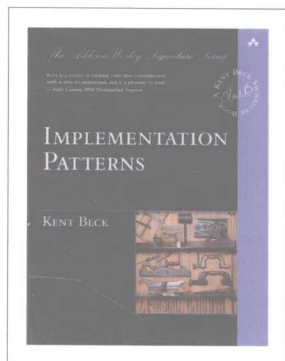


PEARSON



软件开发方法学 精选系列



Implementation Patterns

[美] Kent Beck 著
李剑 熊节 郭晓刚 译

实现模式

(修订版)

人民邮电出版社
POSTS & TELECOM PRESS

九江学院图书馆



1554295

1538786



软件开发方法学精选系列

Implementation Patterns

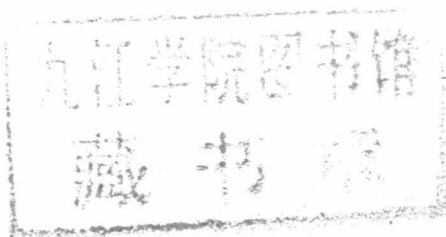
[美] Kent Beck 著

李剑 熊节 郭晓刚 译

实现模式



(修订版)



TP311.1/
23272.

人民邮电出版社

北京

图书在版编目 (CIP) 数据

实现模式 / (美) 贝克 (Beck, K.) 著 ; 李剑, 熊节, 郭晓刚译. — 修订本. — 北京 : 人民邮电出版社, 2012. 12

(软件开发方法学精选系列)

书名原文: Implementation Patterns

ISBN 978-7-115-29460-9

I. ①实… II. ①贝… ②李… ③熊… ④郭… III. ①程序设计 IV. ①TP311.1

中国版本图书馆CIP数据核字(2012)第220865号

版权声明

Authorized translation from the English language edition, entitled Implementation Patterns, 9780321413093 by Kent Beck, published by Pearson Education, Inc, publishing as Addison-Wesley, Copyright © 2008 Pearson Education, Inc.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc. CHINESE SIMPLIFIED language edition published by PEARSON EDUCATION ASIA LTD., and POSTS & TELECOMMUNICATIONS PRESS Copyright © 2009.

本书封面贴有 Pearson Education (培生教育出版集团) 激光防伪标签。无标签者不得销售。

软件开发方法学精选系列

实现模式 (修订版)

-
- ◆ 著 [美] Kent Beck
 - 译 李 剑 熊 节 郭晓刚
 - 责任编辑 杨海玲
 - ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
 - 邮编 100061 电子邮件 315@ptpress.com.cn
 - 网址 <http://www.ptpress.com.cn>
 - 北京隆昌伟业印刷有限公司印刷
 - ◆ 开本: 700×1000 1/16
 - 印张: 12
 - 字数: 183 千字 2012 年 12 月第 2 版
 - 印数: 1-3 000 册 2012 年 12 月北京第 1 次印刷

著作权合同登记号 图字: 01-2008-3844 号

ISBN 978-7-115-29460-9

定价: 35.00 元

读者服务热线: (010)67132692 印装质量热线: (010)67129223

反盗版热线: (010)67171154

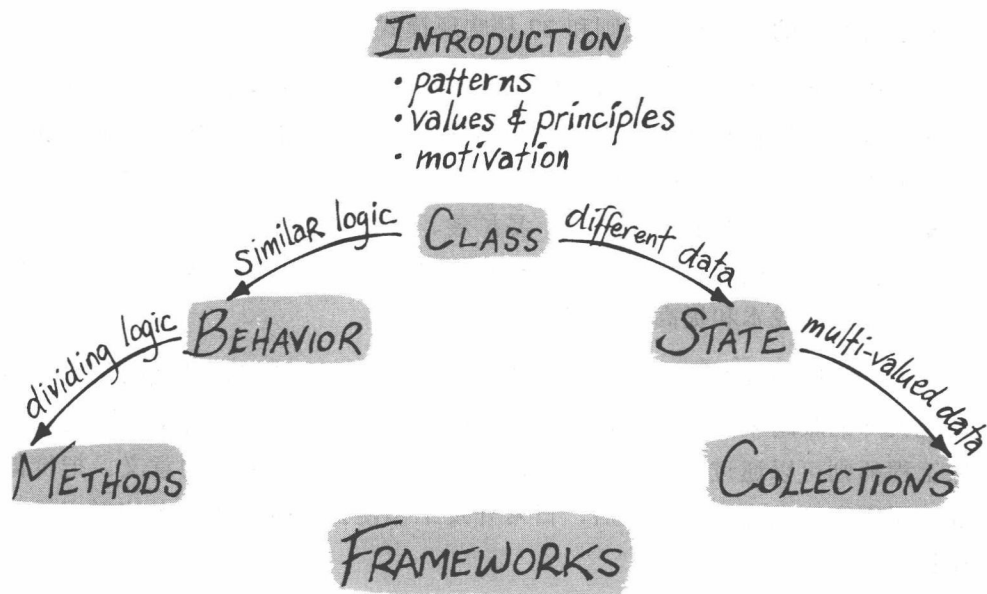
广告经营许可证: 京崇工商广字第 0021 号

内容提要

在本书中，作者将自己多年形成的编程习惯以及阅读既有代码的体验凝练成了编程中的价值观、原则和 77 种实现模式。

沟通、简单和灵活的价值观应当被所有开发人员所铭记。局部影响、最小化重复、将逻辑与数据捆绑等原则同样是通用性的指导思想，比价值观更贴近编程场景，在价值观和模式之间搭建了桥梁。在 77 种实现模式中，每一种模式都覆盖了编写简洁、清晰、易扩展、易维护的代码这一原则的某个方面。它们为日常的编程提供了丰富翔实的参考依据，并告诉大家这些代码如何为降低沟通成本和提高有效产出提供保障。

本书适用于各个阶段的开发者群体。刚刚涉足软件开发领域的新人能够透过大师的眼睛来看待编程，了解编程的价值观与原则；具有丰富经验的资深工程师则可以通过这些模式进行反思，探究成功实践背后的意义。把价值观、原则和开发实践结合之后，日常开发工作便会以崭新而迷人的形式呈现在我们面前。



实现模式概览

译者序

这是一本关于如何写好代码的书。

如果你不认为写好代码是一件重要、困难并且有趣的事，请立即放下这本书。

什么是好的代码？可以工作的、性能良好的、不出 bug 的代码，就是好的代码吗？

所谓好的代码，除了其他所有要求以外，还应该清晰准确地传达写作者的想法。

Martin Fowler 在《重构：改善既有代码的设计》里说，“任何一个傻瓜都能写出机器能懂的代码。好的程序员应该写出人能懂的代码。”

如果你不同意这句话，请立即放下这本书。因为这是一本关于如何用代码与他人（而非机器）沟通的书。

任何读到这一行的程序员都应该读完这本书。

Steve McConnell 在《代码大全》里说，“不要过早优化，但也不要过早劣化”。这本书将告诉你如何在几乎不引入任何额外成本的前提下避免一些常见的低级错误——它们是常见的，因为几乎每个人都犯过并且还在犯着这些错误。

如果你确实没有时间，至少应该读完第 6 章“状态”。因为在各种常见的低级错误中最常见的就是关于“什么信息在什么地方”的决策错误。

在这样一本书的序言里说任何废话都将是佛头着粪。

所以，现在就祝你阅读愉快、编程愉快。

是为序。

前言

这是一本关于编程的书，更具体一点，是关于“如何编写别人能懂的代码”的书。编写出别人能读懂的代码没有任何神奇之处，这就与任何其他形式的写作一样：了解你的阅读者，在脑子里构想一个清晰的整体结构，让每个细节为故事的整体作出贡献。Java 提供了一些很好的交流机制，本书介绍的实现模式正是一些 Java 编程习惯，它们能让你编写出的代码更加易读。

也可以把实现模式看作思考“关于这段代码，我想要告诉阅读者什么？”的一种方式。程序员大部分的时间都在自己的世界里绞尽脑汁，以至于用别人的视角来看待世界对他们来说是一次重大的转变。他们不仅要考虑“计算机用这段代码做什么”，还要考虑“如何用这段代码与别人沟通我的想法”。这种视角上的转换有利于你的健康，也很可能有利于你的钱包，因为在软件开发中有大量的开销都被用在理解现有代码上了。

有一个叫做 Jeopardy 的美国游戏节目，由主持人给出问题的答案，参赛观众则来猜问题是什么。“猜一个词，表示扔出窗外。”“是 defenestration 吗？”“答对了。”

编程就好像 Jeopardy 游戏：答案用 Java 的基本语言构造给出，程序员则经常需要找出问题究竟是什么，即这些语言构造究竟是在解决什么问题。比如说，如果答案是“把一个字段声明为 Set”，那么问题可能就是“我要怎样告诉其他程序员，这是一个不允许包含重复元素的集合？”本书介绍的实现模式列举了一组常见的编程问题，还有 Java 解决这些问题的方式。

和软件开发一样，范围的管理对于写书同样重要。我现在就告诉你本书不是什么。它不是编程风格指南，因为其中包含了太多的解释，最终的决定权则完全交给你。它不是设计书籍，因为其中关注的主要是小范围的、程序

员每天要做很多次的决策。它不是模式书籍，因为这些实现模式的格式各不相同、随需而变。它不是语言书籍，因为尽管其中谈到了一些 Java 语言的特性，但我在写作时假设你已经熟悉 Java 了。

实际上本书建立在一个相当不可靠的前提之上：好的代码是有意义的。我见过太多丑陋的代码给它们的主人赚着大把钞票，所以在我看来，软件要取得商业成功或者被广泛使用，“好的代码质量”既不必要也不充分。即便如此，我仍然相信，尽管代码质量不能保证美好的未来，但它仍然有其意义：有了质量良好的代码以后，业务需求能够被充满信心地开发和交付，软件用户能够及时调整方向以便应对机遇和竞争，开发团队能够在挑战和挫折面前保持高昂的斗志。总而言之，比起质量低劣、错误重重的代码，好的代码更有可能帮助用户取得业务上的成功。

即便不考虑长期的经济效应，我仍然会选择尽我所能地编写出好代码。就算活到古稀之年，你的一生也只有二十多亿秒而已，这宝贵的时间不该被浪费在自己不能引以为傲的工作上。编写好的代码带给我满足感，不仅因为编程本身，还因为知道别人能够理解、欣赏、使用和扩展我的工作。

所以，说到底，这是一本关于责任的书。作为一个程序员，你拥有时间、拥有才华、拥有金钱、拥有机会。对于这些天赐的礼物，你要如何负责地使用？下面的篇幅里包含了我对于这个问题的答案：不仅为我自己、为我的 CPU 老弟编程，也为其他人编程。

致谢

我首先、最后并且始终要感谢 Cynthia Andres，我的搭档、编辑、支持者和首席讨债鬼。我的朋友 Paul Petralia 推动了这本书的写作，而且不断给我鼓励的电话。编辑 Chris Guzikowski 和我通过本书学会了如何在一起工作，他从 Pearson 的角度给了我一切需要的支持，让我能够完成写作。还要感谢 Pearson 的制作团队：Julie Nahil、John Fuller 和 Cynthia Kogut。Jennifer Kohnke

的插图不但包含了丰富的信息，而且非常人性化。本书的审阅者给我的书稿提供了清晰而又及时的反馈，为此我要感谢 Erich Gamma、Steve Metsker、Diomidis Spinellis、Tom deMarco、Michael Feathers、Doug Lea、Brad Abrams、Cliff Click、Pekka Abrahamson、Gregor Hohpe 和 Michele Marchesi。感谢 David Saff 指出“状态”与“行为”之间的平衡。最后，还要感谢我的孩子们，一想到他们乖乖呆在家里，我就有了尽快完成本书的动力。Lincoln、Lindsey、Forrest 和 Joëlle Andres，感谢你们。

目 录

第1章 引言	1
1.1 章节概览	4
1.2 那么，现在.....	5
第2章 模式	7
第3章 一种编程理论	11
3.1 价值观	12
3.1.1 沟通	12
3.1.2 简单	13
3.1.3 灵活	15
3.2 原则	16
3.2.1 局部化影响	16
3.2.2 最小化重复	16
3.2.3 将逻辑与数据捆绑	17
3.2.4 对称性	18
3.2.5 声明式表达	19
3.2.6 变化率	20
3.3 小结	21
第4章 动机	23
第5章 类	27
5.1 类	28
5.2 简单的超类名	29

目录

5.3	限定性的子类名	30
5.4	抽象接口	31
5.5	interface	33
5.6	抽象类	34
5.7	有版本的interface	35
5.8	值对象	36
5.9	特化	39
5.10	子类	40
5.11	实现器	42
5.12	内部类	43
5.13	实例特有的行为	44
5.14	条件语句	45
5.15	委派	47
5.16	可插拔的选择器	50
5.17	匿名内部类	51
5.18	库类	52
5.19	小结	53
第6章	状态	55
6.1	状态	56
6.2	访问	57
6.3	直接访问	58
6.4	间接访问	59
6.5	通用状态	60
6.6	可变状态	61
6.7	外生状态	63
6.8	变量	63
6.9	局部变量	65
6.10	字段	66
6.11	参数	68
6.12	收集参数	69

6.13	可选参数	70
6.14	变长参数	71
6.15	参数对象	72
6.16	常量	73
6.17	按角色命名	74
6.18	声明时的类型	75
6.19	初始化	76
6.20	及早初始化	77
6.21	延迟初始化	78
6.22	小结	78
第7章 行为		79
7.1	控制流	80
7.2	主体流	80
7.3	消息	81
7.4	选择性消息	82
7.5	双重分发	82
7.6	分解性（序列性）消息	84
7.7	反置性消息	84
7.8	邀请性消息	86
7.9	解释性消息	86
7.10	异常流	87
7.11	卫述句	88
7.12	异常	90
7.13	已检查异常	91
7.14	异常传播	91
7.15	小结	92
第8章 方法		93
8.1	组合方法	96
8.2	揭示意图的名称	97

目录

8.3	方法可见性	98
8.4	方法对象	100
8.5	覆盖方法	102
8.6	重载方法	103
8.7	方法返回类型	103
8.8	方法注释	104
8.9	助手方法	105
8.10	调试输出方法	106
8.11	转换	107
8.12	转换方法	107
8.13	转换构造器	108
8.14	创建	109
8.15	完整的构造器	110
8.16	工厂方法	111
8.17	内部工厂	111
8.18	容器访问器方法	112
8.19	布尔值设置方法	114
8.20	查询方法	115
8.21	相等性判断方法	116
8.22	取值方法	117
8.23	设置方法	118
8.24	安全复制	120
8.25	小结	121
第9章	容器	123
9.1	隐喻	124
9.2	要点	125
9.3	接口	127
9.3.1	Array	127
9.3.2	Iterable	128
9.3.3	Collection	128

9.3.4	List	129
9.3.5	Set	129
9.3.6	SortedSet	130
9.3.7	Map	131
9.4	实现	131
9.4.1	Collection	132
9.4.2	List	133
9.4.3	Set	134
9.4.4	Map	135
9.5	Collections	135
9.5.1	查询	136
9.5.2	排序	136
9.5.3	不可修改的容器	137
9.5.4	单元素容器	138
9.5.5	空容器	138
9.6	继承容器	139
9.7	小结	140
第10章	改进框架	141
10.1	修改框架而不修改应用	141
10.2	不兼容的更新	143
10.3	鼓励可兼容的变化	144
10.3.1	程序库类	145
10.3.2	对象	146
10.4	小结	155
附录A	性能度量	157
A.1	示例	158
A.2	API	158
A.3	实现	160
A.4	MethodTimer	160

目录

A.5 冲抵额外开销	163
A.6 测试	164
A.6.1 容器的比较	164
A.6.2 ArrayList 和 LinkedList 的比较	167
A.6.3 Set 之间的比较	169
A.6.4 Map 之间的比较	170
A.7 小结	172
参考书目	173

第 1 章

引言

现在开始吧。你选中了我的书（现在它就是你的了），你也编写过代码，很可能你已经从自己的经验中建立了一套属于自己的风格。

这本书的目标是要帮助你通过代码表达自己的意图。首先，我们对编程和模式做一个概述（第 2 章～第 4 章）。接下来（第 5 章～第 8 章）用一系列短文和模式，讲述了“如何用 Java 编写出可读的代码”。如果你正在编写框架（而不是应用程序），最后一章会告诉你如何调整前面给出的建议。总而言之，本书关注的焦点是用编程技巧来增进沟通。

用代码来沟通有几个步骤。首先，必须在编程时保持清醒。第一次开始记录实现模式时，我编程已经有一些年头了。我惊讶地发现，尽管能够快捷流畅地作出各种编程中的决定，但我没法解释自己为什么如此确定诸如“这个方法为什么应该被这样调用”或者“那块代码为什么属于那个对象”之类的事情。迈向沟通的第一步就是让自己慢下来，弄明白自己究竟想了些什么，不再假装自己是在凭本能编程。

第二步是要承认他人的重要性。我很早就发现编程是一件令人满足的事，但我是个以自我为中心的人，所以必须学会相信其他人也跟我一样重要，然后才能写出能与他人沟通的代码。编程很少会是一个人与一台机器之间孤独的交流，我们应该学会关心其他人，而这需要练习。

所以我迈出了第三步。一旦把自己的想法暴露在光天化日下，并且承认别人也有权和我一样地存在，我就必须实实在在地展示自己的新观点了。我使用本书中介绍的实现模式，目的是更有意识地编程，为他人编程，而不仅仅是为自己编程。

你当然可以仅为其中的技术内容——有用的技巧和解释——而阅读本书，但我认为应该预先提醒你，除了这些技术内容，本书还包含了很多东西，至少对我而言是这样。

这些技术内容都可以在介绍模式的章节里找到。学习这部分内容有一个高效的策略：需要用的时候再去读。如果用这种“just-in-time”的方式来读，那么可以直接跳到第5章，把后续的章节快速扫一遍，然后在编程时把本书放在手边。用过书中的很多模式之后，你可以重新回到前面介绍性的内容中来，读一下那些技巧背后的道理。如果有兴趣透彻理解手上的这本书，也可以细细地从头读到尾。和我写过的大部分书不同，这本书的每一章都相当长，因此在阅读时要保持专注才行。

书中的大部分内容都以模式的形式加以组织。编程中需要做的抉择大多曾经出现过。一个程序员的职业生涯中可能要给上百万个变量命名，不可能每次都用全新的方式来命名。命名的普遍约束总是一致的：需要把变量的用途、类型和生命周期告诉给阅读者，需要挑选一个容易读懂的名字，需要挑选一个容易写、符合标准格式的名字。把这些普遍约束加诸一个具体的变量之上，然后就得到了一个合用的名字。“给变量命名”就是一个模式：尽管每次都可能创造出不同的名字，但决策的方法和约束条件总是重复出现的。

我觉得，模式需要以多种形式来呈现，有时一篇充满争议的文章能最好地阐释一个模式，有时候是一幅图，有时候是一个故事，有时候是一段示例。所以我并没有尝试把所有模式都塞进同一种死板的格式里，而是以我认为最恰当的方式来描述它们。

书中总共包含了77个明确命名的模式，它们分别涵盖了“编写可读的代