而就——它们都经历了艰难和反恩。
Kerievsky还重构了阐述方式本身。
使其更加清晰，更容易接受。
实际上，他解决了我在写作Thinking in Patterns一书中遇到的许多组织问题。
本书透彻地介绍并结合了测试、重构和设计模式诸多方面，字里行间洋溢着叙述的轻松、良好的技术感觉和难得的真知灼见。
——Bruce Eckel，Mindview公司总裁，《Java编程思想》和《C++编程思想》作者

## 版权说明