<<重构>>

图书基本信息

书名：<<重构>>

13位ISBN编号：9787115239143

10位ISBN编号：7115239142

出版时间：20101028

出版时间：人民邮电出版社

作者：Martin Fowler

页数：431

<<重构>>

前言

"重构"这个概念来自Smalltalk圈子，没多久就进入了其他语言阵营之中。
由于重构是框架开发中不可缺少的一部分，所以当框架开发人员讨论自己的工作时，这个术语就诞生了。
当他们精练自己的类继承体系时，当他们叫喊自己可以拿掉多少多少行代码时，重构的概念慢慢浮出水面。
框架设计者知道，这东西不可能一开始就完全正确，它将随着设计者的经验成长而进化；他们也知道，代码被阅读和被修改的次数远远多于它被编写的次数。
保持代码易读、易修改的关键，就是重构——对框架而言如此，对一般软件也如此。
好极了，还有什么问题吗？
问题很显然：重构具有风险。
它必须修改运作中的程序，这可能引入一些不易察觉的错误。
如果重构方式不恰当，可能毁掉你数天甚至数星期的成果。
如果重构时不做好准备，不遵守规则，风险就更大。
你挖掘自己的代码，很快发现了一些值得修改的地方，于是你挖得更深。
挖得愈深，找到的重构机会就越多，于是你的修改也愈多……最后你给自己挖了个大坑，却爬不出去了。
为了避免自掘坟墓，重构必须系统化进行。
我在《设计模式》书中和另外三位作者曾经提过：设计模式为重构提供了目标。
然而"确定目标"只是问题的一部分而已，改造程序以达到目标，是另一个难题。

<<重构>>

内容概要

《重构：改善既有代码的设计（英文版）》清晰揭示了重构的过程，解释了重构的原理和最佳实践方式，并给出了何时以及何地应该开始挖掘代码以求改善。
书中给出了70多个可行的重构，每个重构都介绍了一种经过验证的代码变换手法的动机和技术。
《重构：改善既有代码的设计（英文版）》提出的重构准则将帮助你一次一小步地修改你的代码，从而减少了开发过程中的风险。

作者简介

作者：（美国）福勒（Martin Fowler）Martin Fowler世界软件开发大师，在面向对象分析设计、UML、模式、xP和重构等领域都有卓越贡献。
现为著名软件开发咨询公司Thoughtworks的首席科学家。
他的多部著作《分析模式》、《UML精粹》和《企业应用架构模式》等都已经成为脍炙人口的经典。
其他参编者Kent Beck软件开发方法学的泰斗，极限编程的创始人。
他是Three Rivers lnslitule公司总裁，也是Agitar S0ftware的成员。
John Brant和Don Roberts The Refactory公司的创始人。
Refactory Browser的开发者。
多年来一直从事研究重构的实践与理论。
William Opdyke 目前在朗讯贝尔实验室工作，他写的关于面向对象框架的博士论文是重构方面的第一篇著名文章。

书籍目录

章节摘录

插图：Any technical author has the problem of deciding when to publish. The earlieryou publish, the quicker people can take advantage of the ideas. However, people are always learning. If you publish half-baked ideas too early, the ideas canbe incomplete and even lead to problems for those who try to use them. The basic technique of refactoring, taking small steps and testing often, hasbeen well tested over many years, especially in the Smalltalk community. So I'mconfident that the basic idea of refactoring is very stable. The refactorings in this book are my notes about the refactorings I use. Ihave used them all. However, there is a difference between using a refactoringand boiling it down into the mechanical steps I give herein. In particular, youoccasionally see problems that crop up only in very specific circumstances. Icannot say that I have had a lot of people work from these steps to spot many of these kinds of problems. As you use the refactorings, be aware of what youare doing. Remember that like working with a recipe, you have to adapt the refactorings to your circumstances. If you run into an interesting problem, dropme an e-mail, and I'll try to pass on these circumstances for others.

编辑推荐

《重构:改善既有代码的设计(英文版)》：软件开发的不朽经典、生动阐述重构原理和具体做法、普通程序员进阶到编程高手必须修炼的秘笈。

版权说明