

# Spring 学习指南

## (第4版)

[印度] J. 夏尔马 (J. Sharma) 阿西施·萨林 (Ashish Sarin) 著  
周密 译



# Spring 学习指南

## (第4版)

[印度] J. 夏尔马 (J. Sharma) 阿西施·萨林 (Ashish Sarin) 著  
周密 译



人民邮电出版社  
北京

## 图书在版编目(CIP)数据

Spring学习指南：第4版 / (印) J.夏尔马  
(J. Sharma), (印) 阿西施·萨林 (Ashish Sarin) 著 ;  
周密译. — 北京 : 人民邮电出版社, 2020.1  
ISBN 978-7-115-51929-0

I. ①S… II. ①J… ②阿… ③周… III. ①JAVA语言  
—程序设计 IV. ①TP312.8

中国版本图书馆CIP数据核字(2019)第189458号

## 版权声明

Getting Started with Spring Framework:Covers Spring 5, 4th Edition ISBN-13: 978-1979962780

Copyright ©2017 by Jyotsna Sharma and Ashish Sarin.

Simplified Chinese translation copyright ©2019 by Posts and Telecommunications Press Co..

Published by arrangement with Jyotsna Sharma.

All Rights Reserved.

本书中文简体版由 J. Sharma 和 Ashish Sarin 授权人民邮电出版社有限公司出版。未经出版者书面许可，  
对本书的任何部分不得以任何方式或任何手段复制和传播。  
版权所有，侵权必究。

---

◆ 著 [印度] J. 夏尔马 (J. Sharma) 阿西施·萨林(Ashish Sarin)  
译 周密  
责任编辑 吴晋瑜  
责任印制 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>  
大厂聚鑫印刷有限责任公司印刷

◆ 开本: 787×1092 1/16  
印张: 29.75  
字数: 954 千字 2020 年 1 月第 1 版  
印数: 1-2 400 册 2020 年 1 月河北第 1 次印刷

著作权合同登记号 图字: 01-2018-7368 号

---

定价: 89.00 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: 京东工商广登字 20170147 号

# 内 容 提 要

Spring 框架是以简化 J2EE 应用程序开发为特定目标而创建的，是当前最流行的 Java 开发框架。本书主要介绍 Beans 的配置、依赖注入、定制 bean、基于 Java 的容器、AOP、Spring Data、Spring MVC 等内容。本书基于非常适合构建 JavaWeb 微服务框架的 Spring 5 编写，涵盖 Java 的函数式编程、RxJava 2 的反应式编程、用 Spring WebFlux、Spring Data 和 Spring Security 开发反应式 RESTful Web 服务等内容。

本书适合 Web 开发者和想使用 Spring 的初学者参考，也可供对 Web 开发和 Spring 感兴趣的读者参考。

# 前 言

## 如何使用这本书

### 下载示例项目

这本书有许多示例项目，你可以从 GitHub 中找到相应的项目。你既可以将示例项目作为单个 ZIP 文件下载，也可以使用 Git 检出示例项目。

### 将示例项目导入你的 Eclipse IDE

如果你在阅读本书时发现有如 `IMPORT chapter<chapter-number>/<project name>` 这样的标识，应该将指定的项目导入你的 Eclipse IDE（或者使用其他的 IDE）。附录 B 介绍了导入和运行示例项目所需的步骤。

### 参考代码示例

本书在每个程序示例中都会指明示例项目的名称（使用 Project 标签）和源文件位置（使用 Source location 标签）。如果没有在程序示例中指明 Project 和 Source location 标签，那么你可以认为程序示例中的代码并不是从示例项目中摘录出来的，纯粹是为了帮助你理解而给出的。

## 本书使用的体例

斜体字用于强调术语。

Consolas 用于程序示例，Java 代码，XML 中的配置细节、属性文件以及展示程序输出。

Consolas 用于在程序示例和程序输出中突出重要的代码或者配置。



### 注意

这样的标注突出了一个重要的点或概念。

## 反馈和问题

你可以在异步社区中本书的图书详情页发送反馈和问题。

## 关于作者

J. 夏尔马 (J. Sharma) 拥有丰富的 Spring 应用开发经验，是一位自由职业的 Java 开发者。

阿西施·萨林 (Ashish Sarin) 拥有超过 18 年的应用程序架构设计经验，是 Sun 认证企业架构师。他也是 *Spring Roo 1.1 Cookbook* (Packt 出版社) 和 *Portlets in Action* (Manning 出版社) 的作者。

# 资源与支持

本书由异步社区出品，社区（<https://www.epubit.com/>）为您提供相关资源和后续服务。

## 配套资源

本书为读者提供源代码。

请在异步社区本书页面中单击 **配套资源**，跳转到下载界面，按提示进行操作即可。注意：为保证购书读者的权益，该操作会给出相关提示，要求输入提取码进行验证。

## 提交勘误

作者和编辑尽最大努力来确保书中内容的准确性，但难免会存在疏漏。欢迎您将发现的问题反馈给我们，帮助我们提升图书的质量。

当您发现错误时，请登录异步社区，按书名搜索，进入本书页面，单击“提交勘误”，输入勘误信息，单击“提交”按钮即可。本书的作者和编辑会对您提交的勘误进行审核，确认并接受后，将赠予您异步社区的 100 积分（积分可用于在异步社区兑换优惠券、样书或奖品）。

The screenshot shows a web form for submitting勘误 (勘误). At the top, there are three tabs: '详细信息' (Detailed Information), '写书评' (Write a Review), and '提交勘误' (Submit勘误), with the latter being the active tab. Below the tabs, there are three input fields: '页码:' (Page Number), '页内位置 (行数):' (Position within page (Line Number)), and '勘误印次:' (勘误印次). Below these fields is a rich text editor with a toolbar containing icons for Bold (B), Italic (I), Underline (U), Bulleted List, Numbered List, Link, and Unlink. The editor area contains some faint, illegible text. In the bottom right corner of the form, there is a '字数统计' (Character Count) label and a '提交' (Submit) button.

## 扫码关注本书

扫描下方二维码，您将会在异步社区微信服务号中看到本书信息及相关的服务提示。



## 与我们联系

我们的联系邮箱是 [contact@epubit.com.cn](mailto:contact@epubit.com.cn)。

如果您对本书有任何疑问或建议，请您发邮件给我们，并在邮件标题中注明本书书名，以便我们更高效地做出反馈。

如果您有兴趣出版图书、录制教学视频，或者参与图书翻译、技术审校等工作，可以发邮件给我们；有意出版图书的作者也可以到异步社区在线提交投稿（直接访问 [www.epubit.com/selfpublish/submission](http://www.epubit.com/selfpublish/submission) 即可）。

如果您来自学校、培训机构或企业，想批量购买本书或异步社区出版的其他图书，也可以发邮件给我们。

如果您在网上发现有针对异步社区出品图书的各种形式的盗版行为，包括对图书全部或部分内容的非授权传播，请您将怀疑有侵权行为的链接发邮件给我们。您的这一举动是对作者权益的保护，也是我们持续为您提供有价值的内容的动力之源。

## 关于异步社区和异步图书

“异步社区”是人民邮电出版社旗下 IT 专业图书社区，致力于出版精品 IT 技术图书和相关学习产品，为译者提供优质出版服务。异步社区创办于 2015 年 8 月，提供大量精品 IT 技术图书和电子书，以及高品质技术文章和视频课程。更多详情请访问异步社区官网 <https://www.epubit.com>。

“异步图书”是由异步社区编辑团队策划出版的精品 IT 专业图书的品牌，依托于人民邮电出版社近 30 年的计算机图书出版积累和专业编辑团队，相关图书在封面上印有异步图书的 LOGO。异步图书的出版领域包括软件开发、大数据、AI、测试、前端、网络技术等。



异步社区



微信服务号

# 目 录

<b>第 1 章 Spring 框架概述</b> .....	<b>1</b>
1.1 简介.....	1
1.2 Spring 框架的模块.....	1
1.3 Spring IoC 容器.....	2
1.4 使用 Spring 框架的好处.....	4
1.4.1 管理本地和全局事务的一致方法.....	4
1.4.2 声明式事务管理.....	6
1.4.3 安全.....	6
1.4.4 Java 管理扩展.....	7
1.4.5 Java 消息服务.....	8
1.4.6 缓存.....	8
1.5 一个简单的 Spring 应用程序.....	9
1.5.1 确定应用程序对象及其依赖关系.....	9
1.5.2 根据确定的应用程序对象创建 POJO 类.....	10
1.5.3 创建配置元数据.....	10
1.5.4 通过 setter 方法注入依赖项.....	12
1.5.5 创建 Spring 容器的一个实例.....	14
1.5.6 从 Spring 容器访问 bean.....	15
1.6 Spring 框架 5 的新功能.....	16
1.7 建立在 Spring 之上的框架.....	16
1.8 小结.....	17
<b>第 2 章 Spring 框架基础</b> .....	<b>18</b>
2.1 简介.....	18
2.2 面向接口编程的设计方法.....	18
2.3 使用静态和实例工厂方法创建 Spring bean.....	21
2.3.1 通过静态工厂方法实例化 bean.....	21
2.3.2 通过实例工厂方法实例化 bean.....	22
2.4 基于构造函数的 DI.....	25
2.4.1 回顾基于 setter 的 DI.....	25
2.4.2 基于构造函数的 DI.....	26
2.5 将配置详细信息传递给 bean.....	27
2.6 bean 的作用域.....	29
2.6.1 singleton.....	29
2.6.2 prototype.....	35
2.6.3 为你的 bean 选择适当的范围.....	36
2.7 小结.....	36
<b>第 3 章 bean 的配置</b> .....	<b>37</b>

3.1	简介	37
3.2	bean 定义的继承	37
3.2.1	MyBank —— bean 定义继承示例	37
3.2.2	继承了什么	39
3.3	构造函数参数匹配	43
3.3.1	使用<constructor-arg>元素传递简单的值和 bean 引用	43
3.3.2	基于类型的构造方法参数匹配	44
3.3.3	基于名称的构造函数参数匹配	47
3.4	配置不同类型的 bean 属性和构造函数参数	50
3.4.1	Spring 的内置属性编辑器	50
3.4.2	指定不同集合类型的值	52
3.4.3	指定数组的值	57
3.4.4	与<list>、<set>和<map>元素相对应的默认实现	58
3.5	内置属性编辑器	58
3.5.1	CustomCollectionEditor	58
3.5.2	CustomMapEditor	60
3.5.3	CustomDateEditor	61
3.6	向 Spring 容器注册属性编辑器	61
3.6.1	创建一个 PropertyEditorRegistrar 实现	61
3.6.2	配置 CustomEditorConfigurer 类	62
3.7	具有 p 和 c 命名空间的简明 bean 定义	62
3.7.1	p 命名空间	63
3.7.2	c 命名空间	63
3.8	Spring 的 util 模式	65
3.8.1	<list>元素	66
3.8.2	<map>元素	66
3.8.3	<set>元素	67
3.8.4	<properties>元素	68
3.8.5	<constant>元素	68
3.8.6	<property-path>元素	69
3.9	FactoryBean 接口	70
3.9.1	MyBank application —— 将事件存储在数据库中	70
3.9.2	MyBank —— FactoryBean 示例	71
3.9.3	访问 FactoryBean 实例	73
3.10	模块化 bean 配置	75
3.11	小结	75
<b>第 4 章 依赖注入</b>		<b>76</b>
4.1	简介	76
4.2	内部 bean	76
4.3	使用 depends-on 特性控制 bean 的初始化顺序	77
4.3.1	MyBank —— bean 之间隐式的依赖关系	78
4.3.2	隐性依赖问题	78

4.4	singleton 和 prototype 范围的 bean 的依赖项	82
4.4.1	singleton bean 的依赖项	82
4.4.2	Prototype bean 的依赖项	84
4.5	通过 singleton bean 中获取 prototype bean 的新实例	86
4.5.1	ApplicationContextAware 接口	87
4.5.2	<lookup-method> 元素	88
4.5.3	<replaced-method> 元素	91
4.6	自动装配依赖项	93
4.6.1	byType	93
4.6.2	constructor	95
4.6.3	byName	96
4.6.4	default / no	96
4.6.5	使 bean 无法用于自动装配	97
4.6.6	自动装配的局限性	98
4.7	小结	98
<b>第 5 章 自定义 bean 和 bean 定义</b>		<b>99</b>
5.1	简介	99
5.2	自定义 bean 的初始化和销毁逻辑	99
5.2.1	使 Spring 调用由 destroy-method 特性指定的 cleanup 方法	101
5.2.2	清理方法和 prototype bean	103
5.2.3	为所有 bean 指定默认的 bean 初始化和销毁方法	103
5.2.4	InitializingBean 和 DisposableBean 生命周期接口	103
5.2.5	JSR 250's @PostConstruct 和 @PreDestroy 注解	103
5.3	使用 BeanPostProcessor 与新创建的 bean 实例进行交互	105
5.3.1	BeanPostProcessor 示例——验证 bean 实例	106
5.3.2	BeanPostProcessor 示例——解析 bean 依赖项	109
5.3.3	FactoryBeans 的 BeanPostProcessor 行为	112
5.3.4	RequiredAnnotationBeanPostProcessor	113
5.3.5	DestructionAwareBeanPostProcessor	114
5.4	使用 BeanFactoryPostProcessor 修改 bean 定义	114
5.4.1	BeanFactoryPostProcessor 示例	115
5.4.2	PropertySourcesPlaceholderConfigurer	118
5.4.3	PropertyOverrideConfigurer	123
5.5	小结	125
<b>第 6 章 使用 Spring 进行注解驱动开发</b>		<b>126</b>
6.1	简介	126
6.2	用@Component 标识 Spring bean	126
6.3	@Autowired 通过类型自动装配依赖项	128
6.4	@Qualifier 按名称自动装配依赖项	131
6.4.1	使用 qualifiers 自动装配 bean	132
6.4.2	创建自定义限定符注解	133

6.5	JSR 330 的@Inject 和@Named 注解	135
6.6	JSR 250 的@Resource 注解	137
6.7	@Scope、@Lazy、@DependsOn 和@Primary 注解	138
6.7.1	@Scope	139
6.7.2	@Lazy	139
6.7.3	@DependsOn	142
6.7.4	@Primary	142
6.8	使用@Value 简化注解的 bean 类的配置	142
6.8.1	在@Value 注解中使用 Spring 表达式语言 (SpEL)	143
6.8.2	在方法级和方法参数级使用@Value 注解	145
6.8.3	在 SpEL 中使用数学、关系和逻辑运算符	145
6.8.4	使用 SpEL 获取 bean 的引用	146
6.8.5	在 SpEL 中使用正则表达式	146
6.8.6	在 SpEL 中使用映射和列表	146
6.8.7	在基于 XML 的 bean 定义中指定 SpEL 表达式	147
6.9	使用 Spring 的 Validator 接口验证对象	148
6.10	使用 JSR 380 (Bean Validation 2.0) 注解指定约束	151
6.10.1	Spring 中的 JSR 380 支持	152
6.10.2	JSR 380 有什么新功能	157
6.11	bean 定义配置文件	159
6.12	小结	163
<b>第 7 章 基于 Java 的容器配置</b>		<b>164</b>
7.1	简介	164
7.2	使用@Configuration 和@Bean 注解配置 bean	164
7.3	注入 bean 依赖项	168
7.4	配置 Spring 容器	170
7.5	生命周期回调	172
7.6	导入基于 Java 的配置	173
7.7	附加主题	175
7.7.1	覆盖@Bean 方法	175
7.7.2	配置 BeanPostProcessors 和 BeanFactoryPostProcessors	178
7.7.3	导入应用程序上下文 XML 文件	178
7.7.4	有条件地包含@Bean 和@Configuration 类	180
7.8	小结	185
<b>第 8 章 使用 Spring 进行数据库交互</b>		<b>186</b>
8.1	简介	186
8.2	MyBank 应用程序的需求	186
8.3	使用 Spring JDBC 模块开发 MyBank 应用程序	187
8.3.1	配置数据源	187
8.3.2	创建使用 Spring 的 JDBC 模块类的 DAO	188
8.4	使用 Hibernate 开发 MyBank 应用程序	194

8.4.1	配置 SessionFactory 实例	194
8.4.2	创建使用 Hibernate API 进行数据库交互的 DAO	195
8.5	使用 Spring 的事务管理	196
8.5.1	MyBank 的事务管理需求	196
8.5.2	编程式事务管理	196
8.5.3	声明式事务管理	199
8.5.4	Spring 对 JTA 的支持	202
8.6	使用基于 Java 的配置开发 MyBank 应用程序	203
8.6.1	配置 javax.sql.DataSource	203
8.6.2	配置 Hibernate 的 SessionFactory	204
8.6.3	启用@Transactional 支持	204
8.7	小结	205
<b>第 9 章 Spring Data</b>		<b>206</b>
9.1	简介	206
9.2	核心概念和接口	206
9.3	Spring Data JPA	209
9.3.1	代替存储库方法的自定义实现	210
9.3.2	将自定义方法添加到存储库	211
9.3.3	配置 Spring Data JPA —— 基于 Java 的配置方法	212
9.3.4	配置 Spring Data JPA —— 基于 XML 的配置方法	214
9.3.5	查询方法	215
9.4	使用 Querydsl 创建查询	220
9.4.1	将 Spring Data 与 Querydsl 集成	221
9.4.2	构造 Predicate	221
9.5	按示例查询	223
9.6	Spring Data MongoDB	224
9.6.1	建模域实体	225
9.6.2	配置 Spring Data MongoDB —— 基于 Java 的配置	226
9.6.3	配置 Spring Data MongoDB —— 基于 XML 的配置	227
9.6.4	创建自定义存储库	228
9.6.5	将自定义方法添加到存储库	228
9.6.6	使用 Querydsl 创建查询	229
9.6.7	使用 Query by Example 创建查询	230
9.7	小结	231
<b>第 10 章 使用 Spring 进行消息传递、电子邮件发送、异步方法执行和缓存</b>		<b>232</b>
10.1	简介	232
10.2	MyBank 应用程序的需求	232
10.3	发送 JMS 消息	233
10.3.1	配置 ActiveMQ 代理以在内嵌模式下运行	234
10.3.2	配置一个 JMS ConnectionFactory	234
10.3.3	使用 JmsTemplate 发送 JMS 消息	235

10.3.4	在事务中发送 JMS 消息	236
10.3.5	动态 JMS 目标和 JmsTemplate 配置	239
10.3.6	JmsTemplate 和消息转换	239
10.4	接收 JMS 消息	240
10.4.1	使用 JmsTemplate 同步接收 JMS 消息	240
10.4.2	使用消息侦听器容器异步接收 JMS 消息	240
10.4.3	使用 @JmsListener 注册 JMS 侦听器端点	242
10.4.4	使用 spring-messaging 模块的消息传递	243
10.5	发送电子邮件	245
10.5.1	使用 MimeMessageHelper 准备 MIME 消息	248
10.5.2	使用 MimeMessagePreparator 准备 MIME 消息	249
10.6	任务调度和异步执行	249
10.6.1	TaskExecutor 接口	249
10.6.2	TaskScheduler 接口	251
10.6.3	调度 bean 方法的执行	252
10.6.4	@Async 和 @Scheduled 注解	252
10.7	缓存	254
10.7.1	配置一个 CacheManager	255
10.7.2	缓存注解——@Cacheable、@CacheEvict 和 @CachePut	255
10.7.3	使用 Spring cache 模式进行缓存配置	258
10.8	运行 MyBank 应用程序	259
10.9	小结	261
<b>第 11 章 面向切面编程</b>		<b>262</b>
11.1	简介	262
11.2	一个简单的 AOP 示例	262
11.3	Spring AOP 框架	264
11.3.1	代理的创建	265
11.3.2	expose-proxy 特性	266
11.4	切入点表达式	267
11.4.1	@Pointcut 注解	267
11.4.2	execution 和 args 切入点指示符	268
11.4.3	bean 切入点指示器	270
11.4.4	基于注解的切入点指示符	271
11.5	通知类型	272
11.5.1	前置通知	272
11.5.2	返回后通知	272
11.5.3	抛出后通知	273
11.5.4	后置通知	274
11.5.5	围绕通知	274
11.5.6	通过实现特殊接口创建通知	275
11.6	Spring AOP - XML 模式样式	276
11.6.1	配置一个 AOP 切面	276

11.6.2	配置一个通知	277
11.6.3	将切入点表达式与通知相关联	278
11.7	小结	278
<b>第 12 章 Spring Web MVC 基础知识</b>		<b>279</b>
12.1	简介	279
12.2	示例 Web 项目的目录结构	279
12.3	了解“Hello World”网络应用程序	280
12.3.1	HelloWorldController.java——Hello World Web 应用程序的控制器类	280
12.3.2	helloworld.jsp——展示“Hello World !!”消息的 JSP 页面	282
12.3.3	myapp-config.xml——Web 应用程序上下文 XML 文件	282
12.3.4	web.xml——Web 应用程序部署描述符	283
12.4	DispatcherServlet——前端控制器	285
12.5	使用@Controller 和@RequestMapping 注解开发控制器	287
12.6	MyBank Web 应用程序的需求	289
12.7	Spring Web MVC 注解——@RequestMapping 和@RequestParam	290
12.7.1	使用@RequestMapping 将请求映射到控制器或者控制器方法	290
12.7.2	@RequestMapping 注解方法的参数	295
12.7.3	@RequestMapping 注解方法的返回类型	295
12.7.4	使用@RequestParam 将请求参数传递给控制器方法	297
12.8	验证	300
12.9	使用@ExceptionHandler 注解处理异常	302
12.10	加载根 Web 应用程序上下文 XML 文件	303
12.11	小结	304
<b>第 13 章 Spring Web MVC 中的验证和数据绑定</b>		<b>305</b>
13.1	简介	305
13.2	使用@ModelAttribute 注解添加和获取模型特性	305
13.2.1	使用方法级的@ModelAttribute 注解添加模型特性	306
13.2.2	使用@ModelAttribute 注解获取模型特性	309
13.2.3	请求处理及@ModelAttribute 注解的方法	310
13.2.4	使用@ModelAttribute 注解的方法参数的行为	311
13.2.5	RequestToViewNameTranslator 对象	311
13.3	使用@SessionAttributes 注解缓存模型特性	312
13.4	Spring 中对数据绑定的支持	314
13.4.1	WebDataBinder——Web 请求参数的数据绑定器	316
13.4.2	配置 WebDataBinder 实例	317
13.4.3	允许或禁止字段参与数据绑定过程	320
13.4.4	使用 BindingResult 对象检查数据绑定和验证错误	322
13.5	Spring 中的验证支持	323
13.5.1	使用 Spring 的 Validator 接口验证模型特性	323
13.5.2	使用 JSR 380 注解指定约束	326
13.5.3	使用 JSR 380 注解验证对象	327

13.6	Spring 的 form 标签库	329
13.7	使用基于 Java 的配置方式来配置 Web 应用程序	331
13.8	小结	333
<b>第 14 章 使用 Spring Web MVC 开发 RESTful Web 服务</b>		<b>334</b>
14.1	简介	334
14.2	定期存款 Web 服务	334
14.3	使用 Spring Web MVC 实现 RESTful Web 服务	335
14.3.1	JSON (JavaScript 对象表示法)	336
14.3.2	FixedDepositWS Web 服务的实现	337
14.4	使用 RestTemplate 和 WebClient 访问 RESTful Web 服务	342
14.4.1	RestTemplate 的配置	342
14.4.2	使用 RestTemplate 访问 FixedDepositWS Web 服务	343
14.4.3	使用 WebClient 异步访问 RESTful Web 服务	347
14.5	使用 HttpMessageConverter 将 Java 对象与 HTTP 请求和响应相互转换	348
14.6	@PathVariable 和 @MatrixVariable 注解	349
14.7	小结	352
<b>第 15 章 Spring Web MVC 进阶——国际化、文件上传和异步请求处理</b>		<b>353</b>
15.1	简介	353
15.2	使用处理程序拦截器对请求进行预处理和后处理	353
15.3	使用资源束进行国际化	355
15.3.1	MyBank Web 应用程序的需求	355
15.3.2	MyBank Web 应用程序的国际化和本地化	355
15.4	异步地处理请求	357
15.4.1	异步请求处理配置	358
15.4.2	从 @RequestMapping 方法返回 Callable	358
15.4.3	从 @RequestMapping 方法中返回 DeferredResult	359
15.4.4	设置默认超时时间	365
15.4.5	拦截异步请求	365
15.5	Spring 中的类型转换和格式化支持	366
15.5.1	创建自定义转换器	366
15.5.2	配置和使用自定义转换器	366
15.5.3	创建一个自定义的格式化器	368
15.5.4	配置一个自定义格式化器	369
15.5.5	创建 AnnotationFormatterFactory 以格式化仅使用 @AmountFormat 注解的字段	369
15.5.6	配置 AnnotationFormatterFactory 的实现	370
15.6	Spring Web MVC 中的文件上传支持	372
15.6.1	使用 CommonsMultipartResolver 上传文件	372
15.6.2	使用 StandardServletMultipartResolver 传文件	374
15.7	小结	374
<b>第 16 章 使用 Spring Security 保护应用程序</b>		<b>375</b>

16.1	简介	375
16.2	MyBank Web 应用程序的安全性需求	375
16.3	使用 Spring Security 保护 MyBank Web 应用程序	376
16.3.1	Web 请求安全的配置	376
16.3.2	身份认证配置	378
16.3.3	使用 Spring Security 的 JSP 标签库保护 JSP 内容	379
16.3.4	保护方法	380
16.4	MyBank Web 应用程序——使用 Spring Security 的 ACL 模块保护 FixedDeposit Details 实例	383
16.4.1	部署和使用 ch16-bankapp-db-security 项目	383
16.4.2	存储 ACL 和用户信息的数据库表	385
16.4.3	用户认证	387
16.4.4	Web 请求安全	388
16.4.5	JdbcMutableAclService 配置	389
16.4.6	方法级安全配置	391
16.4.7	域对象实例的安全	392
16.4.8	以编程方式管理 ACL 条目	394
16.4.9	MutableAcl 及安全性	396
16.5	使用基于 Java 的配置方法配置 Spring Security	397
16.5.1	使用 WebSecurityConfigurerAdapter 类配置 Web 请求安全	397
16.5.2	使用 GlobalMethodSecurity Configuration 类配置方法级安全	398
16.5.3	将 DelegatingFilterProxy 过滤器注册到 ServletContext	398
16.5.4	将 DispatcherServlet 和 Context LoaderListener 注册到 ServletContext	399
16.6	小结	399
<b>第 17 章 Java 的函数式编程</b>		<b>400</b>
17.1	简介	400
17.2	命令式和函数式编程风格	400
17.3	lambda 表达式	401
17.4	创建简单函数和高阶函数	404
17.4.1	简单函数	405
17.4.2	高阶函数	406
17.5	流 API	408
17.5.1	中间操作和终结操作	409
17.5.2	延迟求值	411
17.5.3	顺序流和并行流	413
17.6	方法引用	415
17.7	小结	416
<b>第 18 章 RxJava 2 的反应式编程</b>		<b>417</b>
18.1	简介	417
18.2	反应式流	418
18.3	冷发布者和热发布者	422
18.4	背压	430

18.5 小结	434
<b>第 19 章 用 Spring WebFlux、Spring Data 和 Spring Security 开发反应式 RESTful Web 服务</b>	<b>435</b>
19.1 简介	435
19.2 Reactor 和 RxJava 2 定义的反应式类型	435
19.3 用 Spring Data 开发数据访问层	437
19.3.1 Reactor	437
19.3.2 RxJava 2	441
19.4 使用 Spring WebFlux 开发 Web 层	444
19.4.1 编写反应式 Web 控制器	445
19.4.2 配置 Spring WebFlux	445
19.4.3 配置 ServletContext	446
19.4.4 使用 WebClient 与反应式 RESTful Web 服务交互	446
19.4.5 使用服务器发送事件接收数据	449
19.5 保护 WebFlux 应用程序的安全性	450
19.6 小结	454
<b>附录 A 下载和安装 MongoDB 数据库</b>	<b>455</b>
A.1 下载并安装 MongoDB 数据库	455
A.2 连接 MongoDB 数据库	455
<b>附录 B 在 Eclipse IDE 中导入和运行示例项目</b>	<b>457</b>
B.1 下载和安装 Eclipse IDE 和 Tomcat 9	457
B.2 将示例项目导入 Eclipse IDE 中	457
B.2.1 将示例项目导入 Eclipse IDE 中	457
B.2.2 在 Eclipse IDE 中配置 M2_REPO 类路径变量	458
B.3 在 Eclipse IDE 中配置 Tomcat 9 服务器	458
B.4 在 Tomcat 9 服务器上部署 Web 项目	460