

Apress®

Spring Recipes

A Problem-Solution Approach

SECOND EDITION

涵盖
Spring 3.0框架

Spring 攻略

(第2版)

[美] Gary Mak Josh Long Daniel Rubio 著

陈宗恒 姚军 蒋亮 译

- Spring经典畅销书最新版
- 被誉为Spring开发“百科全书”
- 海量编码技巧助你释放Spring 3.0无限潜能



人民邮电出版社
POSTS & TELECOM PRESS

Spring 攻略

(第2版)

[美] Gary Mak Josh Long Daniel Rubio 著
陈宗恒 姚军 蒋亮 译

人民邮电出版社

北京

图书在版编目 (C I P) 数据

Spring 攻略 : 第2版 / (美) 麦克 (Mak, G.), (美) 隆 (Long, J.), (美) 卢比奥 (Rubio, D.) 著 ; 陈宗恒, 姚军, 蒋亮译. — 北京 : 人民邮电出版社, 2012. 3
ISBN 978-7-115-27449-6

I. ①S… II. ①麦… ②隆… ③卢… ④陈… ⑤姚…
⑥蒋… III. ①JAVA语言—程序设计 IV. ①TP312

中国版本图书馆CIP数据核字 (2012) 第015823号

版权声明

Original English language edition, entitled Spring Recipes: A Problem-Solution Approach by Gary Mak, Josh Long, Daniel Rubio, published by Apress 2855 Telegraph Avenue, #600, Berkeley, CA 94705 USA.

Copyright (c) 2010 by Apress L.P. Simplified Chinese-language edition copyright(c) 2012 by POSTS & TELECOMMUNICATIONS PRESS. All rights reserved.

本书中文简体字版由Apress L.P.授权人民邮电出版社独家出版。未经出版者书面许可, 不得以任何方式复制或抄袭本书内容。版权所有, 侵权必究。

内 容 提 要

本书以大量的实例, 全面透彻地揭示了 Spring 框架的各项特性以及围绕该框架新推出的许多周边框架, 以实际问题—解决方案—具体做法的方式, 为读者展示了这一流行框架从基本概念到各种应用, 最后到企业集成的各种实际运用, 是 Spring 框架使用者必备的完全指南。

本书适合于对 Java 开发和企业应用集成有一定了解, 希望在实际开发中掌握一种全面、快速、可伸缩、可移植的工具平台的开发人员。

Spring 攻略 (第 2 版)

-
- ◆ 著 [美] Gary Mak Josh Long Daniel Rubio
译 陈宗恒 姚 军 蒋 亮
责任编辑 汪 振
 - ◆ 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>
北京艺辉印刷有限公司印刷
 - ◆ 开本: 800×1000 1/16
印张: 60.25
字数: 1322 千字 2012 年 3 月第 1 版
印数: 1—3 000 册 2012 年 3 月北京第 1 次印刷

著作权合同登记号 图字: 01-2009-7793 号

ISBN 978-7-115-27449-6

定价: 128.00 元

读者服务热线: (010)67132705 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

前言

Spring 框架正在成长。它始终与选择相关。Java EE 关注于少数几项技术，很大程度上阻碍了更好的替代解决方案出现。当 Spring 框架出现时，没有多少人还会承认 Java EE 是当今最佳的架构。

随后 Spring 被大张旗鼓地推出，因为它寻求简化 Java EE。此后其每个版本都引入设计用来简化和实现解决方案的新特性。

从 2.0 版本之后，Spring 框架开始针对多平台。和往常一样，该框架提供了现有平台之上的服务，但是尽可能去除与底层平台的耦合。Java EE 仍然是主要的参考点，但是不是唯一的目标。OSGi（一种有前景的模块化架构技术）已经成为 SpringSource 战略的重要部分。而且，Spring framework 在 Google App Engine 之上运行。引入注解为中心的框架和 XML schema，SpringSource 已经建立了有效地构造特定问题域模型的框架，实际上创建了领域特定语言（DSL）。

如今建立在 Spring 框架之上的框架已经出现，支持应用集成、批处理、Flex 和 Flash 集成、GWT、OSGi 和许多其他技术。

在更新开创性的《Spring Recipes》的时候，我们很快发现，很长的时间实际上只有一个核心的 Spring 框架。尽管如此，SpringSource portfolio 还是描述了多个框架，每个框架都远比其他产品中的竞争对手强大。本书将很好地带你经历各种框架。如果你不需要这些技术，就没有必要在你的项目中使用或者添加它们。如果你需要，知道它们的存在是很好的事情。

本书的读者

本书是为希望简化架构和解决 Java EE 平台之外的问题的 Java 开发人员编写的。如果你已经在项目中使用 Spring 进行开发，更高级的章节讨论了你可能尚不知晓的新技术。如果你对于这个框架来说是新手，本书将马上引你入门。

本书假定你对 Java 和某种 IDE 已经有了一定的熟悉。虽然有可能在客户端应用中仅使用 Java，这也确实有用，但是 Java 的最大社区存在于企业空间中，而且企业也是你能看到的大部分技术能发挥最大作用的地方。因此，本书也假定读者对基本企业编程概念如 Servlet API 有一定的熟悉。

本书的结构

第 1 章,“Spring 简介”给出了 Spring 框架的一般概述:如何安装,什么是 Spring 框架,以及如何使用它。

第 2 章,“高级 Spring IoC 容器”研究了不像第 1 章中讨论的概念那么广泛使用,但对于完全利用该容器仍很重要的一些概念。

第 3 章,“Spring AOP 和 AspectJ 支持”讨论了 Spring 对使用 AspectJ 进行面向方面编程的支持,这种技术构成许多 Spring 框架提供的其他服务的基础。

第 4 章,“Spring 中的脚本”讨论 Spring 框架中 Groovy、BeanShell 和 JRuby 这样的脚本语言的使用。

第 5 章,“Spring Security”提供对 Spring Security 项目的概述,这个项目过去叫做 Acegi,用于帮助你更好地加强应用程序的安全。

第 6 章,“将 Spring 与其他 Web 框架集成”介绍 Spring 提供的核心 Web 层支持,这为 Spring 在 Web 层中提供的所有技术打下基础。

第 7 章,“Spring Web Flow”提供 Spring Web Flow 的简介,这让你在 Web 层之上构建 UIflow。

第 8 章,“Spring @MVC”介绍了使用 Spring Web MVC 框架的基于 Web 应用开发。

第 9 章,“Spring REST”提供 Spring 对 REST 风格的 Web 服务的支持。

第 10 章,“Spring 和 Flex”讨论使用 Spring BlazeDS 来将你的富互联网应用(RIA)与 Spring bean 集成。

第 11 章,“Grails”讨论 Grails 框架,它可以使用最优的部件并且用 Groovy 代码将它们粘合在一起,以此提高你的生产率。

第 12 章,“Spring Roo”介绍 Spring Roo,这个来自 SpringSource 的新的关键框架设计用于为 Java 开发人员提供力量倍增框架。

第 13 章,“Spring 测试”讨论 Spring 框架中的单元测试。

第 14 章,“Spring Portlet MVC 框架”介绍使用 Spring Portlet MVC 框架构建应用,以及 Portlet 容器长处的利用。

第 15 章,“数据访问”讨论 Spring 与使用 JDBC、Hibernate 和 JPA 这样的 API 的数据存储之间的交互。

第 16 章,“Spring 中的事务管理”介绍 Spring 健壮的事务管理机制背后的概念。

第 17 章,“EJB、Spring Remoting 和 Web 服务”介绍各种 RPC 机制,包括 Spring Web Services 项目。

第 18 章,“企业中的 Spring”讨论 Spring 平台提供的许多实用程序(如 JMX 支持、计划任务以及电子邮件支持)。

第 19 章，“消息”讨论使用 Spring 通过 JMS 的面向消息中间件以及简化 Spring 抽象。

第 20 章，“Spring Integration”讨论使用 Spring Integration 框架集成不同的服务和数据。

第 21 章，“Spring Batch”介绍 Spring Batch 框架，它提供了一种方法以建立传统上被看作主机领域的解决方案的模型。

第 22 章，“分布式的 Spring”讨论了使用分布状态和网格处理进行 Spring 缩放的各种方式。

第 23 章，“Spring 和 jBPM”为你介绍业务过程管理概念以及如何将流行的框架——JBoss 的 jBPM 与 Spring 框架集成。

第 24 章，“OSGi and Spring”简单地介绍 Spring 框架提供的 OSGi 支持。

惯例

有时，当我们希望你特别注意代码示例中的一个部分时，会使用粗体。请注意，粗体字并不一定反映代码与前一版本中相比有改动。

如果代码行超过页宽，我们将用代码续行符来换行。请注意当你试图输入代码时，必须自己连接该行，不能有任何空格。

前提

因为 Java 编程语言是平台独立的，你可以自由地选择任何支持的操作系统。但是，本书的某些示例使用平台相关的路径。在输入示例之前必须将它们转换成你的操作系统的格式。

为了最大限度地利用本书，安装 JDK 版本 1.5 或者更高版本。你应该安装一个 Java IDE 来简化开发。对于本书，样板代码是基于 Maven 的。如果你运行 Eclipse 并安装 m2Eclipse 插件，可以在 Eclipse 中打开相同的代码，CLASSPATH 和依赖将由 Maven 元数据填写。

如果你使用 Eclipse，可能更喜欢 SpringSource 的 SpringSource 工具套件（STS），因为它预先装入在 Eclipse 中更有效使用 Spring 框架所需的插件。如果你使用 NetBeans 或 IntelliJ IDEA，就没有特殊的配置要求：它们已经支持 Maven。

本书使用 Maven 是因为 Spring 框架从版本 3.0.3 开始，不再带有使用该框架所需的所有依赖。建议的方法是简单地使用 Maven（或者 Ant 和 Ivy）这样的工具来处理依赖管理。如果你不熟悉 Maven，可以先简单地看看第 12 章（Spring Roo），那里我们介绍了 Spring Roo 环境的设置，包括 Apache Maven。

下载代码

本书的源代码在 Apress 网站（www.apress.com）的 Source Code /Download（源代码/下

载)部分中可以找到。源代码按照章节组织,每个都包含一个或者多个独立的示例。

与作者联络

我们始终欢迎关于本书内容的问题和反馈。你可以发邮件到 josh@joshlong.com (或者通过他的网站 www.joshlong.com) 与 Josh Long 联系。可以通过电子邮件地址 springrecipes@metaarchit.com (或者通过他的网站 www.metarchit.com) 与 Gary Mak 联系。你也可以通过 Daniel Rubio 的网站 www.webforefront.com 与他联系。

目 录

第 1 章 Spring 简介.....1	
1.1 实例化 Spring IoC 容器.....1	
1.1.1 问题.....1	
1.1.2 解决方案.....1	
1.1.3 工作原理.....3	
1.2 配置 Spring IoC 容器中的 Bean.....4	
1.2.1 问题.....4	
1.2.2 解决方案.....4	
1.2.3 工作原理.....4	
1.3 调用构造程序创建 Bean.....14	
1.3.1 问题.....14	
1.3.2 解决方案.....14	
1.3.3 工作原理.....14	
1.4 解决构造程序歧义.....17	
1.4.1 问题.....17	
1.4.2 解决方案.....17	
1.4.3 工作原理.....17	
1.5 指定 Bean 引用.....20	
1.5.1 问题.....20	
1.5.2 解决方案.....20	
1.5.3 工作原理.....20	
1.6 为集合元素指定数据类型.....24	
1.6.1 问题.....24	
1.6.2 解决方案.....24	
1.6.3 工作原理.....24	
1.7 使用 Spring 的 FactoryBean.....27	
创建 Bean.....27	
1.7.1 问题.....27	
1.7.2 解决方案.....27	
1.7.3 工作原理.....27	
1.8 使用工厂 Bean 和 Utility Schema.....29	
定义集合.....29	
1.8.1 问题.....29	
1.8.2 解决方案.....29	
1.8.3 工作原理.....29	
1.9 用依赖检查属性.....31	
1.9.1 问题.....31	
1.9.2 解决方案.....32	
1.9.3 工作原理.....32	
1.10 用@Required 注解检查属性.....34	
1.10.1 问题.....34	
1.10.2 解决方案.....34	
1.10.3 工作原理.....34	
1.11 用 XML 配置自动装配 Bean.....36	
1.11.1 问题.....36	
1.11.2 解决方案.....36	
1.11.3 工作原理.....37	
1.12 用@Autowired 和@Resource.....41	
自动装配 Bean.....41	
1.12.1 问题.....41	
1.12.2 解决方案.....41	
1.12.3 工作原理.....41	
1.13 继承 Bean 配置.....47	
1.13.1 问题.....47	
1.13.2 解决方案.....47	
1.13.3 工作原理.....48	
1.14 从 Classpath 中扫描组件.....50	
1.14.1 问题.....50	
1.14.2 解决方案.....51	

1.14.3 工作原理	51	2.8.2 解决方案	77
1.15 小结	56	2.8.3 工作原理	77
第 2 章 高级 Spring IoC 容器	57	2.9 使 Bean 感知容器	81
2.1 调用静态工厂方法创建 Bean	57	2.9.1 问题	81
2.1.1 问题	57	2.9.2 解决方案	81
2.1.2 解决方案	57	2.9.3 工作原理	82
2.1.3 工作原理	57	2.10 加载外部资源	82
2.2 调用一个实例工厂方法创建 Bean	58	2.10.1 问题	82
2.2.1 问题	58	2.10.2 解决方案	83
2.2.2 解决方案	59	2.10.3 工作原理	83
2.2.3 工作原理	59	2.11 创建 Bean 后处理器	85
2.3 从静态字段中声明 Bean	60	2.11.1 问题	85
2.3.1 问题	60	2.11.2 解决方案	85
2.3.2 解决方案	60	2.11.3 工作原理	86
2.3.3 工作原理	61	2.12 外部化 Bean 配置	89
2.4 从对象属性中声明 Bean	62	2.12.1 问题	89
2.4.1 问题	62	2.12.2 解决方案	89
2.4.2 解决方案	62	2.12.3 工作原理	90
2.4.3 工作原理	62	2.13 解析文本消息	91
2.5 使用 Spring 表达式语言	64	2.13.1 问题	91
2.5.1 问题	64	2.13.2 解决方案	91
2.5.2 解决方案	64	2.13.3 工作原理	91
2.5.3 工作原理	65	2.14 使用应用事件进行通信	93
2.6 设置 Bean 作用域	69	2.14.1 问题	93
2.6.1 问题	69	2.14.2 解决方案	93
2.6.2 解决方案	69	2.14.3 工作原理	94
2.6.3 工作原理	70	2.15 在 Spring 中注册属性编辑器	96
2.7 自定义 Bean 初始化和析构	72	2.15.1 问题	96
2.7.1 问题	72	2.15.2 解决方案	96
2.7.2 解决方案	72	2.15.3 工作原理	97
2.7.3 工作原理	72	2.16 创建自定义属性编辑器	99
2.8 用 Java Config 简化 XML 配置	77	2.16.1 问题	99
2.8.1 问题	77	2.16.2 解决方案	100

2.16.3 工作原理	100	3.7.1 问题	132
2.17 使用 TaskExecutor 实现 并发性	101	3.7.2 解决方案	132
2.17.1 问题	101	3.7.3 工作原理	132
2.17.2 解决方案	101	3.8 为你的 Bean 引入状态	135
2.17.3 工作原理	102	3.8.1 问题	135
2.18 小结	110	3.8.2 解决方案	135
第 3 章 Spring AOP 和 AspectJ 支持	112	3.8.3 工作原理	135
3.1 启用 Spring 的 AspectJ 注解 支持	113	3.9 用基于 XML 的配置 声明 aspect	137
3.1.1 问题	113	3.9.1 问题	137
3.1.2 解决方案	113	3.9.2 解决方案	137
3.1.3 工作原理	113	3.9.3 工作原理	137
3.2 用 AspectJ 注解声明 aspect	115	3.10 Spring 中的 AspectJ 加载时 织入 aspect	140
3.2.1 问题	115	3.10.1 问题	140
3.2.2 解决方案	115	3.10.2 解决方案	141
3.2.3 工作原理	116	3.10.3 工作原理	141
3.3 访问连接点信息	121	3.11 在 Spring 中配置 AspectJ aspect	146
3.3.1 问题	121	3.11.1 问题	146
3.3.2 解决方案	122	3.11.2 解决方案	146
3.3.3 工作原理	122	3.11.3 工作原理	146
3.4 指定 aspect 优先级	123	3.12 将 Spring Bean 注入领域 对象	147
3.4.1 问题	123	3.12.1 问题	147
3.4.2 解决方案	123	3.12.2 解决方案	147
3.4.3 工作原理	123	3.12.3 工作原理	148
3.5 重用切入点定义	125	3.13 小结	151
3.5.1 问题	125	第 4 章 Spring 中的脚本	152
3.5.2 解决方案	125	4.1 用脚本语言实现 Bean	152
3.5.3 工作原理	125	4.1.1 问题	152
3.6 编写 AspectJ 切入点表达式	127	4.1.2 解决方案	153
3.6.1 问题	127	4.1.3 工作原理	153
3.6.2 解决方案	127	4.2 将 Spring Bean 注入脚本中	157
3.6.3 工作原理	128		
3.7 在你的 Bean 中引入行为	132		

4.2.1 问题	157	5.6.1 问题	196
4.2.2 解决方案	157	5.6.2 解决方案	196
4.2.3 工作原理	157	5.6.3 工作原理	196
4.3 从脚本中刷新 Bean	160	5.7 处理领域对象安全性	198
4.3.1 问题	160	5.7.1 问题	198
4.3.2 解决方案	160	5.7.2 解决方案	198
4.3.3 工作原理	160	5.7.3 工作原理	199
4.4 定义内联脚本源码	161	5.8 小结	208
4.4.1 问题	161		
4.4.2 解决方案	161		
4.4.3 工作原理	161		
4.5 小结	163		
第 5 章 Spring Security	164	第 6 章 将 Spring 与其他 Web 框架集成	209
5.1 加强 URL 访问安全	165	6.1 在一般 Web 应用中访问 Spring	209
5.1.1 问题	165	6.1.1 问题	209
5.1.2 解决方案	165	6.1.2 解决方案	210
5.1.3 工作原理	166	6.1.3 工作原理	210
5.2 登录到 Web 应用	175	6.2 在你的 Servlet 和过滤器中使用 Spring	214
5.2.1 问题	175	6.2.1 问题	214
5.2.2 解决方案	175	6.2.2 解决方案	215
5.2.3 工作原理	175	6.2.3 工作原理	215
5.3 验证用户	179	6.3 将 Spring 与 Struts 1.x 集成	220
5.3.1 问题	179	6.3.1 问题	220
5.3.2 解决方案	180	6.3.2 解决方案	220
5.3.3 工作原理	180	6.3.3 工作原理	220
5.4 做出访问控制决策	190	6.4 将 Spring 与 JSF 集成	226
5.4.1 问题	190	6.4.1 问题	226
5.4.2 解决方案	190	6.4.2 解决方案	226
5.4.3 工作原理	191	6.4.3 工作原理	227
5.5 加强方法调用的安全	193	6.5 将 Spring 与 DWR 集成	232
5.5.1 问题	193	6.5.1 问题	232
5.5.2 解决方案	193	6.5.2 解决方案	232
5.5.3 工作原理	194	6.5.3 工作原理	233
5.6 处理视图中的安全性	196	6.6 小结	236

第 7 章 Spring Web Flow	238	8.1.1 问题	280
7.1 用 Spring Web Flow 管理简单的 UI 流程	238	8.1.2 解决方案	281
7.1.1 问题	238	8.1.3 工作原理	283
7.1.2 解决方案	239	8.2 用 @RequestMapping 映射请求	293
7.1.3 工作原理	239	8.2.1 问题	293
7.2 用不同状态类型建立 Web 流程模型	246	8.2.2 解决方案	294
7.2.1 问题	246	8.2.3 工作原理	294
7.2.2 解决方案	246	8.3 用处理程序拦截器拦截请求	297
7.2.3 工作原理	246	8.3.1 问题	297
7.3 加强 Web 流程安全	257	8.3.2 解决方案	298
7.3.1 问题	257	8.3.3 工作原理	298
7.3.2 解决方案	258	8.4 解析用户区域	302
7.3.3 工作原理	258	8.4.1 问题	302
7.4 持续存储 Web 流程中的对象	260	8.4.2 解决方案	302
7.4.1 问题	260	8.4.3 工作原理	302
7.4.2 解决方案	260	8.5 外部化区分区域的文本信息	304
7.4.3 工作原理	260	8.5.1 问题	304
7.5 将 Spring Web Flow 与 JSF 集成	267	8.5.2 解决方案	304
7.5.1 问题	267	8.5.3 工作原理	305
7.5.2 解决方案	267	8.6 按照名称解析视图	306
7.5.3 工作原理	267	8.6.1 问题	306
7.6 使用 RichFaces 与 Spring Web Flow 协作	275	8.6.2 解决方案	306
7.6.1 问题	275	8.6.3 工作原理	306
7.6.2 解决方案	275	8.7 视图和内容协商	309
7.6.3 方法	275	8.7.1 问题	309
7.7 小结	279	8.7.2 解决方案	309
		8.7.3 工作原理	309
第 8 章 Spring @MVC	280	8.8 映射异常视图	312
8.1 用 Spring MVC 开发简单的 Web 应用	280	8.8.1 问题	312
		8.8.2 解决方案	312
		8.8.3 工作原理	312
		8.9 用 @Value 在控制器中赋值	314
		8.9.1 问题	314
		8.9.2 解决方案	314
		8.9.3 工作原理	314

8.10 用控制器处理表单	316	9.4.1 问题	372
8.10.1 问题	316	9.4.2 解决方案	372
8.10.2 解决方案	316	9.4.3 工作原理	372
8.10.3 工作原理	317	9.5 访问具有复杂 XML 响应的	
8.11 用向导表单控制器处理多页		REST 服务	375
表单	331	9.5.1 问题	375
8.11.1 问题	331	9.5.2 解决方案	375
8.11.2 解决方案	331	9.5.3 工作原理	375
8.11.3 工作原理	332	9.6 小结	385
8.12 使用注解 (JSR-303) 的 Bean		第 10 章 Spring 和 Flex	386
校验	341	10.1 Flex 入门	388
8.12.1 问题	341	10.1.1 问题	388
8.12.2 解决方案	342	10.1.2 解决方案	388
8.12.3 工作原理	342	10.1.3 工作原理	388
8.13 创建 Excel 和 PDF 视图	344	10.2 离开沙箱	393
8.13.1 问题	344	10.2.1 问题	393
8.13.2 解决方案	345	10.2.2 解决方案	394
8.13.3 工作原理	345	10.2.3 工作原理	394
8.14 小结	351	10.3 为应用添加 Spring BlazeDS	
第 9 章 Spring REST	352	支持	406
9.1 用 Spring 发布一个 REST		10.3.1 问题	406
服务	352	10.3.2 解决方案	406
9.1.1 问题	352	10.3.3 工作原理	406
9.1.2 解决方案	353	10.4 通过 BlazeDS/Spring 暴露	
9.1.3 工作原理	353	服务	411
9.2 用 Spring 访问 REST 服务	358	10.4.1 问题	411
9.2.1 问题	358	10.4.2 解决方案	411
9.2.2 解决方案	358	10.4.3 工作原理	411
9.2.3 工作原理	358	10.5 使用服务器端对象	418
9.3 发布 RSS 和 Atom 信息源	362	10.5.1 问题	418
9.3.1 问题	362	10.5.2 解决方案	418
9.3.2 解决方案	363	10.5.3 工作原理	418
9.3.3 工作原理	363	10.6 使用 BlazeDS 和 Spring 消费	
9.4 用 REST 服务发布 JSON	372	面向消息的服务	421

10.6.1 问题	421	11.6.2 解决方案	454
10.6.2 解决方案	422	11.6.3 工作原理	455
10.6.3 工作原理	422	11.7 国际化 (I18n) 信息属性	458
10.7 将依赖注入带给你的 ActionScript 客户	434	11.7.1 问题	458
10.7.1 问题	434	11.7.2 解决方案	458
10.7.2 解决方案	434	11.7.3 工作原理	458
10.7.3 工作原理	435	11.8 改变永久性存储系统	461
10.8 小结	439	11.8.1 问题	461
第 11 章 Grails	441	11.8.2 解决方案	461
11.1 获取和安装 Grails	441	11.4.3 工作原理	461
11.1.1 问题	441	11.9 日志	464
11.1.2 解决方案	442	11.9.1 问题	464
11.1.3 工作原理	442	11.9.2 解决方案	464
11.2 创建 Grails 应用	443	11.9.3 工作原理	464
11.2.1 问题	443	11.10 运行单元和集成测试	466
11.2.2 解决方案	443	11.10.1 问题	466
11.2.3 工作原理	443	11.10.2 解决方案	467
11.3 Grails 插件	447	11.10.3 工作原理	467
11.3.1 问题	447	11.11 使用自定义布局和模板	472
11.3.2 解决方案	448	11.11.1 问题	472
11.3.3 工作原理	448	11.11.2 解决方案	472
11.4 在 Grails 环境中开发、生产和 测试	449	11.11.3 工作原理	472
11.4.1 问题	449	11.12 使用 GORM 查询	475
11.4.2 解决方案	449	11.12.1 问题	475
11.4.3 工作原理	450	11.12.2 解决方案	475
11.5 创建应用的领域类	452	11.12.3 工作原理	475
11.5.1 问题	452	11.13 创建自定义标记	477
11.5.2 解决方案	452	11.13.1 问题	477
11.5.3 工作原理	452	11.13.2 解决方案	477
11.6 为一个应用的领域类生成 CRUD 控制器和视图	454	11.13.3 工作原理	478
11.6.1 问题	454	11.14 小结	479
		第 12 章 Spring Roo	481
		12.1 设置 Spring Roo 开发环境	483
		12.1.1 问题	483

12.1.2 解决方案	483	13.3.3 工作原理	518
12.1.3 工作原理	483	13.4 管理集成测试中的应用上	
12.2 创建第一个 Spring Roo 项目	486	下文	520
12.2.1 问题	486	13.4.1 问题	520
12.2.2 解决方案	486	13.4.2 解决方案	520
12.2.3 工作原理	486	13.4.3 工作原理	521
12.3 把现有项目导入 SpringSource		13.5 向集成测试注入测试夹具	526
Tool Suite	491	13.5.1 问题	526
12.3.1 问题	491	13.5.2 解决方案	526
12.3.2 解决方案	492	13.5.3 工作原理	527
12.3.3 工作原理	492	13.6 管理集成测试中的事务	530
12.4 更快地构建更好的应用程序	493	13.6.1 问题	530
12.4.1 问题	493	13.6.2 解决方案	530
12.4.2 解决方案	494	13.6.3 工作原理	531
12.4.3 工作原理	494	13.7 在集成测试中访问数据库	536
12.5 从项目中删除 Spring Roo	500	13.7.1 问题	536
12.5.1 问题	500	13.7.2 解决方案	536
12.5.2 解决方案	500	13.7.3 工作原理	537
12.5.3 工作原理	501	13.8 使用 Spring 的常用测试	
12.6 小结	502	注解	540
第 13 章 Spring 测试	503	13.8.1 问题	540
13.1 用 JUnit and TestNG		13.8.2 解决方案	540
创建测试	504	13.8.3 工作原理	541
13.1.1 问题	504	13.9 小结	542
13.1.2 解决方案	504	第 14 章 Spring Portlet MVC 框架	544
13.1.3 工作原理	504	14.1 用 Spring Portlet MVC 开发一个	
13.2 创建单元测试和集成测试	509	简单的 Portlet	544
13.2.1 问题	509	14.1.1 问题	544
13.2.2 解决方案	509	14.1.2 解决方案	545
13.2.3 工作原理	510	14.1.3 工作原理	546
13.3 Spring MVC 控制器的单元		14.2 将 Portlet 请求映射到	
测试	518	处理程序	553
13.3.1 问题	518	14.2.1 问题	553
13.3.2 解决方案	518	14.2.2 解决方案	553

14.2.3 工作原理	554	15.6 在 JDBC 模板中使用命名 参数	595
14.3 用简单的表单控制器处理 portlet 表单	561	15.6.1 问题	595
14.3.1 问题	561	15.6.2 解决方案	595
14.3.2 解决方案	561	15.6.3 工作原理	595
14.3.3 工作原理	561	15.7 在 Spring JDBC 框架中 处理异常	597
14.4 小结	569	15.7.1 问题	597
第 15 章 数据访问	570	15.7.2 解决方案	597
15.1 Direct JDBC 的问题	571	15.7.3 工作原理	598
15.1.1 建立应用数据库	571	15.8 直接使用 ORM 框架的问题	602
15.1.2 理解数据访问对象 设计模式	573	15.8.1 问题	602
15.1.3 用 JDBC 实现 DAO	573	15.8.2 解决方案	603
15.1.4 在 Spring 中配置数据源	575	15.8.3 工作原理	603
15.1.5 运行 DAO	577	15.8.4 使用 Hibernate API, 用 Hibernate XML 映射 持续化对象	604
15.1.6 更进一步	577	15.8.5 使用 Hibernate API, 以 JPA 注解持续化对象	608
15.2 使用 JDBC 模板更新 数据库	578	15.8.6 使用 JPA, 以 Hibernate 为 引擎持续化对象	610
15.2.1 问题	578	15.9 在 Spring 中配置 ORM 资源 工厂	613
15.2.2 解决方案	578	15.9.1 问题	613
15.2.3 工作原理	578	15.9.2 解决方案	614
15.3 使用 JDBC 模板查询数据库	583	15.9.3 工作原理	614
15.3.1 问题	583	15.10 用 Spring ORM 模板持续化 对象	620
15.3.2 解决方案	583	15.10.1 问题	620
15.3.3 工作原理	583	15.10.2 解决方案	620
15.4 简化 JDBC 模板创建	588	15.10.3 工作原理	621
15.4.1 问题	588	15.11 用 Hibernate 的上下文会话持 续化对象	626
15.4.2 解决方案	588	15.11.1 问题	626
15.4.3 工作原理	589	15.11.2 解决方案	626
15.5 在 Java 1.5 中使用简单的 JDBC 模板	591		
15.5.1 问题	591		
15.5.2 解决方案	591		
15.5.3 工作原理	591		

15.11.3 工作原理.....	626	16.7.2 解决方案.....	652
15.12 用 JPA 的上下文注入持 续化对象.....	629	16.7.3 工作原理.....	653
15.12.1 问题.....	629	16.8 设置隔离事务属性.....	657
15.12.2 解决方案.....	629	16.8.1 问题.....	657
15.12.3 工作原理.....	630	16.8.2 解决方案.....	657
15.13 小结.....	632	16.8.3 工作原理.....	658
第 16 章 Spring 中的事务管理.....	634	16.9 设置 Rollback 事务属性.....	664
16.1 事务管理的问题.....	635	16.9.1 问题.....	664
16.2 选择一个事务管理器实现.....	641	16.9.2 解决方案.....	664
16.2.1 问题.....	641	16.9.3 工作原理.....	664
16.2.2 解决方案.....	641	16.10 设置超时和只读事务属性.....	666
16.2.3 工作原理.....	641	16.10.1 问题.....	666
16.3 用事务管理器 API 编程 管理事务.....	642	16.10.2 解决方案.....	666
16.3.1 问题.....	642	16.10.3 工作原理.....	666
16.3.2 解决方案.....	643	16.11 用加载时织入管理事务.....	667
16.3.3 工作原理.....	643	16.11.1 问题.....	667
16.4 用事务模板编程管理事务.....	644	16.11.2 解决方案.....	667
16.4.1 问题.....	644	16.11.3 工作原理.....	667
16.4.2 解决方案.....	645	16.12 小结.....	671
16.4.3 工作原理.....	645	第 17 章 EJB、Spring Remoting 和 Web 服务.....	672
16.5 用事务通知声明式地 管理事务.....	647	17.1 通过 RMI 暴露和调用服务.....	672
16.5.1 问题.....	647	17.1.1 问题.....	672
16.5.2 解决方案.....	648	17.1.2 解决方案.....	673
16.5.3 工作原理.....	648	17.1.3 工作原理.....	673
16.6 用@Transactional 注解声明式地 管理事务.....	650	17.2 用 Spring 创建 EJB 2.x 组件.....	676
16.6.1 方法.....	650	17.2.1 问题.....	676
16.6.2 解决方案.....	650	17.2.2 解决方案.....	677
16.6.3 工作原理.....	650	17.2.3 工作原理.....	677
16.7 设置事务传播属性.....	652	17.3 在 Spring 中访问遗留的 EJB 2.x 组件.....	683
16.7.1 问题.....	652	17.3.1 问题.....	683
		17.3.2 解决方案.....	683