



Pro Spring 5

An In-Depth Guide to the Spring
Framework and Its Tools

—

Fifth Edition

—

Iuliana Cosmina
Rob Harrop
Chris Schaefer
Clarence Ho

Apress®

Pro Spring 5

An In-Depth Guide to the Spring
Framework and Its Tools

Fifth Edition



Iuliana Cosmina

Rob Harrop

Chris Schaefer

Clarence Ho

Apress®

Pro Spring 5: An In-Depth Guide to the Spring Framework and Its Tools

Iuliana Cosmina
Sibiu, Sibiu, Romania

Rob Harrop
Reddish, United Kingdom

Chris Schaefer
Venice, Florida, USA

Clarence Ho
Hong Kong, China

ISBN-13 (pbk): 978-1-4842-2807-4
DOI 10.1007/978-1-4842-2808-1

ISBN-13 (electronic): 978-1-4842-2808-1

Library of Congress Control Number: 2017955423

Copyright © 2017 by Iuliana Cosmina, Rob Harrop, Chris Schaefer, and Clarence Ho

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

Trademarked names, logos, and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image we use the names, logos, and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Cover image by Freepik (www.freepik.com)

Managing Director: Welmoed Spahr
Editorial Director: Todd Green
Acquisitions Editor: Steve Anglin
Development Editor: Matthew Moodie
Technical Reviewer: Massimo Nardone
Coordinating Editor: Mark Powers
Copy Editor: Kim Wimpsett

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail rights@apress.com, or visit www.apress.com/rights-permissions.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/9781484228074. For more detailed information, please visit www.apress.com/source-code.

Printed on acid-free paper

I dedicate this book to my friends, to my godson ștefan, and to all the musicians who have made working on this book easy with their music.

—Iuliana Cosmina

Contents at a Glance

About the Authors	xxiii
About the Technical Reviewer	xxv
Acknowledgments	xxvii
Introduction	xxix
■ Chapter 1: Introducing Spring	1
■ Chapter 2: Getting Started	19
■ Chapter 3: Introducing IoC and DI in Spring	37
■ Chapter 4: Spring Configuration in Detail and Spring Boot	125
■ Chapter 5: Introducing Spring AOP	211
■ Chapter 6: Spring JDBC Support	297
■ Chapter 7: Using Hibernate in Spring	355
■ Chapter 8: Data Access in Spring with JPA2	393
■ Chapter 9: Transaction Management	467
■ Chapter 10: Validation with Type Conversion and Formatting	509
■ Chapter 11: Task Scheduling	537
■ Chapter 12: Using Spring Remoting	557
■ Chapter 13: Spring Testing	615
■ Chapter 14: Scripting Support in Spring	639
■ Chapter 15: Application Monitoring	655

■ Chapter 16: Web Applications	665
■ Chapter 17: WebSocket	751
■ Chapter 18: Spring Projects: Batch, Integration, XD, and More	773
■ Appendix A: Setting Up Your Development Environment	829
Index	841

Contents

About the Authors	xxiii
About the Technical Reviewer	xxv
Acknowledgments	xxvii
Introduction	xxix
■ Chapter 1: Introducing Spring	1
What Is Spring?	1
Evolution of the Spring Framework	2
Inverting Control or Injecting Dependencies?.....	8
Evolution of Dependency Injection	8
Beyond Dependency Injection	10
The Spring Project	15
Origins of Spring.....	15
The Spring Community	15
The Spring Tool Suite.....	16
The Spring Security Project.....	16
Spring Boot.....	16
Spring Batch and Integration.....	17
Many Other Projects	17
Alternatives to Spring	17
JBoss Seam Framework.....	17
Google Guice.....	17
PicoContainer	17
JEE 7 Container	18
Summary	18

- **Chapter 2: Getting Started** **19**
- Obtaining the Spring Framework **20**
 - Getting Started Quickly **20**
 - Checking Spring Out of GitHub **20**
 - Using the Right JDK **21**
- Understanding Spring Packaging **21**
 - Choosing Modules for Your Application **24**
 - Accessing Spring Modules on the Maven Repository **24**
 - Accessing Spring Modules Using Gradle **26**
 - Using Spring Documentation **26**
 - Putting a Spring into Hello World **27**
 - Building the Sample Hello World Application **27**
 - Refactoring with Spring **31**
- Summary **35**
- **Chapter 3: Introducing IoC and DI in Spring** **37**
- Inversion of Control and Dependency Injection **37**
- Types of Inversion of Control **38**
 - Dependency Pull **38**
 - Contextualized Dependency Lookup **39**
 - Constructor Dependency Injection **40**
 - Setter Dependency Injection **41**
 - Injection vs. Lookup **41**
 - Setter Injection vs. Constructor Injection **42**
- Inversion of Control in Spring **45**
- Dependency Injection in Spring **46**
 - Beans and BeanFactory **46**
 - BeanFactory Implementations **46**
 - ApplicationContext **48**
- Configuring ApplicationContext **48**
 - Setting Spring Configuration Options **49**
 - Basic Configuration Overview **49**

Declaring Spring Components	50
Using Method Injection	84
Understanding Bean Naming	95
Understanding Bean Instantiation Mode	105
Resolving Dependencies	109
Autowiring Your Bean	112
When to Use Autowiring	121
Setting Bean Inheritance	122
Summary	124
■ Chapter 4: Spring Configuration in Detail and Spring Boot	125
Spring’s Impact on Application Portability	126
Bean Life-Cycle Management	127
Hooking into Bean Creation	128
Executing a Method When a Bean Is Created	128
Implementing the InitializingBean Interface	132
Using the JSR-250 @PostConstruct Annotation	134
Declaring an Initialization Method Using @Bean	137
Understanding Order of Resolution	138
Hooking into Bean Destruction	139
Executing a Method When a Bean Is Destroyed	139
Implementing the DisposableBean Interface	141
Using the JSR-250 @PreDestroy Annotation	143
Declaring a Destroy Method Using @Bean	144
Understanding Order of Resolution	146
Using a Shutdown Hook	146
Making Your Beans “Spring Aware”	146
Using the BeanNameAware Interface	147
Using the ApplicationContextAware Interface	148
Use of FactoryBeans	151
FactoryBean Example: The MessageDigestFactoryBean	151

Accessing a FactoryBean Directly	156
Using the factory-bean and factory-method Attributes.....	157
JavaBeans PropertyEditors	158
Using the Built-in PropertyEditors	159
Creating a Custom PropertyEditor	164
More Spring ApplicationContext Configuration	167
Internationalization with the MessageSource	168
Using MessageSource in Stand-Alone Applications	171
Application Events	171
Accessing Resources	174
Configuration Using Java Classes	175
ApplicationContext Configuration in Java.....	175
Spring Mixed Configuration	185
Java or XML Configuration?.....	187
Profiles	187
An Example of Using the Spring Profiles Feature	187
Spring Profiles Using Java Configuration	190
Considerations for Using Profiles	193
Environment and PropertySource Abstraction	193
Configuration Using JSR-330 Annotations	198
Configuration Using Groovy	201
Spring Boot.....	204
Summary	210
■ Chapter 5: Introducing Spring AOP.....	211
AOP Concepts.....	212
Types of AOP.....	213
Using Static AOP	213
Using Dynamic AOP	213
Choosing an AOP Type	213

AOP in Spring	214
The AOP Alliance.....	214
Hello World in AOP	214
Spring AOP Architecture	216
Joinpoints in Spring.....	216
Aspects in Spring.....	217
About the ProxyFactory Class.....	217
Creating Advice in Spring	217
Interfaces for Advice.....	219
Creating Before Advice	219
Securing Method Access by Using Before Advice	220
Creating After-Returning Advice	224
Creating Around Advice	227
Creating Throws Advice	230
Choosing an Advice Type	232
Advisors and Pointcuts in Spring	233
The Pointcut Interface	233
Available Pointcut Implementations	235
Using DefaultPointcutAdvisor	236
Creating a Static Pointcut by Using StaticMethodMatcherPointcut.....	236
Creating a Dynamic Pointcut by Using DyanmicMethodMatcherPointcut	239
Using Simple Name Matching	242
Creating Pointcuts with Regular Expressions.....	244
Creating Pointcuts with AspectJ Pointcut Expression	246
Creating Annotation Matching Pointcuts	247
Convenience Advisor Implementations.....	248
Understanding Proxies	249
Using JDK Dynamic Proxies.....	250
Using CGLIB Proxies	250
Comparing Proxy Performance.....	251
Choosing a Proxy to Use.....	256

Advanced Use of Pointcuts	256
Using Control Flow Pointcuts.....	256
Using a Composable Pointcut.....	259
Composition and the Pointcut Interface	262
Pointcut Summary	262
Getting Started with Introductions	263
Introduction Basics.....	263
Object Modification Detection with Introductions	265
Introduction Summary	270
Framework Services for AOP	271
Configuring AOP Declaratively	271
Using ProxyFactoryBean.....	271
Using the aop Namespace.....	277
Using @AspectJ-Style Annotations	284
Considerations for Declarative Spring AOP Configuration	291
AspectJ Integration	291
About AspectJ.....	291
Using Singleton Aspects	292
Summary	295
■ Chapter 6: Spring JDBC Support	297
Introducing Lambda Expressions	298
Sample Data Model for Example Code	298
Exploring the JDBC Infrastructure	304
Spring JDBC Infrastructure	309
Overview and Used Packages	309
Database Connections and DataSources	310
Embedded Database Support.....	315
Using DataSources in DAO Classes	317
Exception Handling.....	319

The JdbcTemplate Class.....	321
Initializing JdbcTemplate in a DAO Class.....	321
Using Named Parameters with NamedParameterJdbcTemplate.....	323
Retrieving Domain Objects with RowMapper<T>	325
Retrieving Nested Domain Objects with ResultSetExtractor	327
Spring Classes That Model JDBC Operations	330
Querying Data by Using MappingSqlQuery<T>.....	333
Updating Data by Using SqlUpdate	337
Inserting Data and Retrieving the Generated Key	340
Batching Operations with BatchSqlUpdate	342
Calling Stored Functions by Using SqlFunction.....	347
Spring Data Project: JDBC Extensions.....	349
Considerations for Using JDBC.....	349
Spring Boot JDBC	350
Summary.....	353
■ Chapter 7: Using Hibernate in Spring	355
Sample Data Model for Example Code	356
Configuring Hibernate’s SessionFactory	358
ORM Mapping Using Hibernate Annotations	362
Simple Mappings.....	363
One-to-Many Mappings	367
Many-to-Many Mappings	369
The Hibernate Session Interface	371
Querying Data by Using the Hibernate Query Language	372
Simple Querying with Lazy Fetching	372
Querying with Associations Fetching	375
Inserting Data.....	378
Updating Data.....	382
Deleting Data.....	384

Configuring Hibernate to Generate Tables from Entities	386
Annotating Methods or Fields?	389
Considerations When Using Hibernate	391
Summary	392
■ Chapter 8: Data Access in Spring with JPA2.....	393
Introducing JPA 2.1	394
Sample Data Model for Example Code	394
Configuring JPA's EntityManagerFactory.....	395
Using JPA Annotations for ORM Mapping	398
Performing Database Operations with JPA	400
Using the Java Persistence Query Language to Query Data.....	400
Querying with Untyped Results	410
Querying for a Custom Result Type with a Constructor Expression	412
Inserting Data	415
Updating Data	417
Deleting data	419
Using a Native Query	420
Using a Simple Native Query	421
Native Querying with SQL ResultSet Mapping	421
Using the JPA 2 Criteria API for a Criteria Query	422
Introducing Spring Data JPA	429
Adding Spring Data JPA Library Dependencies	429
Using Spring Data JPA Repository Abstraction for Database Operations	430
Using JpaRepository	436
Spring Data JPA with Custom Queries	437
Keeping Track of Changes on the Entity Class.....	440
Keeping Entity Versions by Using Hibernate Envers.....	451
Adding Tables for Entity Versioning	452
Configuring EntityManagerFactory for Entity Versioning	453

Enabling Entity Versioning and History Retrieval	456
Testing Entity Versioning.....	457
Spring Boot JPA.....	459
Considerations When Using JPA.....	465
Summary.....	466
■ Chapter 9: Transaction Management.....	467
Exploring the Spring Transaction Abstraction Layer	468
Transaction Types	468
Implementations of the PlatformTransactionManager	469
Analyzing Transaction Properties	470
The TransactionDefinition Interface	471
The TransactionStatus Interface	472
Sample Data Model and Infrastructure for Example Code	473
Creating a Simple Spring JPA Project with Dependencies	473
Sample Data Model and Common Classes.....	475
Using AOP Configuration for Transaction Management	486
Using Programmatic Transactions.....	488
Considerations on Transaction Management.....	490
Global Transactions with Spring	490
Infrastructure for Implementing the JTA Sample.....	491
Implementing Global Transactions with JTA.....	491
Spring Boot JTA	501
Considerations on Using JTA Transaction Manager.....	507
Summary.....	507
■ Chapter 10: Validation with Type Conversion and Formatting	509
Dependencies.....	510
Spring Type Conversion System	510
Conversion from a String Using PropertyEditors	511

Introducing Spring Type Conversion.....	514
Implementing a Custom Converter	514
Configuring ConversionService	515
Converting Between Arbitrary Types.....	517
Field Formatting in Spring.....	521
Implementing a Custom Formatter.....	521
Configuring ConversionServiceFactoryBean	523
Validation in Spring	524
Using the Spring Validator Interface.....	525
Using JSR-349 Bean Validation	527
Configuring Bean Validation Support in Spring.....	528
Creating a Custom Validator	531
Using AssertTrue for Custom Validation	534
Considerations for Custom Validation	535
Deciding Which Validation API to Use	535
Summary.....	535
■ Chapter 11: Task Scheduling	537
Dependencies for the Task Scheduling Samples	537
Task Scheduling in Spring.....	538
Introducing the Spring TaskScheduler Abstraction.....	539
Exploring a Sample Task.....	540
Using Annotations for Task Scheduling	547
Asynchronous Task Execution in Spring.....	551
Task Execution in Spring	554
Summary.....	556
■ Chapter 12: Using Spring Remoting	557
Using a Data Model for Samples	558
Adding Required Dependencies for the JPA Back End.....	560

Implementing and Configuring SingerService.....	562
Implementing SingerService	562
Configuring SingerService.....	564
Exposing the Service	567
Invoking the Service	568
Using JMS in Spring.....	570
Implementing a JMS Listener in Spring	573
Sending JMS Messages in Spring.....	574
Spring Boot Artemis Starter	576
Using RESTful-WS in Spring.....	579
Introducing RESTful Web Services	579
Adding Required Dependencies for Samples	580
Designing the Singer RESTful Web Service	580
Using Spring MVC to Expose RESTful Web Services.....	581
Configuring Castor XML.....	582
Implementing SingerController.....	584
Configuring a Spring Web Application	586
Using curl to Test RESTful-WS.....	590
Using RestTemplate to Access RESTful-WS.....	592
Securing RESTful-WS with Spring Security.....	597
RESTful-WS with Spring with Spring Boot	602
Using AMQP in Spring.....	605
Using AMQP with Spring Boot.....	611
Summary.....	613
■ Chapter 13: Spring Testing	615
Introducing Testing Categories.....	616
Using Spring Test Annotations.....	617
Implementing Logic Unit Tests	618
Adding Required Dependencies.....	619
Unit Testing Spring MVC Controllers	620

Implementing an Integration Test.....	623
Adding Required Dependencies.....	623
Configuring the Profile for Service-Layer Testing.....	623
Java Configuration Version	625
Implementing the Infrastructure Classes	627
Unit Testing the Service Layer	630
Dropping DbUnit	634
Implementing a Front-End Unit Test.....	637
Introducing Selenium	638
Summary.....	638
■ Chapter 14: Scripting Support in Spring	639
Working with Scripting Support in Java.....	640
Introducing Groovy	641
Dynamic Typing	642
Simplified Syntax.....	643
Closure.....	643
Using Groovy with Spring	644
Developing the Singer Domain	645
Implementing the Rule Engine.....	646
Implementing the Rule Factory as a Spring Refreshable Bean	648
Testing the Age Category Rule.....	650
Inlining Dynamic Language Code	652
Summary.....	654
■ Chapter 15: Application Monitoring	655
JMX Support in Spring	655
Exporting a Spring Bean to JMX.....	656
Using Java VisualVM for JMX Monitoring.....	657
Monitoring Hibernate Statistics.....	659
JMX with Spring Boot.....	661
Summary.....	664

■ Chapter 16: Web Applications	665
Implementing the Service Layer for Samples	666
Using a Data Model for the Samples	666
Implementing the DAO Layer	670
Implementing the Service Layer	670
Configuring SingerService.....	672
Introducing MVC and Spring MVC	673
Introducing MVC	674
Introducing Spring MVC.....	675
Spring MVC WebApplicationContext Hierarchy	675
Spring MVC Request Life Cycle	676
Spring MVC Configuration.....	678
Creating the First View in Spring MVC.....	681
Configuring DispatcherServlet.....	683
Implementing SingerController.....	684
Implementing the Singer List View	685
Testing the Singer List View	686
Understanding the Spring MVC Project Structure	686
Enabling Internationalization (i18n).....	687
Configuring i18n in the DispatcherServlet Configuration	688
Modifying the Singer List View for i18n Support	690
Using Theming and Templating	691
Theming Support	691
View Templating with Apache Tiles	693
Designing the Template Layout	693
Implementing Page Layout Components	694
Configuring Tiles in Spring MVC	698
Implementing the Views for Singer Information.....	699
Mapping URLs to the Views	699
Implementing the Show Singer View.....	700

Implementing the Edit Singer View	703
Implementing the Add Singer View.....	708
Enabling JSR-349 (Bean Validation)	709
Using jQuery and jQuery UI.....	711
Introducing jQuery and jQuery UI.....	711
Enabling jQuery and jQuery UI in a View.....	712
Rich-Text Editing with CKEditor	714
Using jqGrid for a Data Grid with Pagination	715
Enabling jqGrid in the Singer List View.....	715
Enabling Pagination on the Server Side	717
Handling File Upload	721
Configuring File Upload Support.....	721
Modifying Views for File Upload Support.....	723
Modifying Controllers for File Upload Support.....	724
Securing a Web Application with Spring Security	726
Configuring Spring Security.....	726
Adding Login Functions to the Application	729
Using Annotations to Secure Controller Methods	731
Creating Spring Web Applications with Spring Boot.....	732
Setting Up the DAO Layer	733
Setting Up the Service Layer	735
Setting Up the Web Layer	735
Setting Up Spring Security	737
Creating Thymeleaf Views	738
Using Thymeleaf Extensions.....	743
Using Webjars.....	747
Summary.....	749
■ Chapter 17: WebSocket	751
Introducing WebSocket	751
Using WebSocket with Spring	752

Using the WebSocket API	752
Using SockJS.....	760
Sending Messages with STOMP.....	765
Summary.....	772
■ Chapter 18: Spring Projects: Batch, Integration, XD, and More.....	773
Spring Batch.....	774
JSR-352.....	783
Spring Boot Batch	786
Spring Integration.....	790
Spring XD	796
Spring Framework's Five Most Notable Features	798
The Functional Web Framework	799
Java 9 Interoperability	811
JDK Modularity	811
Reactive Programming with Java 9 and Spring WebFlux.....	814
Spring Support for JUnit 5 Jupiter.....	817
Summary.....	827
■ Appendix A: Setting Up Your Development Environment.....	829
Introducing Project pro-spring-15.....	829
Understanding the Gradle Configuration	831
Building and Troubleshooting	834
Deploy on Apache Tomcat	837
Index.....	841

About the Authors

Iuliana Cosmina is a Spring-certified Web Application Developer and is also a Spring-certified Spring Professional, as defined by Pivotal, the makers of Spring Framework, Boot, and other tools. She has authored books with Apress on core Spring certification and Spring-certified web development. She is a software architect at Bearing Point Software and is an active coder and software contributor on GitHub, Stack Overflow, and more.

Rob Harrop is a software consultant specializing in delivering high-performance, highly scalable enterprise applications. He is an experienced architect with a particular flair for understanding and solving complex design issues. With a thorough knowledge of both Java and .NET, Harrop has successfully deployed projects across both platforms. He also has extensive experience across a variety of sectors, retail and government in particular. Harrop is the author of five books, including *the book you are currently reading, not at its fifth edition*, a widely acclaimed, comprehensive resource on the Spring Framework.

Chris Schaefer is a principle software developer for Spring projects at Pivotal, the makers of Spring Framework, Boot, and other Spring tools.

Clarence Ho is the senior Java architect of a Hong Kong-based software consultancy firm, SkywideSoft Technology Limited. Having been worked in the IT field for more than 20 years, Clarence has been the team leader of many in-house application development projects, as well as providing consultancy services on enterprise solutions to clients.

About the Technical Reviewer



Massimo Nardone has more than 23 years of experience in security, web/mobile development, cloud computing, and IT architecture. His true IT passions are security and Android.

He currently works as the chief information security officer (CISO) for Cargotec Oyj and is a member of the ISACA Finland Chapter board. Over his long career, he has held these positions: project manager, software engineer, research engineer, chief security architect, information security manager, PCI/SCADA auditor, and senior lead IT security/cloud/SCADA architect. In addition, he has been a visiting lecturer and supervisor for exercises at the Networking Laboratory of the Helsinki University of Technology (Aalto University).

Massimo has a master of science degree in computing science from the University of Salerno in Italy, and he holds four international patents (PKI, SIP, SAML, and proxy areas). Besides working on this book, Massimo has reviewed more than 40 IT books for different publishing companies and is the coauthor of *Pro Android Games* (Apress, 2015).

Acknowledgments

It is a huge honor for me to be the main author of the fifth edition of this book. Would you believe I got this assignment by mistake? I thought I was getting an assignment as a technical reviewer for this book. Only when I received the files did I realize that I was going to be one of the authors of the fifth edition of one of the best Spring books on the market.

Apress has published many of the books I have read and used to improve myself professionally during my studies and even after that. This is my third book with Apress, and it is great to contribute to the education of the next generation of developers.

I am grateful to all my friends who had the patience to listen to me complain about losing sleep, having too much work to do, and encountering writer's block. Thank you all for being supportive and making sure I still had some fun while writing this book.

Also, I would like to give a big thanks to all my favorite singers who made my work easier with their wonderful music, especially John Mayer; I was so determined to finish this book on time just so I could go to the United States to one of his concerts. That is why I changed the topic of the examples in this book to be about singers and their music; it's a tribute to their art and talent.

—Iuliana Cosmina

Introduction

Covering version 5 of the Spring Framework, this book is the most comprehensive Spring reference and practical guide available for harnessing the power of this leading enterprise Java application development framework.

This edition covers core Spring and its integration with other leading Java technologies, such as Hibernate, JPA 2, Tiles, Thymeleaf, and WebSocket. The focus of the book is on using Java configuration classes, lambda expressions, Spring Boot, and reactive programming. We share our insights and real-world experiences with enterprise application development, including remoting, transactions, the web and presentation tiers, and much more.

With Pro Spring 5, you'll learn how to do the following:

- Use inversion of control (IoC) and dependency injection (DI)
- Discover what's new in Spring Framework 5
- Build Spring-based web applications using Spring MVC and WebSocket
- Build Spring web reactive applications with Spring WebFlux
- Test Spring applications using Junit 5
- Utilize the new Java 8 lambda syntax
- Use Spring Boot to an advanced level to get any kind of Spring application up and running in no time
- Use Java 9 features in Spring applications

Because the Java 9 release date kept being postponed, Spring 5 was released based on Java 8. Thus, interoperability with Java 9 is covered in this book based on an early-access build.

There is a multimodule project associated with this book, configured using Gradle 4. The project is available on the Apress official repository: <https://github.com/Apress/pro-spring-5>. The project can be built right after cloning according to the instructions in its README.adoc file as long as Gradle is installed locally. If you do not have Gradle installed, you can rely on IntelliJ IDEA to download it and use it to build your project by using the Gradle Wrapper. (https://docs.gradle.org/current/userguide/gradle_wrapper.html). There is a small appendix at the end of the book describing the project structure, configuration and additional details related to development tools that can be used to develop and run the code samples of the book, which are available on GitHub.

As the book was being written, new release candidate versions of Spring 5 were released, a new version of IntelliJ IDEA was released, and new versions of Gradle and other technologies used in the book were updated. We upgraded to the new versions to provide the most recent information and keep this book synchronized with the official documentation. Several reviewers have checked the book for technical accuracy, but if you notice any inconsistencies, please send an email to editorial@apress.com and errata will be created.

You can access the example source code for this book via the Download Source Code button at www.apress.com/9781484228074. It will be maintained, synchronized with new versions of the technologies, and enriched based on the recommendations of the developers using it to learn Spring.

We truly hope you will enjoy using this book to learn Spring as much as we enjoyed writing it.

CHAPTER 1



Introducing Spring

When we think of the community of Java developers, we are reminded of the hordes of gold rush prospectors of the late 1840s, frantically panning the rivers of North America, looking for fragments of gold. As Java developers, our rivers run rife with open source projects, but, like the prospectors, finding a useful project can be time-consuming and arduous.

A common gripe with many open source Java projects is that they are conceived merely out of the need to fill the gap in the implementation of the latest buzzword-heavy technology or pattern. Having said that, many high-quality, usable projects meet and address a real need for real applications, and in the course of this book, you will meet a subset of these projects. You will get to know one in particular rather well—Spring. The first version of Spring was released in October 2002 and consisted of a small core with an inversion of control (IoC) container that was easy to configure and use. Over the years Spring has become the main replacement of Java Enterprise Edition (JEE) servers and has grown into a full-blown technology made up of many distinct projects, each with its own purpose, so whether you want to build microservices, applications, or classical ERPs, Spring has a project for that.

Throughout this book, you will see many applications of different open source technologies, all of which are unified under the Spring Framework. When working with Spring, an application developer can use a large variety of open source tools, without needing to write reams of code and without coupling an application too closely to any particular tool.

In this chapter, as its title indicates, we introduce you to the Spring Framework, rather than presenting any solid examples or explanations. If you are already familiar with Spring, you might want to skip this chapter and proceed straight to Chapter 2.

What Is Spring?

Perhaps one of the hardest parts of explaining Spring is classifying exactly what it is. Typically, Spring is described as a lightweight framework for building Java applications, but that statement brings up two interesting points.

First, you can use Spring to build any application in Java (for example, stand-alone, web, or JEE applications), unlike many other frameworks (such as Apache Struts, which is limited to web applications).

Second, the *lightweight* part of the description doesn't really refer to the number of classes or the size of the distribution but rather defines the principle of the Spring philosophy as a whole—that is, minimal impact. Spring is lightweight in the sense that you have to make few, if any, changes to your application code to gain the benefits of Spring Core, and should you choose to stop using Spring at any point, you will find doing so quite simple.

Notice that we qualified that last statement to refer to Spring Core only—many of the extra Spring components, such as data access, require a much closer coupling to the Spring Framework. However, the benefits of this coupling are quite clear, and throughout the book we present techniques for minimizing the impact this has on your application.

Evolution of the Spring Framework

The Spring Framework originated from the book *Expert One-on-One: J2EE Design and Development* by Rod Johnson (Wrox, 2002). Over the last decade, the Spring Framework has grown dramatically in core functionality, associated projects, and community support. With the new major release of the Spring Framework, it's worthwhile to take a quick look back at important features that have come along with each milestone release of Spring, leading up to Spring Framework 5.0.

- *Spring 0.9*: This is the first public release of the framework, based on the book *Expert One-on-One: J2EE Design and Development*, that offered a bean configuration foundation, AOP support, a JDBC abstraction framework, abstract transaction support, and so on. This version does not have official reference documentation, but you can find the existing sources and documentation on SourceForge.¹
- *Spring 1.x*: This is the first version released with official reference documentation. It is composed of the seven modules shown in Figure 1-1.

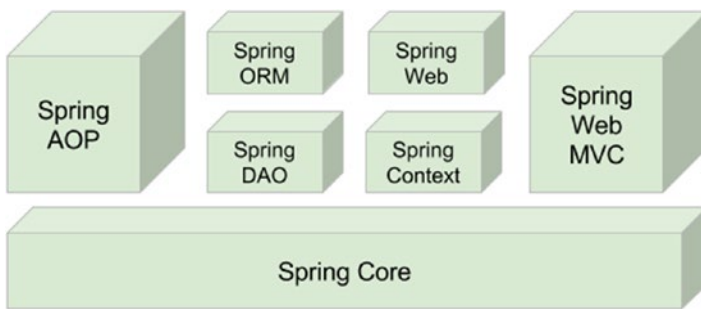


Figure 1-1. Overview of the Spring Framework, version 1.x

- *Spring Core*: Bean container and supporting utilities
- *Spring Context*: `ApplicationContext`, UI, validation, JNDI, Enterprise JavaBeans (EJB), remoting, and mail support
- *Spring DAO*: Transaction infrastructure, Java Database Connectivity (JDBC), and data access object (DAO) support
- *Spring ORM*: Hibernate, iBATIS, and Java Data Objects (JDO) support
- *Spring AOP*: An AOP Alliance-compliant aspect-oriented programming (AOP) implementation
- *Spring Web*: Basic integration features such as multipart functionality, context initialization through servlet listeners, and a web-oriented application context
- *Spring Web MVC*: Web-based Model-View-Controller (MVC) framework
- *Spring 2.x*: This is composed of the six modules shown in Figure 1-2. The Spring Context module is now included in Spring Core, and all Spring web components have been represented here by a single item.

¹You can download older versions of Spring including 0.9 from the SourceForge site: <https://sourceforge.net/projects/springframework/files/springframework/>.

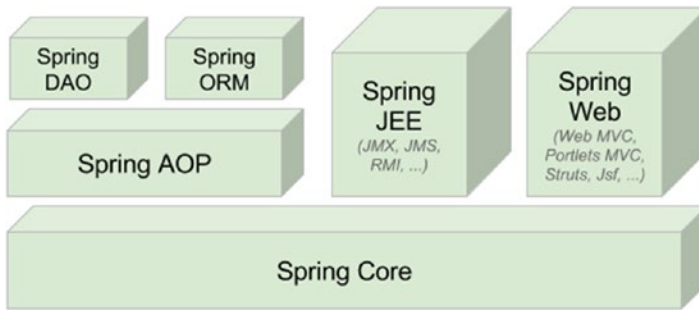


Figure 1-2. Overview of the Spring Framework, version 2.x

- Easier XML configuration through the use of the new XML Schema-based configuration rather than the DTD format. Notable areas of improvement include bean definitions, AOP, and declarative transactions.
- New bean scopes for web and portal usage (request, session, and global sessions).
- `@AspectJ` annotation support for AOP development.
- Java Persistence API (JPA) abstraction layer.
- Full support for asynchronous JMS message-driven POJOs (for plain old Java objects).
- JDBC simplifications including `SimpleJdbcTemplate` when using Java 5+.
- JDBC named parameter support (`NamedParameterJdbcTemplate`).
- Form tag library for Spring MVC.
- Introduction of the Portlet MVC framework.
- Dynamic language support. Beans can be written in JRuby, Groovy, and BeanShell.
- Notification support and controllable MBean registration in JMX.
- `TaskExecutor` abstraction introduced for scheduling tasks.
- Java 5 annotation support, specifically for `@Transactional`, `@Required`, and `@AspectJ`.
- *Spring 2.5.x:* This version has the following features:
 - A new configuration annotation called `@Autowired` and support for JSR-250 annotations (`@Resource`, `@PostConstruct`, `@PreDestroy`)
 - New stereotype annotations: `@Component`, `@Repository`, `@Service`, `@Controller`
 - Automatic classpath-scanning support to detect and wire classes annotated with stereotype annotations
 - AOP updates, including a new bean pointcut element and AspectJ load-time weaving
 - Full WebSphere transaction management support

- In addition to the Spring MVC `@Controller` annotation, `@RequestMapping`, `@RequestParam`, and `@ModelAttribute` annotations added to support request handling through annotation configuration
 - Tiles 2 support
 - JSF 1.2 support
 - JAX-WS 2.0/2.1 support
 - Introduction of the Spring TestContext Framework, providing annotation-driven and integration testing support, agnostic of the testing framework being used
 - Ability to deploy a Spring application context as a JCA adapter
- *Spring 3.0.x*: This is the first version of Spring based on Java 5 and is designed to take full advantage of Java 5 features such as generics, varargs, and other language improvements. This version introduces the Java-based `@Configuration` model. The framework modules have been revised to be managed separately with one source tree per module JAR. This is abstractly depicted in Figure 1-3.

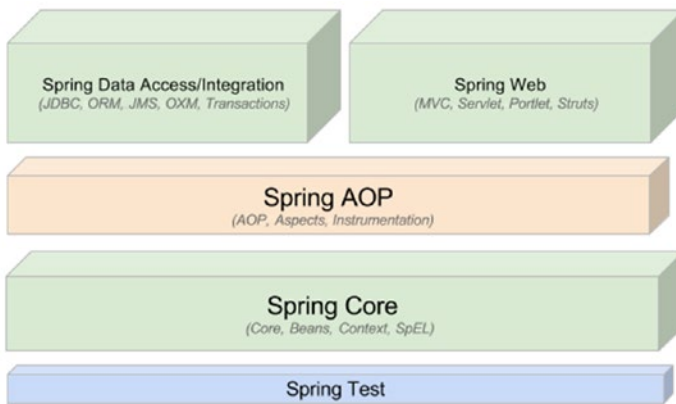


Figure 1-3. Overview of the Spring Framework, version 3.0.x

- Support for Java 5 features such as generics, varargs, and other improvements
- First-class support for Callables, Futures, ExecutorService adapters, and ThreadFactory integration
- Framework modules now managed separately with one source tree per module JAR
- Introduction of the Spring Expression Language (SpEL)
- Integration of core Java Config features and annotations
- General-purpose type conversion system and field-formatting system
- Comprehensive REST support
- New MVC XML namespace and additional annotations such as `@CookieValue` and `@RequestHeaders` for Spring MVC

- Validation enhancements and JSR-303 (Bean Validation) support
 - Early support for Java EE 6, including `@Async/@Asynchronous` annotation, JSR-303, JSF 2.0, JPA 2.0, and so on
 - Support for embedded databases such as HSQL, H2, and Derby
- *Spring 3.1.x*: This version has the following features:
 - New cache abstraction
 - Bean definition profiles can be defined in XML as well as support for the `@Profile` annotation
 - Environment abstraction for unified property management
 - Annotation equivalents for common Spring XML namespace elements such as `@ComponentScan`, `@EnableTransactionManagement`, `@EnableCaching`, `@EnableWebMvc`, `@EnableScheduling`, `@EnableAsync`, `@EnableAspectJAutoProxy`, `@EnableLoadTimeWeaving`, and `@EnableSpringConfigured`
 - Support for Hibernate 4
 - Spring TestContext Framework support for `@Configuration` classes and bean definition profiles
 - `c:` namespace for simplified constructor injection
 - Support for Servlet 3 code-based configuration of the Servlet container
 - Ability to bootstrap the JPA EntityManagerFactory without `persistence.xml`
 - `Flash` and `RedirectAttributes` added to Spring MVC, allowing attributes to survive a redirect by using the HTTP session
 - URI template variable enhancements
 - Ability to annotate Spring MVC `@RequestBody` controller method arguments with `@Valid`
 - Ability to annotate Spring MVC controller method arguments with the `@RequestPart` annotation
 - *Spring 3.2.x*: This version has the following features:
 - Support for Servlet 3–based asynchronous request processing.
 - New Spring MVC test framework.
 - New Spring MVC annotations `@ControllerAdvice` and `@MatrixVariable`.
 - Support for generic types in `RestTemplate` and in `@RequestBody` arguments.
 - Jackson JSON 2 support.
 - Support for Tiles 3.
 - `@RequestBody` or an `@RequestPart` argument can now be followed by an `Errors` argument, making it possible to handle validation errors.
 - Ability to exclude URL patterns by using the MVC namespace and Java Config configuration options.
 - Support for `@DateTimeFormat` without Joda Time.