Sun公司核心技术丛书

设计模式Java手册

Design Patterns Java workbook

(美) Steven John Metsker 著
龚波 冯军 程群梅 等译

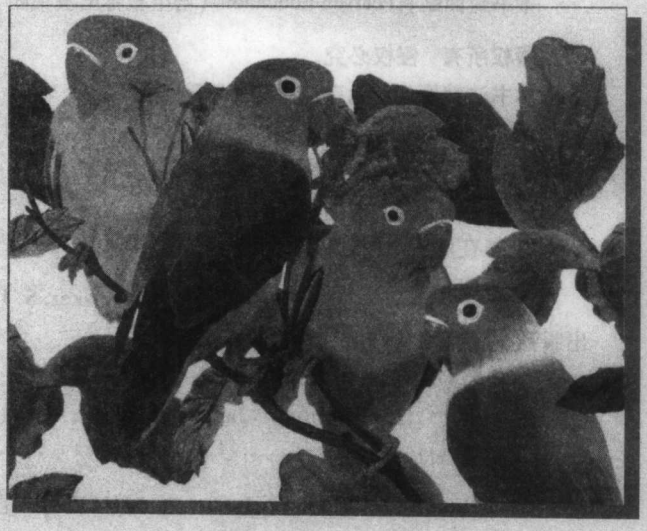
Rebecca Wirfs-Brock 序

机械工业出版社
China Machine Press

Sun公司核心技术丛书

设计模式Java手册

Design Patterns Java workbook



(美) Steven John Metsker 著

龚波 冯军 程群梅 等译

机械工业出版社
China Machine Press

设计模式是资深程序员们总结出来的一种可重用的、针对面向对象软件设计的解决方案，而本书借助Java语言为读者讲解了GoF《设计模式》中的全部23种设计模式。在本书中，这23种设计模式被组织成五类，它们分别是：接口型模式、责任型模式、构造型模式、操作型模式以及扩展型模式。本书的五个部分分别讲解了这五类设计模式，以帮助读者领会设计模式的思想及精华，然后再将它们融会贯通、灵活应用到自己的开发过程中。

本书可以作为软件项目管理人员、软件开发工程师等专业人员的指导用书，也可作为高等院校计算机及相关专业学生的参考书。

本书版权登记号：图字：01-2002-5451

图书在版编目（CIP）数据

设计模式Java手册 /（美）麦特思科（Metsker, S. J.）著；龚波等译. －北京：机械工业出版社，2006. 3

（Sun公司核心技术丛书）

书名原文：Design Patterns Java Workbook

ISBN 7-111-18395-9

Ⅰ. 设… Ⅱ. ①麦… ②龚… Ⅲ. JAVA语言－程序设计－手册 Ⅳ. TP312

中国版本图书馆CIP数据核字（2006）第004676号

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：刘 晖 李云静

北京瑞德印刷有限公司印刷 · 新华书店北京发行所发行

2006年3月第1版第1次印刷

787mm × 1020mm 1/16 · 21印张

定价：45.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线（010）68326294

接口型模式

适配器模式 旨在利用现有类所提供的服务，修改其接口，从而达到用户期望。
外观模式 旨在为子系统提供一个接口，使之更加容易被使用。
组合模式 旨在让用户能够用统一的接口处理单个对象以及对象组合。
桥接模式 旨在将抽象（依赖抽象操作的类）与抽象操作的实现相分离，从而使得抽象与实现能够独立变化。

责任型模式

单体模式 旨在确保某个类只有一个实例，并且为之提供一个全局访问点。
观察者模式 旨在在多个对象之间定义一个一对多的依赖关系，以便当一个对象状态改变的时候，其他所有依赖于这个对象的对象能够得到通知，并被自动更新。
中介者模式 旨在定义一个对象来封装一组对象之间交互的方式，这样就避免了对象间的显式引用，而且还可以单独对这些对象的交互进行修改。
代理模式 旨在为某个对象提供一个代理来控制对该对象的访问。
责任链模式 旨在将一个方法调用请求沿着责任链依次转发给下一个对象，让每个对象都有一次机会决定自己是否处理该请求，从而降低了请求的发送者与其接收者之间的耦合程度。
享元模式 旨在通过共享来为大量的细粒度对象提供有效的支持。

构造型模式

生成器模式 旨在把构造对象实例的代码逻辑移到类的外部。
工厂方法模式 旨在定义一个用于创建对象的接口，同时控制对哪个类进行实例化。
抽象工厂模式 旨在创建一系列相互关联或相互依赖的对象。
原型模式 通过拷贝一个现有对象生成新的对象。
备忘录模式 旨在为对象提供状态存储和状态恢复功能。

操作型模式

模板方法模式 旨在在一个方法中实现一个算法，并将算法中某些步骤的定义推迟，从而使得其他类可以重新定义这些步骤。

状 态 模 式 旨在将与状态有关的处理逻辑分散到代表对象状态的各个类中。

策 略 模 式 旨在把可选的策略或方案封装成在不同的类中，并在这些类中实现一个共同的操作。

命 令 模 式 旨在将请求封装为一个对象，并将该请求对象作为参数；用户可以提供不同的请求对象，如入队请求，时间请求，或者日志请求．也可以请求伴随操作，如undo( )操作。

解 释 器 模 式 旨在使用户可以根据自己定义的组合规则生成可执行的对象。

扩展型模式

装 饰 器 模 式 旨在使开发者能够动态地组织对象的行为。

迭 代 模 式 旨在为开发人员提供一种顺序访问集合元素的方法。

访 问 者 模 式 旨在让代码用户能够在不修改现有类层次结构的前提下扩展该类层次结构的行为。

本书获得的赞誉

“真是一本好书……它是如此深入浅出地讲解设计模式，给我留下了深刻的印象。我相信，只要是熟悉Java语言的读者，在看了这本书之后，也都会像我一样很容易地掌握其中各个章节的内容。现在许多地方都要求开发人员阅读本书，我的办公室也是这样要求的。”

——Joshua Engel

“这本书借助Java语言为读者提供了一种全新的方式来学习GoF的23种设计模式”

——Bob Hanmer

“这本书借助Java语言讲述了《Design Patterns》一书中所介绍的设计模式，从而方便Java程序员学习这些设计模式。本书各章给出了一些简短而又富于思考性的编程和设计问题，书后给出了参考答案。读者可以通过思考这些问题来加深自己对设计模式的理解。”

——Rebecca Wirfs-Brock

“这是一本激动人心的书籍。这本书浅显易懂，可读性强，内容有趣，教导性强，非常有价值。现在有许多书籍借助Java语言或其他语言来教导开发人员学习GoF设计模式，这本书是其中最好的。”

——John Vlissides

至 Alison

是她点燃了温馨的炉火，那温暖的火光闪烁在我们家中。

至 Emma-Kate 和 Sarah-Jane

我们可爱的孩子，顽皮的小精灵，
他们总是像花枝上的小鸟。

屋中消沉的火星，
微微地尚在闪耀；
跳跃着每个精灵。
像花枝上的小鸟；
随我唱一支曲调，
一齐轻轻地舞蹈。

——威廉·莎士比亚
摘自《仲夏夜之梦》

译 者 序

具体而言，设计模式就是利用面向对象技术解决特定环境中的问题的方法。它是整个软件开发行业不断积累的集体智慧的结晶。应用设计模式将有助于软件开发人员开发出逻辑清晰、架构合理、可重用性高的代码。同时，研究设计模式也方便软件开发人员交流软件及模块设计的经验和心得。目前，设计模式在软件开发过程中的作用已得到普遍的认可，学习和研究设计模式已成为各个层面软件开发人员的必修课。

本书借助Java语言为读者讲解了GoF的全部23种设计模式。本书把这23种设计模式组织成五类：接口型模式，责任型模式，构造型模式，操作型模式以及扩展型模式。本书分五个部分细致讲解了这五类设计模式。

每个部分的开头一章都会介绍Java语言内在支持的各种技术，以及这些技术的不足；随后各章将会对本类各个模式进行详细的讲解，通过列举一些应用实例，借助UML图例和Java代码来演示这些模式的应用方法，同时，本书还为读者留下了一些编程或设计方面的思考题来帮助读者举一反三。

本书的附录A为读者学习设计模式给出了一些具体建议。附录B给出了本书中思考题的参考答案。这些参考答案只是给了一种合理的解决方法，而不是唯一的。读者可以先经过思考给出自己的答案，然后再与书后的参考答案作比较。附录C简要描述了UML建模语言。之前不了解UML的读者在阅读了这节内容之后就可以看懂本书中的UML图例。

实践出真知！本书的最大特点就是便于边学边做。我们期望读者在阅读此书的时候也能够做到边学边做，这样才能加深自己对设计模式的理解，提高自己应用设计模式的能力。

在本书的翻译过程中，龚波、冯军和程群梅作出了最大的贡献，其他对本书翻译给予热情指导的还有：熊杰颖、李红玲、严亚军、陈蓓、易向东、徐小梅、张文敏等人，感谢他们的热心帮助和大力支持。

我们欢迎读者朋友将自己的意见和建议及时反馈给我们，通过电子邮件发送至wfstudio@vip.sina.com。我们乐意和您一起探讨设计模式相关技术问题，共同进步。

序

告诉我，我可能转眼就忘；教我，我会牢记在心；参与其中，我才能心领神会。

——本杰明·富兰克林

在本书中，Steve John Metsker为读者提供了非常丰富的有关设计模式的内容：书中不仅讲述了设计模式，还为读者提供了大量的代码实例，以及许多的思考题；完成这些思考题将帮助读者更加深入地理解设计模式。本书中的代码用于一家虚构的焰火制品公司，该公司除了生产销售焰火制品以外还对外提供焰火表演服务。这些代码实例比起以往常用的ATM机器的例子有趣多了；而且，读者在学习设计模式的过程中还可以了解有关焰火制品的知识。这本书不仅诱人而且有趣！另外，由于本书还描述了每种设计模式是如何应用于并扩展Java语言结构的，因而读者通过阅读此书还可以深入了解Java语言！

模式就是做事的方法，亦即实现某个目标的途径。设计模式就是利用面向对象技术实现某个目标的方法。该技术包括类，方法，继承，以及接口。每种模式都被概括为一个名字。如果同事之间都了解设计模式，那么他们的工作效率将会更高——因为他们都熟悉这些词汇，交流起来就会更加容易！这样开发人员之间就可以自由地讨论他们的意图，而不再苦于找不到合适的词汇用于交流。而对于那些习惯于应用设计模式的开发人员而言，有了通用的模式术语之后，他们的代码将会变得更加灵活，更加便于阅读和修改。

更早之前，Erich Gamma 及其同事曾写了一书描述设计模式的书，书名叫《Design Patterns》(Addison Wesley，1995，中文版《设计模式：可复用面向对象软件的基础》已由机械工业出版社出版——编注)。那本书深入介绍了23种已经过验证的构建和操作对象的设计模式。而本书则从Java程序员的角度清晰地解释了这23种设计模式。

在完成本书提供的自我突破思考题的过程中，读者会编写代码，扩展现有的代码，回答一些微妙的问题，以及解决软件设计过程中遇到的典型问题；通过对这些思考题的思索，读者将获得足够的机会来深化自己对设计模式的理解。不过，不论读多少书，学习知识的最佳方式是将其应用于实践。

Rebecca Wirfs-Brock

Sherwood 于俄勒冈州

2002年1月

前　言

2000年，在Minnesota Minneapolis举办的OOPSLA[⊖]上，我曾经咨询过Addison Wesley的Mike Hendrickson并问他他认为读者想要什么类型的书。当我听他说目前设计模式方面的图书仍有广阔的市场的时候，我对此很感兴趣，并向他提出应该有一本Java手册来帮助读者加深和体验他们对模式的理解。Mike认为这个想法很好，并把我介绍给Paul Becker。Paul Becker负责Addison Wesley软件模式系列的图书。Mike的第一反应是“在五年之前就应出这种书”。我非常感谢Mike和Paul早期给予我的鼓励，是他们启发我写了这样一本书。

自从那次初次会面之后，在这本书的写作过程中，Paul一直给予我大力的支持，直到这本书出版。在这个项目的早期，Paul曾邀请软件模式系列图书的编辑John Vlissides一起审查这个图书项目。John评价说Paul应该全力支持这个项目。在后来的写书过程中，John的话深深地印在了我的脑海中。

John Vlissides当然也是《设计模式》一书的四个作者之一。John与其他合作者——Erich Gamma，Ralph Johnson，以及Richard Helm——写的那本书完全是本书的基石。在写书的过程中，我几乎每天都要参考《设计模式》一书，我一点儿也没有夸大那本书的作用。

另外，我还参考了其他许多书籍，本书后面的参考文献分别将它们一一列出。特别值得一提的是，我参考了《*The Unified Modeling Language User Guide*》（Booch，Rumbaugh，and Jacobson 1999，中文版《UML用户指南》已由机械工业出版社引进出版——编注）一书，阅读了其中对UML的详细解释。为了准确地描述Java相关内容，我几乎每天都参考了《Java™ in a Nutshell》（Flanagan 1999b）一书。另外，我还参考了《Patterns in Java™》（Grand 1998）和《Java™ Design Patterns》（Cooper 2000）这两本书。

在写书的这几个月里，我还在一家金融服务机构工作。该机构在许多地方都有分部。当这本书初稿完成的时候，我开了一门指导课程，专门以这本书为教材。当时，我在Richmond和Virginia教这门课程；而我的同事Tim Snyder和Bill Trudell在其他地方教这门课程。我想在这里谢谢我的这两位同事以及听过这门课的学生们，谢谢他们给予了我灵感，也谢谢他们将自己的见解告诉我。特别地，我还要感谢Srinivasarao Katepalli、Brad Hughes、Thiaga Manian、Randy Fields、Macon Pegram、Joe Paulchell、Ron DiFrango、Ritch Linklater、Patti Richards以及Ben Lewis，谢谢他们给予我的帮助和建议。我要感谢我的朋友Bill Wake和Gagan Kanjlia在写作早期帮我审阅这本书，以及Kiran Raghunathan在写作后期给予我帮助。最后，我还要感谢我的朋友Jeff Damukaitis，

⊖ OOPSLA是由ACM（Association for Computing Machinery）主办的一个年会；该会主要探讨有关面向对象编程、系统以及应用等主题。

他给了我一些不错的建议。

在这本书的写作过程中，Paul Becker安排了许多出色的评审老师来指导我写作。我要再次感谢John Vlissides对这本书的审阅。他每次审阅总能指出需要改进的关键之处，我看得出他是真的喜欢这本书。我要感谢Luke Hohmann，Bob Hanmer，Robert Martin，以及Joshua Kerievsky等人，他们在不同阶段帮助过我，是他们的努力使得这本书更好。我要感谢Joshua Engel，他总是能以一种温文尔雅的方式来表达自己尖锐的观点。最后，我要感谢Rebecca Wirfs-Brock，他给了我许多非常有价值的建议，包括建议我重新组织本书。在写书伊始，我并没有特意将那些重要且易于理解的模式安排在书的前面。正是听从了Rebecca的建议以及在本书所有审阅者的帮助下，这本书才变得更加出色。

Steve Metsker（Steve.Metsker@acm.org）

目　录

第三部分 构造型模式

第四部分 操作型模式

第五部分 扩展型模式

第1章 模式概述

本书所面向的读者是那些了解Java语言，并且浏览过《设计模式》（Design Patterns）（Gamma等1995）一书的软件开发人员。通过阅读本书，读者朋友们将会有以下收获：

- 加深对《设计模式》一书中所描述的各种设计模式的理解。
- 增强识别各种设计模式的信心。
- 增强在Java程序中应用设计模式的能力。

1.1 为什么使用模式

模式（pattern）就是做事的方法，亦即实现某个目标的途径。这种思想可以应用到许多工作当中，比如烹饪，生产焰火制品，开发软件，以及其他一些工作。在任何一个成熟的或者正迈向成熟的行业，人们总会找到一些通用且有效的解决方法来解决不同场合中的问题。某个行业的从业人员在工作的过程中会形成自己的专业术语以便于同行之间进行交流。模式即实现某个目的的标准化做法，它们往往被总结为一些专业术语。一些人将自己所在行业的模式记录归档，从而使得行业的专业术语得到标准化，并使自己行业积累起来的经验可供未来的从业者使用。

Christopher Alexander是最早将某个行业的最佳实践记录为模式的作者之一。他研究的领域是建筑的体系结构，而不是软件。其著作《A Pattern Language: Towns, Buildings, Construction》（Alexander，Ishikawa以及Silverstein 1977）一书介绍了成功地建造房屋和城市的通用模式。该著作影响深远，甚至影响到软件开发行业。该书之所以能够影响其他行业，部分原因是因为它给出了一种独特的观察目标的方式。

有人可能会认为，应用建筑模式的目的就是“设计建筑物”。但是Alexander在书中明确指出建筑模式的目的是服务于建筑物和城市的主人，并让他们感到满意。该书指出模式是获取和传播某个行业知识的最佳途径。另外，书中还提到，合理地认识并记录每项工作的目的非常关键，需要以哲学观从整体考虑，另外还可能面临难以捉摸的挑战。

软件行业的人们对Alexander的思维方式产生了强大的共鸣，并出版了大量的书籍来记录软件开发的模式。这些书记录着软件开发流程、软件分析、高级软件设计以及软件在类级别设计的最佳实践。表1-1列出了一些书籍，它们分别描述了软件开发各个方面的最佳实践。该列表并不全面，每年都有这方面的新书出版。如果需要选择一本有关设计模式的图书，那么你应该先花些时间去浏览一下已出版的关于这方面图书的书评，然后再选择一本能够最大程度帮助自己的图书。

表1-1 以模式形式传播软件开发知识的图书

模式种类	书 名	作者/编者
软件过程	《*Process Patterns: Building Large-Scale Systems Using Object Technology*》	Scott W. Ambler
	《*More Process Patterns: Delivering Large-Scale Systems Using Object Technology*》	Scott W. Ambler
对象建模	《*Analysis Patterns: Reusable Object Models*》	Martin Fowler
	《*Object Models: Strategies, Patterns and Applications*》	Peter Coad Mark Mayfield David North
体系架构	《*CORBA Design Patterns*》	Thomas J. Mowbray Raphael C. Malveau
	《*Core J2EE™ Patterns: Best Practices and Design Strategies*》（中文版《J2EE核心模式》已由机械工业出版社引进出版—编注）	Deepak Alur John Crupi Dan Malks
	《*Pattern-Oriented Software Architecture,Volume 1: A System of Patterns*》	Frank Buschmann Regine Meunier Hans Rohnert Peter Sommerlad Michael Stal
	《*Pattern-Oriented Software Architecture, Volume 2: Patterns for Concurrent and Networked Objects*》	Douglas Schmidt Michael Stal Hans Rohnert Frank Buschmann
软件设计	《*AntiPatterns: Refactoring Software,Architectures, and Projects in Crisis*》	William J. Brown Raphael C. Malveau Hays W. McCormick III Thomas J. Mowbray
	《*Applying UML and Patterns, Second Edition*》	Craig Larman
	《*Concurrent Programming in Java™, Second Edition: Design Principles and Patterns*》	Doug Lea
	《*Design Patterns*》（设计模式）	Erich Gamma Richard Helm Ralph Johnson John Vlissides
	《*Design Patterns for Object-Oriented Software Development*》	Wolfgang Pree
	《*Pattern Hatching: Design Patterns Applied*》	John Vlissides
	《*SanFrancisco™ Design Patterns*》	James Carey Brent Carlson Tim Graser