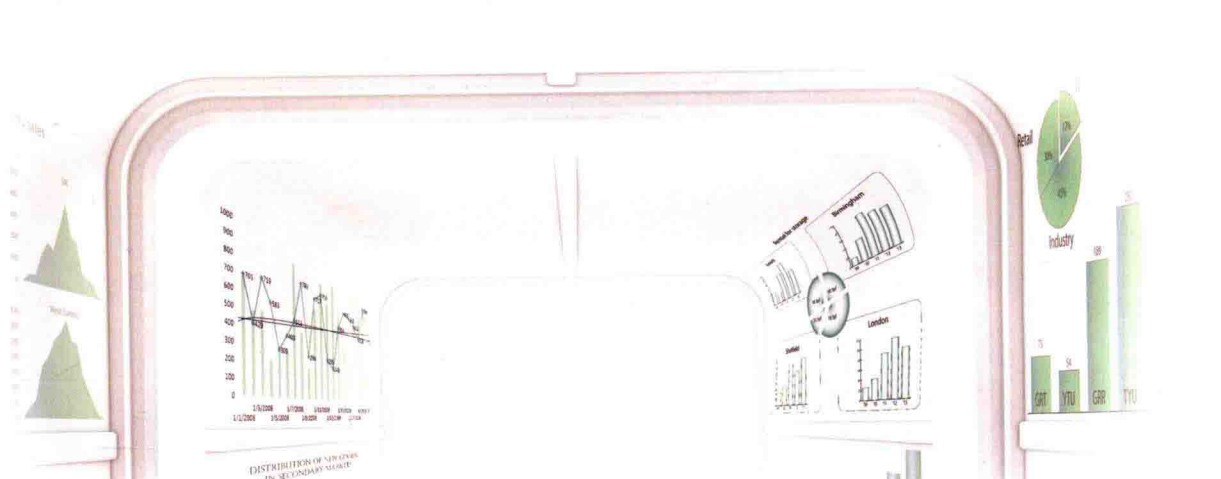




精通Spring 4.x 企业应用开发实战

Proficient in 4.x Spring
Enterprise Application Development

陈雄华 林开雄 文建国◎编著

精通 Spring 4.x

企业应用开发实战

陈雄华 林开雄 文建国◎编著

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

Spring 4.0 是 Spring 在积蓄 4 年后,隆重推出的一个重大升级版本,进一步加强了 Spring 作为 Java 领域第一开源平台的翘楚地位。

Spring 4.0 引入了众多 Java 开发者翘首以盼的基于 Groovy Bean 的配置、HTML 5/WebSocket 支持等新功能,全面支持 Java 8.0,最低要求是 Java 6.0。这些新功能实用性强、易用性高,可大幅降低 Java 应用,特别是 Java Web 应用开发的难度,同时有效提升应用开发的优雅性。

本书是在《精通 Spring 3.x——企业应用开发详解》的基础上,历时一年的重大调整改版而成的,延续了上一版本“追求深度,注重原理,不停留在技术表面”的写作风格,力求使读者在熟练使用 Spring 的各项功能的同时透彻理解 Spring 的内部实现,真正做到知其然并知其所以然。此外,本书重点突出了“实战性”的主题,力求使全书内容体现“从实际项目中来,到实际项目中去”的写作原则。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,侵权必究。

图书在版编目(CIP)数据

精通 Spring 4.x: 企业应用开发实战 / 陈雄华, 林开雄, 文建国编著. —北京: 电子工业出版社, 2017.1

ISBN 978-7-121-30443-9

I. ①精… II. ①陈… ②林… ③文… III. ①JAVA 语言—程序设计 IV. ①TP312.8

中国版本图书馆 CIP 数据核字(2016)第 284564 号

责任编辑: 李 冰

特约编辑: 田学清 赵海军等

印 刷: 三河市华成印务有限公司

装 订: 三河市华成印务有限公司

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱

邮编: 100036

开 本: 787×1092 1/16 印张: 51.25 字数: 1312 千字

版 次: 2017 年 1 月第 1 版

印 次: 2017 年 1 月第 1 次印刷

印 数: 3000 册 定价: 128.00 元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888, 88258888。

质量投诉请发邮件至 zltz@phei.com.cn, 盗版侵权举报请发邮件到 dbqq@phei.com.cn。

本书咨询联系方式: libing@phei.com.cn。

前 言

本书小序

Spring 从 2004 年发布第一个版本以来，至今已有 12 载。12 年刚好是一个生肖轮回，但在一日千里的计算机领域，12 年基本上算是一个世纪了。都说“好花不常开，好景不常在”，但 Spring 这朵 Java 开源世界里芳香馥郁的奇葩不但没有零落成泥，反而满园春色历久弥艳，成为 Java 开发者无法回避的开源框架。

回顾 Spring 的光辉岁月，一路与时俱进，引领时代之潮流。总的来说，Spring 主要经历了三次重大的版本升级：一为 2006 年从 1.0 升级到 2.0，在 Spring 2.0 中新增了 XML 命名空间、AspectJ 及 Spring MVC 等功能，此外，在 Spring 2.5 中还引入了注解驱动配置的支持，同时进一步完善了 Spring MVC 功能；二为 2009 年从 2.5 升级到 3.0，新增了 SpEL、OXM、REST、验证/格式化等功能，全面支持 Java 5.0；三为 2013 年从 3.0 升级到 4.0，新增了 Groovy Bean 配置、HTML 5/WebSocket 支持等功能，全面支持 Java 8.0，最低要求是 Java 6.0。Spring 始终坚持以小版本快速推进、每三年左右发布一个大版本的演化策略，既保证版本的平稳有序，又能紧跟技术发展的潮流。难能可贵的是，Spring 即便发生了这么多次版本的升级，其整体框架依然是向下兼容的，在这一点上，Spring 明显区别于 Struts、Hibernate 等框架的升级风格。

笔者在 2007 年曾编写了《精通 Spring 2.x》，并在 2012 年出版了升级版本《精通 Spring 3.x》。感谢读者朋友的厚爱垂青，其中《精通 Spring 3.x》已经重印了 11 次，成为国内 Spring 领域的畅销书籍。2013 年年底 Spring 4.0 就已经发布了，从那时起，出版社的朋友多次力促笔者进行版本的同步升级，笔者也希望能与时俱进地对原书进行更新，但囿于工作繁忙，一直未能付诸行动。直到 2015 年 8 月左右，才与林开雄、文建国着手筹划《精通 Spring 4.x》的升级编写工作。林开雄和文建国是笔者多年的好朋友，二人都是拥有十多年工作经验的 Java 实战型技术高手，他们不但为人谦和、技术精湛，而且拥有丰厚的创作实力。林开雄早在 2012 年就参与了《精通 Spring 3.x》的撰写工作，文建国则于 2014 年翻译了《Spring Data 实战》。此外，林莉、何彩云、陈谋坤、项群、陈文炎、陈曦、康玉琳、蔡雪峰、康沿清、朱景、朱贤俊等也一起参与了本书的代码审查及测试工作，在此对大家的努力付出一并表示感谢！

本次改版，不但将全书内容同步更新到 Spring 4.0，还对全书结构进行了多方面的优化和调整。移除了两个章节，分别是“JavaMail 发送邮件”（考虑到比较简单）及“在 Spring 中开发 Web Service”（考虑到 Spring MVC 开发 REST WebService 再合适不过了）；新增了 3 个全新的章节，分别是“Spring Boot”、“Spring SpEL”及“Spring Cache”。

本书特点

- ❑ **揭示内幕，深入浅出：**笔者对 Spring 的源码进行了彻底分析，深刻揭示了 Spring 框架的技术内幕，让读者知其然，更知其所以然。Spring 中的许多设计经验、技巧、模式具有很高的借鉴性，在透彻学习 Spring 体系结构的同时，读者可以直接将这些方法借用到具体的应用开发中。
- ❑ **同步更新，与时俱进：**虽然在 2013 年 12 月就发布 Spring 4.0 的第一个候选版本，后来又发布了多个 RC 版本，并最终于 2015 年 8 月发布了 Spring 4.2 的正式版本，但新功能的添加及旧功能的调整从来就没有停止过。本书基于 Spring 4.2 版本讲解，保证全书内容与时俱进。
- ❑ **突出重点，淡化边缘：**虽然全书篇幅达 800 多页，但本书没有片面追求内容的面面俱到，相反，我们特别注意内容的剪裁和取舍；对于实用性强的知识点深入分析、深度挖掘，而对于不常用的知识点则点到为止，甚至不纳入本书的范围。举例来说，我们对使用 Spring JDBC、Spring Cache 及 Spring MVC 等实用性强的技术都进行了深入分析，而对如何集成 EJB、JMX、JCA 等不常用的功能完全不涉及，很好地做到了实用性和深入性的统一。
- ❑ **理论透彻，面向实践：**本书在透彻分析原理、讲解技术知识点的同时，特别注意与实际应用的结合。笔者将自身丰富的实战经验糅合到全书的相关知识点中，很好地做到了知识讲解和实战经验的结合，让读者在掌握纯技术知识的同时能够对如何活用技术做到胸有成竹。如笔者在第 16 章讲解任务调度的内容时，专门辟出 16.6 节讲解实际应用中任务调度的使用经验；在第 18 章中讲述使用实战项目开发时，专门通过 18.11 节讲述了笔者在实际项目中所总结的项目配置文件及数据源的规划方案。此外，我们还适时提供了“实战经验”的插文，它们在不影响上下文连贯性的同时让读者学到了相关技术的实战经验。诸如此类的以实际应用为导向的内容贯穿全书，这是本书区别于其他书籍的特色之一。
- ❑ **代码简洁，图例丰富：**全书代码在排版布局及内容剪裁上颇费心思，实例代码重点关注当前知识点涉及的内容，弱化边缘代码，并采用特殊的排版方式适时添加简明扼要的注释，方便程序代码的阅读和重点内容的把握。全书拥有大量精美的图表，这些图表很好地解构了上下文中的知识难点，大大提高了可读性，降低了理解的难度。
- ❑ **注重趣味，轻松阅读：**由于技术书籍的严谨性、知识性特点，阅读技术书籍往往是枯燥乏味的，更遑论趣味性。笔者对此深有感触，为寻求一些突破，我们

在全书大部分章节都精心设计了一个“轻松一刻”环节。它们和上下文内容存在某种程度的关联性，但其本身是一段趣味性的短文，以便在增强全书趣味性的同时还为读者提供了另一个思考问题的角度。

- ❑ **相关知识，一网打尽：**Spring 不但本身涉及众多 Java 技术，其集成的第三方技术本身也涵盖了丰富的知识。我们在介绍 Spring 相关技术时，都会简明扼要地讲解相关联的基础知识，其中包括 Java 的新知识和被集成技术的知识，而不是在完全脱离背景知识的情况下孤立讲解 Spring 的知识。
- ❑ **通力合作，倾力打造：**本书从筹划到改版完成，历时近 8 个月。我们交叉审核，多次优化重构，为保证全书质量，多次向出版社申请延迟提交稿件，直到 2016 年 6 月才完成所有稿件。

本书结构

本书分为 5 篇。第 1 篇为基础篇，包括第 1~3 章，讲解 Spring 概述性知识，以便读者快速建立对 Spring 的整体认识，能够使用 Spring 快速开发一个简单的项目，而更多深入的知识则在后续篇章中展开。第 2 篇为核心篇，包括第 4~9 章，讲解 Spring 的 IoC、AOP 及 SpEL 的知识，这些知识是 Spring 的核心，也是 Spring 所有衍生服务及功能的基石。第 3 篇为数据篇，包括第 10~14 章，讲解 Spring 的各种数据访问技术及事务管理的内容，对事务管理的实现机制和各种疑难问题进行剖析。第 4 篇为应用篇，包括第 15~18 章，讲解数据缓存、任务调度、Web 开发的内容，对于企业应用开发来说，数据缓存及任务调度是两个无法回避的问题，需要重点学习和掌握；此外，本篇还精心设计了一个实战案例，包含需求分析、数据库设计、项目开发、代码测试、应用部署的整体过程，让读者在项目的实战中整体串接 Spring 的知识点。第 5 篇为提高篇，包括第 19 章和第 20 章，讲解 Spring OXM 及单元测试的内容；全书工程代码放在本书配套网盘中，请读者下载网盘内容阅读，本书网盘下载地址如下。

百度云盘：<http://pan.baidu.com/s/1boCl3d1>。

下面简要介绍一下每章的内容。

第 1 章：对 Spring 框架进行了整体性的概述，可使读者快速建立起对 Spring 整体性的认识。

第 2 章：通过一个简单的例子展现开发 Spring Web 应用的整体过程，通过这个实例，读者可以快速跨入 Spring Web 应用的世界。

第 3 章：Spring Boot 的设计目的是用来简化新 Spring 应用的搭建和开发过程，本章通过实例向读者讲述了 Spring Boot 的使用技巧。

第 4 章：讲解了 Spring IoC 容器的知识，通过具体的实例详细地讲解了 IoC 概念；同时，对 Spring 框架的三个最重要的框架级接口进行了剖析，并对 Bean 的生命周期进行了讲解。

第 5 章：讲解了如何在 Spring 配置文件中使⽤各种配置⽅式配置 Bean 的内容，并对各个配置项的意义进⾏了深⼊说明。

第 6 章：对 Spring 容器进⾏了解构，从内部探究 Spring 容器的体系结构和运⾏流程。此外，还对 Spring 容器⼀些⾼级主题进⾏了深⼊阐述。

第 7 章：从 Spring AOP 的底层实现技术⼊手，⼀步步深⼊到 Spring AOP 的内核中，分析它的底层结构和具体实现。

第 8 章：对如何使⽤基于 AspectJ 配置 AOP 的知识进⾏了深⼊分析，包括使⽤ XML Schema 配置文件、使⽤注解进⾏配置等内容。

第 9 章：SpEL 不仅仅是一个动态语⾔，⽽且 Spring 容器的很多配置都直接依赖于 SpEL ⼯作，因此掌握 SpEL 是掌握 Spring 配置的必修课程。

第 10 章：介绍了 Spring 所提供的 DAO 封装层，包括 Spring DAO 的异常体系、数据访问模板等内容。

第 11 章：声明式事务配置是 Spring 的⼀项重要功能，本章介绍了 Spring 事务管理的工作机制，以及通过 XML、注解等⽅式进⾏事务管理配置等内容。

第 12 章：对实际应⽤中 Spring 事务管理的各种疑难问题进⾏了透彻剖析，让读者对 Spring 事务管理不再有云遮雾罩的感觉。

第 13 章：讲解了如何使⽤ Spring JDBC 进⾏数据访问操作，还重点讲述了 LOB 字段处理、主键的产⽣和获取等难点知识。

第 14 章：讲解了如何在 Spring 中集成 Hibernate、MyBatis 等数据访问框架，同时，读者还将学到 ORM 框架的混⽤和 DAO 层设计的知识。

第 15 章：数据缓存已经成为提⾼系统运⾏性能的⼀个重要⽅法，本章讲解了 Spring Cache 如何通过注解⽅式进⾏透明化数据缓存。

第 16 章：本章重点讲解了在 Spring 中如何使⽤ Quartz 进⾏任务调度，同时还涉及使⽤ JDK Timer 和 JDK 5.0 执⾏器等知识。

第 17 章：对 Spring MVC 框架进⾏了详细介绍，对 REST 风格的编程⽅式进⾏了重点讲解，同时还对 Spring 的校验和格式化框架如何与 Spring MVC 整合进⾏了说明。

第 18 章：以⼀个实际的项⽬为蓝本，带领读者从项⽬需求分析、项⽬设计、代码开发、单元测试直到应⽤部署体验⼀次接近实战的整体项⽬开发过程。

第 19 章：介绍 Spring OXM 的多种实现技术，同时对 XML 技术进⾏了整体说明。

第 20 章：有别于⼀般书籍的单元测试内容，本书以当前最具实战性的 TestNG+Unitils+Mockito 复合测试框架对测试基于数据库的 Web 应⽤进⾏了深⼊讲解。

如何使用本书

由于程序开发是⼀项实践性极强的⼯作，只有亲⾝体验才能掌握其真谛，因此我们强烈建议读者使⽤本书网盘的工程代码进⾏同步学习。本书项⽬基于 Java 7.0+，请

读者先安装好 Java SDK 7.0+; 开发工具使用 IntelliJ IDEA, 可从 <http://www.jetbrains.com/idea> 下载免费社区版进行安装。请下载本书配套网盘的内容, 建议下载到本地的 D:\masterspring 目录下。本书所有章节对应的工程项目均以 Maven 方式组织, 请将网盘 tools\settings.xml 复制到 C:\Users\<操作系统用户名>\.m2 目录下。在 settings.xml 中, 我们使用了国内的 oschina Maven 公共仓库, 下载依赖构件包速度很快; 否则, Maven 项目将默认从国外的中央仓库下载, 速度很慢。

当学习到某个章节时, 请在 IDEA 中打开对应章节的工程项目, 并依照下面的步骤创建章节所对应的 IDEA 工程项目:

(1) 依次选择 File→New→Project From Existing Sources...命令, 打开 Select File or Directory to Import 对话框。

(2) 选择要打开的 D:\masterspring\chapterX 项目工程, 打开 Import Project 向导。

(3) 选择 Import project from external model 选项并在列表中选择 Maven, 然后连续单击 Next 按钮, 按向导指示创建章节的 Maven 项目工程 (注意 SDK 选择 1.7)。

(4) 由于本书工程项目采用 UTF-8 编码, 所以在创建完项目工程后, 请按 **Ctrl+Alt+S** 组合键打开 Setting 对话框, 找到 Editor→File Encodings, 将 IDE Encoding、Project Encoding 及 Default encoding for properties files 三项都设置为 UTF-8; 否则将会因编码问题导致编译失败。

本书很多章节都用到了数据库, 因此请在本机安装 MySQL 5.0+, 本书假设 MySQL 的 root 密码为 123456。在每个需要数据库访问的项目工程中都拥有一个 schema 目录, 其下是数据库初始化 SQL 脚本, 请先执行脚本再运行章节对应的代码。

本书插文

本书会适时加入一些提示、实战经验和轻松一刻的小段插文, 在不打断行文连续性的同时提供一些有益的开发经验、使用技巧并增强阅读的趣味性。这些插文都带有一个小图标加以凸显, 说明如下。

	提示：在上下文中可能存在一些读者容易忽视或容易犯错的地方, 在提示信息中给予针对性的帮助信息
	实战经验：笔者将多年的开发实战经验适时介绍给大家。这些知识往往是不能从一般的书籍或资料中获得的。本书会适时地在行文中将这些实战经验分享出来, 相信可以使读者少走一些弯路
	轻松一刻：为了增强技术书籍阅读的趣味性, 全书每章都有一到两个“轻松一刻”的短文。它们和上下文内容都存在某种程度的关联性, 不但可为阅读带来了趣味性, 还可以启发读者思考

此外，由于 Spring 4.x 拥有多个版本，为了保持行文的简洁，除非特别指出，本书的 Spring 或 Spring 4.0 即代表当前的最新版本（Spring 4.2.x）。

如何与作者联系

由于 Spring 内容涵盖面宽广，涉及的内容非常多，同时由于作者水平有限，错误之处在所难免。我们不但欢迎读者朋友来信交流，更期待各界高手、专家就不足之处给予赐教和斧正，您可以通过 itstamen@qq.com 与笔者联系。

陈雄华

2016 年 5 月 22 日于厦门

目 录

第 1 篇 基础篇

第 1 章 Spring 概述 2

- 1.1 认识 Spring 2
- 1.2 关于 SpringSource 4
- 1.3 Spring 带给我们什么 5
- 1.4 Spring 体系结构 6
- 1.5 Spring 对 Java 版本的要求 8
- 1.6 Spring 4.0 新特性 8
 - 1.6.1 全面支持 Java 8.0 9
 - 1.6.2 核心容器的增强 11
 - 1.6.3 支持用 Groovy 定义 Bean 12
 - 1.6.4 Web 的增强 12
 - 1.6.5 支持 WebSocket 12
 - 1.6.6 测试的增强 13
 - 1.6.7 其他 13
- 1.7 Spring 子项目 13
- 1.8 如何获取 Spring 15
- 1.9 小结 16

第 2 章 快速入门 17

- 2.1 实例概述 17
 - 2.1.1 比 Hello World 更适用的实例 ... 18
 - 2.1.2 实例功能简介 18

2.2 环境准备 20

- 2.2.1 构建工具 Maven 20
- 2.2.2 创建库表 22
- 2.2.3 建立工程 23
- 2.2.4 类包及 Spring 配置文件规划 28

2.3 持久层 29

- 2.3.1 建立领域对象 29
- 2.3.2 UserDao 30
- 2.3.3 LoginLogDao 33
- 2.3.4 在 Spring 中装配 DAO 34

2.4 业务层 35

- 2.4.1 UserService 35
- 2.4.2 在 Spring 中装配 Service 37
- 2.4.3 单元测试 38

2.5 展现层 40

- 2.5.1 配置 Spring MVC 框架 40
- 2.5.2 处理登录请求 42
- 2.5.3 JSP 视图页面 44

2.6 运行 Web 应用 46

2.7 小结 48

第 3 章 Spring Boot 49

- 3.1 Spring Boot 概览 49
 - 3.1.1 Spring Boot 发展背景 50

- 3.1.2 Spring Boot 特点50
- 3.1.3 Spring Boot 启动器50
- 3.2 快速入门 52
- 3.3 安装配置 54
 - 3.3.1 基于 Maven 环境配置54
 - 3.3.2 基于 Gradle 环境配置56
 - 3.3.3 基于 Spring Boot CLI 环境配置57
 - 3.3.4 代码包结构规划58
- 3.4 持久层 59
 - 3.4.1 初始化配置59
 - 3.4.2 UserDao61
- 3.5 业务层 62
- 3.6 展现层 64
 - 3.6.1 配置 pom.xml 依赖64
 - 3.6.2 配置 Spring MVC 框架65
 - 3.6.3 处理登录请求65
- 3.7 运维支持 67
- 3.8 小结 70

第 2 篇 核心篇

- 第 4 章 IoC 容器72
 - 4.1 IoC 概述 72
 - 4.1.1 通过实例理解 IoC 的概念73
 - 4.1.2 IoC 的类型75
 - 4.1.3 通过容器完成依赖关系的注入 77
 - 4.2 相关 Java 基础知识 78
 - 4.2.1 简单实例78
 - 4.2.2 类装载器 ClassLoader80
 - 4.2.3 Java 反射机制83

- 4.3 资源访问利器 85
 - 4.3.1 资源抽象接口85
 - 4.3.2 资源加载88
- 4.4 BeanFactory 和 ApplicationContext ... 91
 - 4.4.1 BeanFactory 介绍92
 - 4.4.2 ApplicationContext 介绍94
 - 4.4.3 父子容器103
- 4.5 Bean 的生命周期 103
 - 4.5.1 BeanFactory 中 Bean 的生命周期103
 - 4.5.2 ApplicationContext 中 Bean 的生命周期112
- 4.6 小结114

第 5 章 在 IoC 容器中装配 Bean115

- 5.1 Spring 配置概述116
 - 5.1.1 Spring 容器高层视图116
 - 5.1.2 基于 XML 的配置117
- 5.2 Bean 基本配置 120
 - 5.2.1 装配一个 Bean120
 - 5.2.2 Bean 的命名120
- 5.3 依赖注入 121
 - 5.3.1 属性注入121
 - 5.3.2 构造函数注入124
 - 5.3.3 工厂方法注入128
 - 5.3.4 选择注入方式的考量130
- 5.4 注入参数详解 130
 - 5.4.1 字面值130
 - 5.4.2 引用其他 Bean131
 - 5.4.3 内部 Bean133
 - 5.4.4 null 值133
 - 5.4.5 级联属性134

5.4.6 集合类型属性.....	134	5.12.2 使用 GenericGroovyApplication Context 启动 Spring 容器.....	171
5.4.7 简化配置方式.....	138	5.13 通过编码方式动态添加 Bean.....	172
5.4.8 自动装配.....	141	5.13.1 通过 DefaultListableBean Factory.....	172
5.5 方法注入.....	142	5.13.2 扩展自定义标签.....	173
5.5.1 lookup 方法注入.....	142	5.14 不同配置方式比较.....	175
5.5.2 方法替换.....	143	5.15 小结.....	177
5.6 <bean>之间的关系.....	144	第 6 章 Spring 容器高级主题.....	178
5.6.1 继承.....	144	6.1 Spring 容器技术内幕.....	178
5.6.2 依赖.....	145	6.1.1 内部工作机制.....	179
5.6.3 引用.....	146	6.1.2 BeanDefinition.....	182
5.7 整合多个配置文件.....	147	6.1.3 InstantiationStrategy.....	183
5.8 Bean 作用域.....	148	6.1.4 BeanWrapper.....	183
5.8.1 singleton 作用域.....	148	6.2 属性编辑器.....	184
5.8.2 prototype 作用域.....	149	6.2.1 JavaBean 的编辑器.....	185
5.8.3 与 Web 应用环境相关的 Bean 作用域.....	150	6.2.2 Spring 默认属性编辑器.....	188
5.8.4 作用域依赖问题.....	152	6.2.3 自定义属性编辑器.....	189
5.9 FactoryBean.....	153	6.3 使用外部属性文件.....	192
5.10 基于注解的配置.....	155	6.3.1 PropertyPlaceholderConfigurer 属性文件.....	192
5.10.1 使用注解定义 Bean.....	155	6.3.2 使用加密的属性文件.....	195
5.10.2 扫描注解定义的 Bean.....	156	6.3.3 属性文件自身的引用.....	198
5.10.3 自动装配 Bean.....	157	6.4 引用 Bean 的属性值.....	199
5.10.4 Bean 作用范围及生命过程 方法.....	162	6.5 国际化信息.....	201
5.11 基于 Java 类的配置.....	164	6.5.1 基础知识.....	201
5.11.1 使用 Java 类提供 Bean 定义 信息.....	164	6.5.2 MessageSource.....	206
5.11.2 使用基于 Java 类的配置信息 启动 Spring 容器.....	167	6.5.3 容器级的国际化信息资源.....	209
5.12 基于 Groovy DSL 的配置.....	169	6.6 容器事件.....	210
5.12.1 使用 Groovy DSL 提供 Bean 定义信息.....	169	6.6.1 Spring 事件类结构.....	211
		6.6.2 解构 Spring 事件体系的具体 实现.....	213

6.6.3 一个实例	214	7.5.2 BeanNameAutoProxyCreator.....	260
6.7 小结.....	215	7.5.3 DefaultAdvisorAutoProxy Creator.....	261
第7章 Spring AOP 基础.....	216	7.5.4 AOP 无法增强疑难问题 剖析.....	262
7.1 AOP 概述.....	216	7.6 小结.....	267
7.1.1 AOP 到底是什么.....	217	第8章 基于@AspectJ 和 Schema 的 AOP	269
7.1.2 AOP 术语	219	8.1 Spring 对 AOP 的支持	269
7.1.3 AOP 的实现者.....	221	8.2 Java 5.0 注解知识快速进阶	270
7.2 基础知识.....	222	8.2.1 了解注解.....	270
7.2.1 带有横切逻辑的实例.....	222	8.2.2 一个简单的注解类.....	271
7.2.2 JDK 动态代理	224	8.2.3 使用注解.....	272
7.2.3 CGLib 动态代理	228	8.2.4 访问注解.....	273
7.2.4 AOP 联盟	229	8.3 着手使用@AspectJ.....	274
7.2.5 代理知识小结.....	230	8.3.1 使用前的准备.....	275
7.3 创建增强类.....	230	8.3.2 一个简单的例子.....	275
7.3.1 增强类型	230	8.3.3 如何通过配置使用@AspectJ 切面.....	277
7.3.2 前置增强	231	8.4 @AspectJ 语法基础.....	278
7.3.3 后置增强	235	8.4.1 切点表达式函数.....	278
7.3.4 环绕增强	236	8.4.2 在函数入参中使用通配符	279
7.3.5 异常抛出增强.....	237	8.4.3 逻辑运算符.....	280
7.3.6 引介增强	239	8.4.4 不同增强类型.....	281
7.4 创建切面.....	243	8.4.5 引介增强用法.....	282
7.4.1 切点类型	243	8.5 切点函数详解.....	283
7.4.2 切面类型	244	8.5.1 @annotation().....	284
7.4.3 静态普通方法名匹配切面	246	8.5.2 execution().....	285
7.4.4 静态正则表达式方法匹配 切面	248	8.5.3 args()和@args().....	287
7.4.5 动态切面	251	8.5.4 within().....	288
7.4.6 流程切面	254	8.5.5 @within()和@target().....	289
7.4.7 复合切点切面.....	256	8.5.6 target()和 this().....	290
7.4.8 引介切面	258		
7.5 自动创建代理.....	259		
7.5.1 实现类介绍	259		

第 11 章 Spring 的事务管理.....	355	11.7.2 WebSphere.....	390
11.1 数据库事务基础知识	355	11.8 小结.....	390
11.1.1 何为数据库事务.....	356	第 12 章 Spring 的事务管理难点剖析...	392
11.1.2 数据并发的问题.....	357	12.1 DAO 和事务管理的牵绊	393
11.1.3 数据库锁机制.....	359	12.1.1 JDBC 访问数据库	393
11.1.4 事务隔离级别.....	360	12.1.2 Hibernate 访问数据库	395
11.1.5 JDBC 对事务的支持	361	12.2 应用分层的迷惑.....	398
11.2 ThreadLocal 基础知识.....	362	12.3 事务方法嵌套调用的迷茫	401
11.2.1 ThreadLocal 是什么	363	12.3.1 Spring 事务传播机制回顾	401
11.2.2 ThreadLocal 的接口方法	363	12.3.2 相互嵌套的服务方法	402
11.2.3 一个 ThreadLocal 实例.....	364	12.4 多线程的困惑.....	405
11.2.4 与 Thread 同步机制的比较.....	366	12.4.1 Spring 通过单实例化 Bean 简化多线程问题.....	405
11.2.5 Spring 使用 ThreadLocal 解决 线程安全问题.....	366	12.4.2 启动独立线程调用事务 方法.....	405
11.3 Spring 对事务管理的支持.....	368	12.5 联合军种作战的混乱	408
11.3.1 事务管理关键抽象.....	369	12.5.1 Spring 事务管理器的应对	408
11.3.2 Spring 的事务管理器实现类	371	12.5.2 Hibernate+Spring JDBC 混合框架的事务管理	408
11.3.3 事务同步管理器.....	374	12.6 特殊方法成漏网之鱼	412
11.3.4 事务传播行为	375	12.6.1 哪些方法不能实施 Spring AOP 事务.....	412
11.4 编程式的事务管理	376	12.6.2 事务增强遗漏实例	413
11.5 使用 XML 配置声明式事务	377	12.7 数据连接泄露.....	416
11.5.1 一个将被实施事务增强的 服务接口	379	12.7.1 底层连接资源的访问问题	416
11.5.2 使用原始的 TransactionProxy FactoryBean	379	12.7.2 Spring JDBC 数据连接泄露.....	417
11.5.3 基于 aop/tx 命名空间的配置	382	12.7.3 事务环境下通过 DataSource Utils 获取数据连接	420
11.6 使用注解配置声明式事务	385	12.7.4 无事务环境下通过 DataSource Utils 获取数据连接.....	422
11.6.1 使用@Transactional 注解.....	385	12.7.5 JdbcTemplate 如何做到对连接 泄露的免疫.....	424
11.6.2 通过 AspectJ LTW 引入事务 切面	389		
11.7 集成特定的应用服务器	390		
11.7.1 BEA WebLogic	390		

12.7.6 使用 TransactionAwareData SourceProxy	425	第 14 章 整合其他 ORM 框架	460
12.7.7 其他数据访问技术的等价类	426	14.1 Spring 整合 ORM 技术	460
12.8 小结	426	14.2 在 Spring 中使用 Hibernate	462
第 13 章 使用 Spring JDBC 访问 数据库	428	14.2.1 配置 SessionFactory	462
13.1 使用 Spring JDBC	428	14.2.2 使用 HibernateTemplate	465
13.1.1 JdbcTemplate 小试牛刀	429	14.2.3 处理 LOB 类型的数据	469
13.1.2 在 DAO 中使用 Jdbc Template	429	14.2.4 添加 Hibernate 事件监听器	470
13.2 基本的数据操作	431	14.2.5 使用原生的 Hibernate API	471
13.2.1 更改数据	431	14.2.6 使用注解配置	472
13.2.2 返回数据库的表自增主键值	434	14.2.7 事务处理	474
13.2.3 批量更改数据	436	14.2.8 延迟加载问题	475
13.2.4 查询数据	437	14.3 在 Spring 中使用 MyBatis	476
13.2.5 查询单值数据	440	14.3.1 配置 SqlMapClient	476
13.2.6 调用存储过程	442	14.3.2 在 Spring 中配置 MyBatis	478
13.3 BLOB/CLOB 类型数据的操作	444	14.3.3 编写 MyBatis 的 DAO	479
13.3.1 如何获取本地数据连接	445	14.4 DAO 层设计	482
13.3.2 相关的操作接口	446	14.4.1 DAO 基类设计	482
13.3.3 插入 LOB 类型的数据	448	14.4.2 查询接口方法设计	484
13.3.4 以块数据方式读取 LOB 数据	450	14.4.3 分页查询接口设计	486
13.3.5 以流数据方式读取 LOB 数据	451	14.5 小结	487
13.4 自增键和行集	452		
13.4.1 自增键的使用	452	第 4 篇 应用篇	
13.4.2 如何规划主键方案	454	第 15 章 Spring Cache	490
13.4.3 以行集返回数据	456	15.1 缓存概述	490
13.5 NamedParameterJdbcTemplate 模板类	456	15.1.1 缓存的概念	490
13.6 小结	459	15.1.2 使用 Spring Cache	493
		15.2 掌握 Spring Cache 抽象	498
		15.2.1 缓存注解	498
		15.2.2 缓存管理器	504
		15.2.3 使用 SpEL 表达式	506
		15.2.4 基于 XML 的 Cache 声明	506

15.2.5	以编程方式初始化缓存.....	507	16.6.2	任务调度对应用程序集群的 影响.....	547
15.2.6	自定义缓存注解.....	509	16.6.3	任务调度云.....	547
15.3	配置 Cache 存储.....	509	16.6.4	Web 应用程序中调度器的 启动和关闭问题.....	549
15.3.1	EhCache.....	510	16.7	小结.....	552
15.3.2	Guava.....	510	第 17 章	Spring MVC.....	553
15.3.3	HazelCast.....	511	17.1	Spring MVC 体系概述.....	554
15.3.4	GemFire.....	511	17.1.1	体系结构.....	554
15.3.5	JSR-107 Cache.....	511	17.1.2	配置 DispatcherServlet.....	555
15.4	实战经验.....	513	17.1.3	一个简单的实例.....	560
15.5	小结.....	514	17.2	注解驱动的控制​​器.....	565
第 16 章	任务调度和异步执行器.....	516	17.2.1	使用 @RequestMapping 映射请求.....	565
16.1	任务调度概述.....	516	17.2.2	请求处理方法签名.....	569
16.2	Quartz 快速进阶.....	517	17.2.3	使用矩阵变量绑定参数.....	570
16.2.1	Quartz 基础结构.....	518	17.2.4	请求处理方法签名详细说明.....	571
16.2.2	使用 SimpleTrigger.....	520	17.2.5	使用 HttpMessageConverter <T>.....	575
16.2.3	使用 CronTrigger.....	522	17.2.6	使用 @RestController 和 AsyncRestTemplate.....	584
16.2.4	使用 Calendar.....	526	17.2.7	处理模型数据.....	586
16.2.5	任务调度信息存储.....	527	17.3	处理方法的数据绑定.....	591
16.3	在 Spring 中使用 Quartz.....	530	17.3.1	数据绑定流程剖析.....	592
16.3.1	创建 JobDetail.....	530	17.3.2	数据转换.....	592
16.3.2	创建 Trigger.....	533	17.3.3	数据格式化.....	598
16.3.3	创建 Scheduler.....	534	17.3.4	数据校验.....	602
16.4	在 Spring 中使用 JDK Timer.....	536	17.4	视图和视图解析器.....	611
16.4.1	Timer 和 TimerTask.....	536	17.4.1	认识视图.....	611
16.4.2	Spring 对 Java Timer 的支持.....	539	17.4.2	认识视图解析器.....	612
16.5	Spring 对 Java 5.0 Executor 的 支持.....	540	17.4.3	JSP 和 JSTL.....	613
16.5.1	了解 Java 5.0 的 Executor.....	541	17.4.4	模板视图.....	618
16.5.2	Spring 对 Executor 所提供的 抽象.....	543			
16.6	实际应用中的任务调度.....	544			
16.6.1	如何产生任务.....	545			