

重构：改善既有代码的设计

作者：[美]马丁·福勒（Martin Fowler）

[目录](#)

[封面](#)

[重构列表](#)

[扉页](#)

[版权](#)

[版权声明](#)

[重构的重新认识（再版序）](#)

[重构的生活方式（译序）](#)

[序](#)

[前言](#)

[第1章 重构，第一个案例](#)

[1.1 起点](#)

[1.2 重构的第一步](#)

[1.3 分解并重组statement\(\)](#)

[1.4 运用多态取代与价格相关的条件逻辑](#)

[1.5 结语](#)

[第2章 重构原则](#)

[2.1 何谓重构](#)

[2.2 为何重构](#)

[2.3 何时重构](#)

[2.4 怎么对经理说](#)

[2.5 重构的难题](#)

[2.6 重构与设计](#)

[2.7 重构与性能](#)

[2.8 重构起源何处](#)

[第3章 代码的坏味道](#)

[3.1 Duplicated Code（重复代码）](#)

[3.2 Long Method（过长函数）](#)

[3.3 Large Class（过大的类）](#)

[3.4 Long Parameter List（过长参数列）](#)

[3.5 Divergent Change（发散式变化）](#)

[3.6 Shotgun Surgery（霰弹式修改）](#)

[3.7 Feature Envy（依恋情结）](#)

[3.8 Data Clumps（数据泥团）](#)

[3.9 Primitive Obsession（基本类型偏执）](#)

[3.10 Switch Statements（switch惊悚现身）](#)

[3.11 Parallel Inheritance Hierarchies（平行继承体系）](#)

[3.12 Lazy Class（冗赘类）](#)

[3.13 Speculative Generality（夸夸其谈未来性）](#)

[3.14 Temporary Field（令人迷惑的暂时字段）](#)

[3.15 Message Chains（过度耦合的消息链）](#)

[3.16 Middle Man（中间人）](#)

[3.17 Inappropriate Intimacy（狎昵关系）](#)

[3.18 Alternative Classes with Different Interfaces（异曲同工的类）](#)

[3.19 Incomplete Library Class（不完美的库类）](#)

[3.20 Data Class（纯稚的数据类）](#)

[3.21 Refused Bequest（被拒绝的遗赠）](#)

[3.22 Comments（过多的注释）](#)

[第4章 构筑测试体系](#)

[4.1 自测试代码的价值](#)

[4.2 JUnit测试框架](#)

[4.3 添加更多测试](#)

[第5章 重构列表](#)

[5.1 重构的记录格式](#)

[5.2 寻找引用点](#)

[5.3 这些重构手法有多成熟](#)

[第6章 重新组织函数](#)

[6.1 Extract Method \(提炼函数\)](#)

[6.2 Inline Method \(内联函数\)](#)

[6.3 Inline Temp \(内联临时变量\)](#)

[6.4 Replace Temp with Query \(以查询取代临时变量\)](#)

[6.5 Introduce Explaining Variable \(引入解释性变量\)](#)

[6.6 Split Temporary Variable \(分解临时变量\)](#)

[6.7 Remove Assignments to Parameters \(移除对参数的赋值\)](#)

[6.8 Replace Method with Method Object \(以函数对象取代函数\)](#)

[6.9 Substitute Algorithm \(替换算法\)](#)

[第7章 在对象之间搬移特性](#)

[7.1 Move Method \(搬移函数\)](#)

[7.2 Move Field \(搬移字段\)](#)

[7.3 Extract Class \(提炼类\)](#)

[7.4 Inline Class \(将类内联化\)](#)

[7.5 Hide Delegate \(隐藏“委托关系”\)](#)

[7.6 Remove Middle Man \(移除中间人\)](#)

[7.7 Introduce Foreign Method \(引入外加函数\)](#)

[7.8 Introduce Local Extension \(引入本地扩展\)](#)

[第8章 重新组织数据](#)

[8.1 Self Encapsulate Field \(自封装字段\)](#)

[8.2 Replace Data Value with Object \(以对象取代数据值\)](#)

[8.3 Change Value to Reference \(将值对象改为引用对象\)](#)

[8.4 Change Reference to Value \(将引用对象改为值对象\)](#)

[8.5 Replace Array with Object \(以对象取代数组\)](#)

[8.6 Duplicate Observed Data \(复制“被监视数据”\)](#)

[8.7 Change Unidirectional Association to Bidirectional \(将单向关联改为双向关联\)](#)

[8.8 Change Bidirectional Association to Unidirectional \(将双向关联改为单向关联\)](#)

[8.9 Replace Magic Number with Symbolic Constant \(以字面常量取代魔法数\)](#)

[8.10 Encapsulate Field \(封装字段\)](#)

[8.11 Encapsulate Collection \(封装集合\)](#)

[8.12 Replace Record with Data Class \(以数据类取代记录\)](#)

[8.13 Replace Type Code with Class \(以类取代类型码\)](#)

[8.14 Replace Type Code with Subclasses \(以子类取代类型码\)](#)

[8.15 Replace Type Code with State/Strategy \(以State/Strategy取代类型码\)](#)

[8.16 Replace Subclass with Fields \(以字段取代子类\)](#)

[第9章 简化条件表达式](#)

[9.1 Decompose Conditional \(分解条件表达式\)](#)

[9.2 Consolidate Conditional Expression \(合并条件表达式\)](#)

[9.3 Consolidate Duplicate Conditional Fragments \(合并重复的条件片段\)](#)

[9.4 Remove Control Flag \(移除控制标记\)](#)

[9.5 Replace Nested Conditional with Guard Clauses \(以卫语句取代嵌套条件表达式\)](#)

[9.6 Replace Conditional with Polymorphism \(以多态取代条件表达式\)](#)

[9.7 Introduce Null Object \(引入Null对象\)](#)

[9.8 Introduce Assertion \(引入断言\)](#)

[第10章 简化函数调用](#)

[10.1 Rename Method \(函数改名\)](#)

[10.2 Add Parameter \(添加参数\)](#)

[10.3 Remove Parameter \(移除参数\)](#)

[10.4 Separate Query from Modifier \(将查询函数和修改函数分离\)](#)

[10.5 Parameterize Method \(令函数携带参数\)](#)

[10.6 Replace Parameter with Explicit Methods \(以明确函数取代参数\)](#)

[10.7 Preserve Whole Object \(保持对象完整\)](#)

[10.8 Replace Parameter with Methods \(以函数取代参数\)](#)

[10.9 Introduce Parameter Object \(引入参数对象\)](#)

[10.10 Remove Setting Method \(移除设值函数\)](#)

[10.11 Hide Method \(隐藏函数\)](#)

[10.12 Replace Constructor with Factory Method \(以工厂函数取代构造函数\)](#)

[10.13 Encapsulate Downcast \(封装向下转型\)](#)

[10.14 Replace Error Code with Exception \(以异常取代错误码\)](#)
[10.15 Replace Exception with Test \(以测试取代异常\)](#)
[第11章 处理概括关系](#)
[11.1 Pull Up Field \(字段上移\)](#)
[11.2 Pull Up Method \(函数上移\)](#)
[11.3 Pull Up Constructor Body \(构造函数本体上移\)](#)
[11.4 Push Down Method \(函数下移\)](#)
[11.5 Push Down Field \(字段下移\)](#)
[11.6 Extract Subclass \(提炼子类\)](#)
[11.7 Extract Superclass \(提炼超类\)](#)
[11.8 Extract Interface \(提炼接口\)](#)
[11.9 Collapse Hierarchy \(折叠继承体系\)](#)
[11.10 Form Template Method \(塑造模板函数\)](#)
[11.11 Replace Inheritance with Delegation \(以委托取代继承\)](#)
[11.12 Replace Delegation with Inheritance \(以继承取代委托\)](#)
[第12章 大型重构](#)
[12.1 Tease Apart Inheritance \(梳理并分解继承体系\)](#)
[12.2 Convert Procedural Design to Objects \(将过程化设计转化为对象设计\)](#)
[12.3 Separate Domain from Presentation \(将领域和表述/显示分离\)](#)
[12.4 Extract Hierarchy \(提炼继承体系\)](#)
[第13章 重构, 复用与现实](#)
[13.1 现实的检验](#)
[13.2 为什么开发者不愿意重构他们的程序](#)
[13.3 再论现实的检验](#)
[13.4 重构的资源 and 参考资料](#)
[13.5 从重构联想到软件复用和技术传播](#)
[13.6 小结](#)
[13.7 参考文献](#)
[第14章 重构工具](#)
[14.1 使用工具进行重构](#)
[14.2 重构工具的技术标准](#)
[14.3 重构工具的实用标准](#)
[14.4 小结](#)
[第15章 总结](#)
[参考书目](#)
[要点列表](#)
[索引](#)
[代码的坏味道](#)

重构列表

Add Parameter (添加参数)	275
Change Bidirectional Association to Unidirectional (将双向关联改为单向关联)	200
Change Reference to Value (将引用对象改为值对象)	183
Change Unidirectional Association to Bidirectional (将单向关联改为双向关联)	197
Change Value to Reference (将值对象改为引用对象)	179
Collapse Hierarchy (折叠继承体系)	344
Consolidate Conditional Expression (合并条件表达式)	240
Consolidate Duplicate Conditional Fragments (合并重复的条件片段)	243
Convert Procedural Design to Objects (将过程化设计转化为对象设计)	368
Decompose Conditional (分解条件表达式)	238
Duplicate Observed Data (复制“被监视数据”)	189
Encapsulate Collection (封装集合)	208
Encapsulate Downcast (封装向下转型)	308
Encapsulate Field (封装字段)	206
Extract Class (提炼类)	149
Extract Hierarchy (提炼继承体系)	375
Extract Interface (提炼接口)	341
Extract Method (提炼函数)	110
Extract Subclass (提炼子类)	330
Extract Superclass (提炼超类)	336
Form Template Method (塑造模板函数)	345
Hide Delegate (隐藏“委托关系”)	157
Hide Method (隐藏函数)	303
Inline Class (将类内联化)	154
Inline Method (内联函数)	117
Inline Temp (内联临时变量)	119
Introduce Assertion (引入断言)	267
Introduce Explaining Variable (引入解释性变量)	124

Introduce Foreign Method (引入外加函数)	162
Introduce Local Extension (引入本地扩展)	164
Introduce Null Object (引入Null对象)	260
Introduce Parameter Object (引入参数对象)	295
Move Field (搬移字段)	146
Move Method (搬移函数)	142
Parameterize Method (令函数携带参数)	283
Preserve Whole Object (保持对象完整)	288
Pull Up Constructor Body (构造函数本体上移)	325
Pull Up Field (字段上移)	320
Pull Up Method (函数上移)	322
Push Down Field (字段下移)	329
Push Down Method (函数下移)	328
Remove Assignments to Parameters (移除对参数的赋值)	131
Remove Control Flag (移除控制标记)	245
Remove Middle Man (移除中间人)	160
Remove Parameter (移除参数)	277
Remove Setting Method (移除设值函数)	300
Rename Method (函数改名)	273
Replace Array with Object (以对象取代数组)	186
Replace Conditional with Polymorphism (以多态取代条件表达式)	255
Replace Constructor with Factory Method (以工厂函数取代构造函数)	304
Replace Data Value with Object (以对象取代数据值)	175
Replace Delegation with Inheritance (以继承取代委托)	355
Replace Error Code with Exception (以异常取代错误码)	310
Replace Exception with Test (以测试取代异常)	315
Replace Inheritance with Delegation (以委托取代继承)	352
Replace Magic Number with Symbolic Constant (以字面常量取代魔法数)	204
Replace Method with Method Object (以函数对象取代函数)	135
Replace Nested Conditional with Guard Clauses (以卫语句取代嵌套条件表达式)	250

Replace Parameter with Explicit Methods (以明确函数取代参数)	285
Replace Parameter with Methods (以函数取代参数)	292
Replace Record with Data Class (以数据类取代记录)	217
Replace Subclass with Fields (以字段取代子类)	232
Replace Temp with Query (以查询取代临时变量)	120
Replace Type Code with Class (以类取代类型码)	218
Replace Type Code with State/Strategy (以State/Strategy取代类型码)	227
Replace Type Code with Subclasses (以子类取代类型码)	223
Self Encapsulate Field (自封装字段)	171
Separate Domain from Presentation (将领域和表述/显示分离)	370
Separate Query from Modifier (将查询函数和修改函数分离)	279
Split Temporary Variable (分解临时变量)	128
Substitute Algorithm (替换算法)	139
Tease Apart Inheritance (梳理并分解继承体系)	362

PEARSON

Refactoring Improving the Design of Existing Code

重构改善既有代码的设计

[美]Martin Fowler 著

熊节 译

人民邮电出版社

北京

本书仅供个人学习之用，请勿用于商业用途。如对本书有兴趣，请购买正版书籍。任何对本书籍的修改、加工、传播自负法律后果。

本书由“行行”整理，如果你不知道读什么书或者想获得更多免费电子书请加小编微信或QQ：2338856113 小编也和结交一些喜欢读书的朋友 或者关注小编个人微信公众号名称：幸福的味道 为了方便书友朋友找书和看书，小编自己做了一个电子书下载网站，网站的名称为：周读网
址：www.ireadweek.com

重构：改善既有代码的设计/（美）福勒（Fowler,M.）著；熊节译.--2版.--北京：人民邮电出版社，2015.8

书名原文：Refactoring:Improving the Design of Existing Code

ISBN 978-7-115-36909-3

I.①重... II.①福...②熊... III.①机器代码程序—程序设计 IV.①TP311.11

中国版本图书馆CIP数据核字（2015）第137778号

内容提要

本书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。本书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。

本书适合软件开发人员、项目管理人员等阅读，也可作为高等院校计算机及相关专业师生的参考读物。

◆著 [美] Martin Fowler

译 熊节

责任编辑 杨海玲

责任印制 张佳莹 焦志炜

◆人民邮电出版社出版发行 北京市丰台区成寿寺路11号

邮编 100164 电子邮件 315@ptpress.com.cn

网址 <http://www.ptpress.com.cn>

三河市中晟雅豪印务有限公司印刷

◆开本：800×1000 1/16

印张：28.25

字数：490千字 2015年8月第2版

印数：1-6000册 2015年8月河北第1次印刷

著作权合同登记号 图字：01-2009-5707号

定价：69.00元

读者服务热线：(010)81055410 印装质量热线：(010)81055316

反盗版热线：(010)81055315

广告经营许可证：京崇工商广字第0021号

重构的重新认识（再版序）

光阴荏苒，从当年译完这本《重构》，到如今重新整理译稿，不知不觉已经过去6年了。6年来，在各种大型系统中进行重构和指导别人重构，一直是我的一项工作。对于这本早已烂熟于心的书，也有了一些新的认识。

不得不遗憾地说，尽管“重构”已经成了常用词汇，但重构技术并没有像我当初乐观认为的那样“变得像空气与水一样普通”。一方面，一种甚嚣尘上的观点认为只要掌握重构的思想就足够了，没必要记住那些详细琐碎的重构手法；另一方面，倒是有很多人高擎“重构”大旗，刀劈斧砍进行着令人触目惊心的大胆修改——有些干脆就是在重做整个系统。

这些人常常忘了一个最基本的定义：重构是在不改变软件可观察行为的前提下改善其内部结构。当你面对一个最需要重构的遗留系统时，其规模之大、历史之久、代码质量之差，常会使得添加单元测试或者理解其逻辑都成为不可能的任务。此时你唯一能依靠的就是那些已经被证明是行为保持的重构手法：用绝对安全的手法从“焦油坑”中整理出可测试的接口，给它添加测试，以此作为继续重构的立足点。

六年来，在各种语言、各种行业、各种软件形态，包括规模达到上百万行代码的项目中进行重构的经验让我明白，“不改变软件行为”只是重构的最基本要求。要想真正让重构技术发挥威力，就必须做到“不需了解软件行为”——听起来很荒谬，但事实如此。如果一段代码能让你容易了解其行为，说明它还不是那么迫切需要被重构。那些最需要重构的代码，你只能看到其中的“坏味道”，接着选择对应的重构手法来消除这些“坏味道”，然后才有可能理解它的行为。而这整个过程之所以可行，全赖你在脑子里记录着一份“坏味道”与重构手法的对应表。

而且，尽管Java和.NET的自动化重构工具已经相当成熟，但另一些重要的面向对象语言（C++、Ruby、Python……）还远未享受到这样的便利。在重构这些语言编写的程序时，我们仍然必须遵循这些看似琐碎的做法指导（加上语言特有的细节调整），按部就班地进行——如果你还想以安全的方式重构的话。

所以，仅仅掌握思想是没用的。如果把重构比作一门功夫的话，它的威力全都来自日积月累的勤学苦练。记住所有的“坏味道”，记住它们对应的重构手法，记住常见的重构步骤，然后你才可能有信心面对各种复杂情况——学会所有的招式，才可能“无招胜有招”。我知道这听起来很难，但我也知道这并不像你想象的那么难。你所需要的只是耐心、毅力和不断重读这本书。

熊节

2009年10月21日

欢迎访问：电子书学习和下载网站 (<https://www.shgis.cn>)

文档名称：《重构：改善既有代码的设计》[美]马丁·福勒 (Martin Fowler) 著.epub

请登录 <https://shgis.cn/post/329.html> 下载完整文档。

手机端请扫码查看：

