



Pivotal Certified Professional Spring Developer Exam

A Study Guide

Examination preparation on core Spring
concepts and principles

Iuliana Cosmina

Apress®

Pivotal Certified Professional Spring Developer Exam

A Study Guide



Iuliana Cosmina

Apress®

Pivotal Certified Spring Web Application Developer Exam

Iuliana Cosmina
Sibiu, Romania

ISBN-13 (pbk): 978-1-4842-0812-0
DOI 10.1007/978-1-4842-0811-3

ISBN-13 (electronic): 978-1-4842-0811-3

Copyright © 2017 by Iuliana Cosmina

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. Exempted from this legal reservation are brief excerpts in connection with reviews or scholarly analysis or material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the work. Duplication of this publication or parts thereof is permitted only under the provisions of the Copyright Law of the Publisher's location, in its current version, and permission for use must always be obtained from Springer. Permissions for use may be obtained through RightsLink at the Copyright Clearance Center. Violations are liable to prosecution under the respective Copyright Law.

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.

Managing Director: Welmoed Spahr
Acquisitions Editor: Steve Anglin
Development Editor: Matthew Moodie
Technical Reviewer: Manuel Jordan
Coordinating Editor: Mark Powers
Copy Editor: David Kramer
Compositor: SPi Global
Indexer: SPi Global
Artist: SPi Global
Cover Image: Designed by Freepik

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 <http://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 <http://www.apress.com/us/services/bulk-sales>.

Any source code or other supplementary material referenced by the author in this book is available to readers via the book's product page, located at www.apress.com/9781484208120. For more detailed information, please visit <http://www.apress.com/source-code>.

Printed on acid-free paper

*To all passionate Java developers, never stop learning and
never stop improving your skills.*

*To all my friends for supporting me to make this book happen;
you have no idea how dear you are to me.*

Contents at a Glance

| | |
|-------------------------------------------------------------------|-------------|
| About the Author | xiii |
| About the Technical Reviewer | xv |
| Acknowledgments | xvii |
| Introduction | xix |
| ■ Chapter 1: Book Overview | 1 |
| ■ Chapter 2: Spring Bean LifeCycle and Configuration..... | 17 |
| ■ Chapter 3: Testing Spring Applications | 115 |
| ■ Chapter 4: Aspect Oriented Programming with Spring | 153 |
| ■ Chapter 5: Data Access | 185 |
| ■ Chapter 6: Spring Web..... | 271 |
| ■ Chapter 7: Spring Advanced Topics..... | 349 |
| ■ Chapter 8: Spring Microservices with Spring Cloud | 435 |
| Index..... | 461 |

Contents

| | |
|------------------------------------------------------------------|-------------|
| About the Author | xiii |
| About the Technical Reviewer | xv |
| Acknowledgments | xvii |
| Introduction | xix |
| ■ Chapter 1: Book Overview | 1 |
| What Is Spring and Why Should You Be Interested in It?..... | 1 |
| What Is the Focus of This Book? | 3 |
| Who Should Read This Book?..... | 3 |
| About the Certification Exam..... | 3 |
| How to Use This Book as a Study Guide..... | 5 |
| How Is This Book Structured? | 5 |
| How Each Chapter Is Structured..... | 6 |
| Recommended Development Environment..... | 7 |
| Recommended JVM..... | 8 |
| Recommended Project Build Tool | 8 |
| Recommended IDE | 10 |
| The Project Sample | 11 |
| ■ Chapter 2: Spring Bean LifeCycle and Configuration..... | 17 |
| Old Style Application Development | 17 |
| Spring IoC and Dependency Injection..... | 24 |
| Spring Configuration | 29 |
| Providing Configuration via XML..... | 29 |
| Spicing Up XML Configuration | 53 |

| | |
|-----------------------------------------------------------------------|------------|
| Application Context and Bean Lifecycle | 64 |
| Providing Configuration Using Java Configuration and Annotations..... | 85 |
| Summary | 110 |
| Quick quiz..... | 111 |
| ■ Chapter 3: Testing Spring Applications | 115 |
| A Few Types of Testing | 115 |
| Test-Driven Development | 115 |
| Unit and Integration Testing | 116 |
| Testing with Stubs | 117 |
| Testing with Mocks..... | 124 |
| Testing with Spring | 134 |
| Using Profiles..... | 144 |
| Summary | 146 |
| Quick Quiz | 146 |
| Practical Exercise..... | 148 |
| ■ Chapter 4: Aspect Oriented Programming with Spring | 153 |
| Problems Solved by AOP | 154 |
| Spring AOP | 157 |
| AOP Terminology..... | 158 |
| Quick Start..... | 159 |
| Aspect Support Configuration using XML..... | 165 |
| Defining Pointcuts | 165 |
| Implementing Advice | 172 |
| Conclusions | 178 |
| Summary | 181 |
| Quick Quiz | 181 |
| Practical Exercise..... | 183 |

| | |
|----------------------------------------------------------------|------------|
| ■ Chapter 5: Data Access | 185 |
| Basic Data Access Using JDBC | 187 |
| Spring Data Access | 189 |
| Introducing JdbcTemplate | 190 |
| Spring Data Access Exceptions | 207 |
| Data Access Configuration In a Transactional Environment | 209 |
| How Transaction Management Works in Spring..... | 212 |
| Configure Transactions Support..... | 214 |
| Introducing Hibernate and ORM | 235 |
| Session and Hibernate Configuration | 235 |
| Session and Hibernate Querying | 240 |
| Exception Mapping | 243 |
| Object Relational Mapping..... | 245 |
| Java Persistence API | 247 |
| Spring Data JPA..... | 256 |
| **Spring and MongoDB..... | 260 |
| Summary..... | 265 |
| Quiz | 265 |
| ■ Chapter 6: Spring Web | 271 |
| Spring Web App Configuration..... | 274 |
| Quickstart | 276 |
| XML..... | 281 |
| @MVC | 285 |
| Java Configuration for Spring MVC..... | 286 |
| Getting Rid of web.xml | 288 |
| Running a Spring Web Application..... | 291 |
| Running with Jetty..... | 292 |
| Running with Tomcat | 294 |

| | |
|--------------------------------------------------|------------|
| Spring Security | 298 |
| Spring Security Configuration..... | 301 |
| XML Configuration | 301 |
| Spring XML Configuration without web.xml | 313 |
| Java Configuration..... | 313 |
| Security Tag Library..... | 317 |
| Method Security | 321 |
| Spring Boot | 326 |
| Configuration | 327 |
| Configuration Using YAML..... | 338 |
| Logging..... | 341 |
| Testing with Spring Boot..... | 341 |
| Summary | 344 |
| Quiz | 345 |
| ■ Chapter 7: Spring Advanced Topics | 349 |
| Spring Remoting | 350 |
| Spring Remote Configuration | 353 |
| Spring JMS | 362 |
| JMS Connections and Sessions..... | 363 |
| JMS Messages | 364 |
| JMS Destinations..... | 365 |
| Apache ActiveMQ..... | 367 |
| Spring JmsTemplate..... | 370 |
| JMS with Spring Boot..... | 378 |
| Spring Web Services | 382 |
| SOAP Messages..... | 384 |
| Generating Java Code with XJC..... | 386 |
| Spring Boot WS Application | 387 |
| Publishing WSDL..... | 391 |
| Testing Web Services applications | 392 |

| | |
|------------------------------------------------------------------|------------|
| Spring REST | 395 |
| Spring Support for REST | 397 |
| Exception Handling | 402 |
| HTTP Message Converters | 404 |
| Spring MVC Configuration for RESTful Applications | 405 |
| Using RestTemplate to Test RESTful Applications | 407 |
| Advantages of REST | 416 |
| Spring JMX | 421 |
| JMX Architecture | 421 |
| Plain JMX | 423 |
| Spring JMX | 424 |
| Summary | 432 |
| Quick Quiz | 433 |
| ■ Chapter 8: Spring Microservices with Spring Cloud | 435 |
| Microservices with Spring | 436 |
| Registration and Discovery Server | 439 |
| Microservices Development | 442 |
| Microservices Communication | 451 |
| More Novelties | 456 |
| Practice Section | 457 |
| Summary | 458 |
| Quick Quiz | 458 |
| Index | 461 |

About the Author



Iuliana Cosmina is a software architect and passionate developer. She has been programming in Java for more than 10 years. She also taught Java at the Gheorge Asachi Technical University in Iasi, Romania. She has a bachelor's degree in computer science and a master's degree in distributed systems from the same university.

She discovered Spring in June 2012 and loved it so much that she trained for and passed the exam to become a Certified Spring Professional in November 2012. She trained for and passed the exam to become a Certified Web Application Developer in May 2014.

Her plan is to become a Spring Enterprise Integration Specialist in the near future.

She has contributed to the development of different types of enterprise applications such as search engines, ERPs, track and trace, and banking. During her career in outsourcing she has been a team leader,

acting software architect, and DevOps professional. She likes to share her knowledge and expertise via tutoring, teaching, and mentoring, but in the summer of 2014, everything changed because of Steve Anglin, who proposed that she write a Spring Web Study Guide. A short time after the first book was released, she was given the chance to write the Spring Core Study Guide as well, and she seized the opportunity. She currently lives in Sibiu, Romania, and works as a software architect for BearingPoint, a multinational management and technology consulting company.

When she is not programming, she spends her time reading, traveling, hiking, or biking.

- You can find some of her personal work on her GitHub account: <https://github.com/iuliana>.
- You can find her complete CV on her LinkedIn account: <https://ro.linkedin.com/in/iulianacosmina>.
- You can contact her at Iuliana.Cosmina@gmail.com.

About the Technical Reviewer

Manuel Jordan is a self-taught developer and researcher who enjoys learning new technologies for his own experiments in creating new integrations among them.

Manuel won the 2010 Springy Award, Community Champion and Spring Champion 2013. In his little free time, he reads the Bible and composes music on his bass and guitar.

Acknowledgments

Creating this guide involved a lot of teamwork. It is the second time I've written a technical book, and I wouldn't have made it without all the help and advice I received from Mark Powers and Manuel Jordan. Mark has been very supportive, sharing with me his experience in book writing and encouraging me when I was ready to give up because I thought my work was not good enough. He was also very understanding and forgiving when deadlines were missed because of writer's block or personal problems.

Manuel has been a great collaborator; I loved our exchanges of technical ideas, for which I am very thankful, because working with them has helped me grow professionally. Many thanks to the team that helped turn my technical verbiage into human-readable literature.

Most of all, I want to thank Steve Anglin for trusting me to get this book done.

Apress has published many of the books I have read and used to improve myself professionally during my studies and beyond. It is a great honor for me to write a book and publish it with Apress, and it gives me enormous satisfaction to be able 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 sleep loss, having too much work to do, and writer's block. Thank you all for being supportive and making sure I still had some fun while writing this book.

And I would also like to add a very special thank you to Marian Lopatnic, Cristina Lutai, and Andreea Jugarean. These three special persons ensured that my determination to finish this book never flagged, by continually reminding me that I am a badass in my profession, and as long as I do my best, the outcome will be great.

Introduction

More than four years have passed since I wrote my first Spring project, and since then, the Spring Framework has turned into a full-blown technology that provides everything needed to build complex and reliable Java Enterprise Applications.

Four major versions of Spring have been released so far, and the fifth is right around the corner. And except for the official study guide required for passing the certification exam, before the conception of this book there was no additional resource like this.

This study guide provides a complete overview of all the technologies involved in creating a Spring core application from scratch. It guides you step by step into the Spring world, covering Spring 3 and Spring 4. More advanced topics such as RMI and JMS have been covered as well, because there are still companies that prefer to use them, and developers might encounter them while in the field.

There is a multimodule project associated with this book named Pet Sitter, covering every example presented in the book. As the book was written, new versions of Spring were released, a new version of IntelliJ IDEA was released, and new versions of Gradle were released as well. I upgraded to the new versions in order to provide the most recent information and keep this book synchronized with the official documentation. A group of reviewers has gone over the book, but if you notice any inconsistencies, please send an email to editorial@apress.com, and a correction will be made.

The example source code for this book can be found on GitHub via the **Download Source Code** button on the book's product page, located at www.apress.com/9781484208120. 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.

The code for the Pet Sitter project will likewise be made available on a public GitHub repository.

An appendix with answers to the questions at the end of every chapter and additional details related to development tools that can be used to develop and run the code samples of the book will also be available as part of the source code package hosted at Github. A sample practice exam will also be published on the Pet Sitter repository.

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

CHAPTER 1



Book Overview

Spring is currently one of the most influential and rapidly growing Java frameworks. Every time a new startup idea is born, if the development language is Java, Spring will be taken into consideration. Spring will be fourteen years old on the first of October 2016, and it has grown into a full-fledged software technology over the years.¹

This book covers much of the core functionality from multiple projects. The topics that are required for the official certification exam are covered deeply, and all extras are covered succinctly enough to give you a taste and make you curious to learn more.

What Is Spring and Why Should You Be Interested in It?

When a project is being built using Java, a great deal of functionality needs to be constructed from scratch. Yet many useful functionalities have already been built and are freely available because of the open source world we are living in. A long time ago, when the Java world was still quite small, when you were using open source code in your project developed by somebody else and shipped as a **.jar*, you would say that you were using a **library**. But as time passed, the software-development world evolved, and the libraries grew too. They became **frameworks**. Because they were no longer a single **.jar* file that you could import, they became a collection of more-or-less decoupled libraries, with different responsibilities, and you had the possibility of importing only what you needed. As frameworks grew, tools to build projects and add frameworks as dependencies evolved. One of the currently most widely used tools to build projects is Maven, but new build tools are now stealing the scene. One of these new-age build tools will be used to build the projects for this book. It will be introduced later.

The Spring Framework was released in October 2002 as an open source framework and inversion of a control container developed using Java. As Java evolved, Spring did too. Spring version 4.3, the one covered in this book, is fully compatible with Java 8, and Spring 5 is planned to be released in the fourth quarter of 2016.² The intention is to make it compatible with Java 9, which is planned for release in September 2016.³ But since the release of Java 8 was delayed over six months, nothing is certain at the moment.

Spring comes with a great deal of default behavior already implemented. Components called “infrastructure beans” have a default configuration that can be used or easily customized to fit the project’s requirements. Having been built to respect the “Convention over Configuration” principle, it reduces the number of decisions a developer has to make when writing code, since the infrastructure beans can be used to create functional basic applications with minimum customization (or none at all) required.

¹Just as a coincidence and a fun fact, in Romania you obtain your first identity card when you are fourteen years old, and it is considered the age of intellectual maturity that allows you to differentiate good from evil.

²Information can be found on the official Spring blog at <https://spring.io/blog/2015/08/03/coming-up-in-2016-spring-framework-4-3-5-0>.

³Information at <https://jaxenter.com/java-9-release-date-announced-116945.html>.

Spring is open source, which means that many talented developers have contributed to it, but the last word on analyzing the quality of the components being developed belongs to the Pivotal Spring Development Team, previously known as the SpringSource team, before Pivotal and VMware merged. The full code of the Spring Framework is available to the public on Github,⁴ and any developer that uses Spring can fork the repositories and propose changes.

A Java application is essentially composed of objects talking to each other. The reason why Spring has gained so much praise in Java application development is that it makes connecting and disconnecting objects easy by providing a comprehensive infrastructure support for assembling objects. Using Spring to develop a Java application is like building a lightly connected Lego castle; each object is a Lego piece that you can easily remove and replace with a different one. Spring is currently the VIP of Java Frameworks, and if all you have read so far has not managed to make you at least a bit interested in it, then I am doing a really bad job at writing this book, and you should write an email and tell me so.

Before going further about what this book will provide you, let’s have an overview of the Spring projects. Figure 1-1 depicts all the Spring projects. For 2016, there are nineteen main projects, two community projects, and three projects that are “in the attic,” so to speak, because there will be no further contributions to them, since they are going to be dropped in the future.

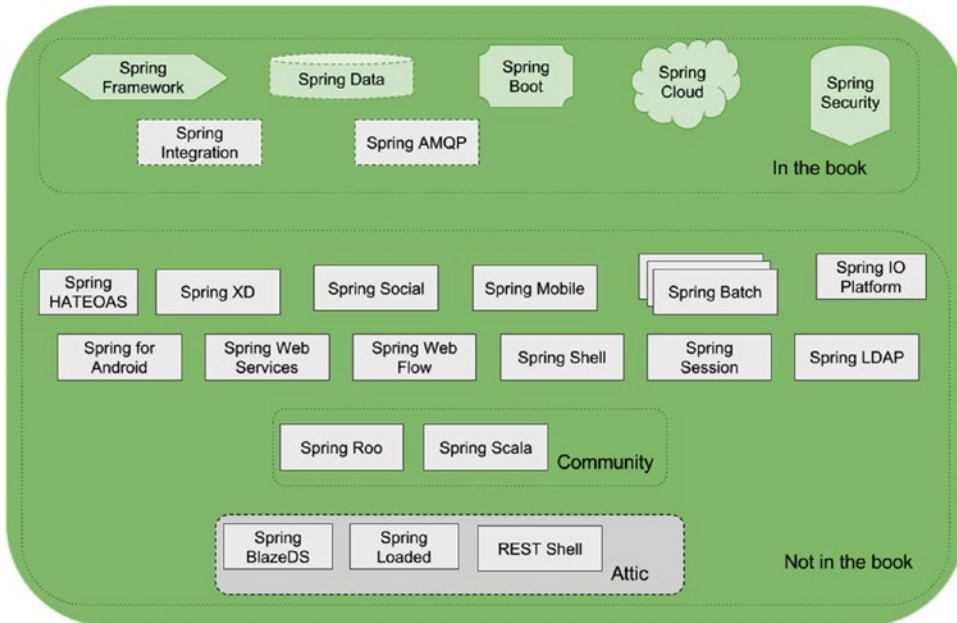


Figure 1-1. Spring Web Stack. (The projects drawn with dotted contours will be covered only partially in this book.)

⁴Github Spring Framework sources: <https://github.com/spring-projects/spring-framework>.

What Is the Focus of This Book?

The topics covered in this book are mostly Spring Framework's support components for the backend tier. We only scratch the surface for Spring Boot, Spring Data JPA, REST, MVC, and Microservices. This book aims to provide a natural path in the development of a complete Spring application. With each chapter, the application will become more complex, until its final form is reached, which will also have a security setup, a simple web application, and will support REST requests.

This book is focused on helping developers understand how Spring's infrastructure was designed and how to write Spring applications in a few easy steps using the maximum of Spring's potential. Its objectives are as follows:

- Use Spring to develop applications
- Use Spring Security to secure resources
- Use Spring Test and other test frameworks (JUnit, JsMock) to test applications
- Create Spring applications using Gradle⁵

Who Should Read This Book?

This book was written to provide clear insight into creating applications using Spring core components. It can also be a big help to a developer who wants to become a **Certified Spring Professional**.⁶ That is why every topic that is found in the official Pivotal Spring Core study guide is given the attention it deserves.

You just need a minimal knowledge of Java in order to make good use of this book, but online documentation for Java⁷ and Spring⁸ should be consulted every time something is not fully covered in the book.

In a nutshell, this book was written to be used by the following audiences:

- Java developers who want a taste of Spring
- Spring developers who are interested in learning to use Spring proficiently, but not interested in official certification
- Spring and Java developers who want to become certified and want all the help they can get.

About the Certification Exam

If you are interested in becoming a **Certified Spring Professional**, the first step you have to take is to go to the Pivotal official learning site <http://pivotal.io/training> and search for the Spring Certification section. There you will find all the details you need regarding the official trainings, including where are they

⁵Gradle is an automated build tool that is easy to configure and use for any type of application. Its build files are written using Groovy. Gradle combines the power and flexibility of Ant with the dependency management and conventions of Maven into a more effective way to build. Read more about it at <https://www.gradle.org/>.

⁶Keep in mind that attending a Spring Web training from Pivotal or a VMware Authorized Training Center is a prerequisite to becoming a Certified Spring Professional, as stated on the official site: <http://pivotal.io/academy#certification>.

⁷JSE8 official reference: <http://docs.oracle.com/javase/8/docs/>; JEE7 official documentation: <http://docs.oracle.com/javaee/7/>.

⁸Spring official Javadoc: <http://docs.spring.io/spring/docs/current/javadoc-api/>; Spring Reference: <http://docs.spring.io/spring/docs/current/spring-framework-reference/>.

taking place and when. The training is four days long. There are online trainings available as well. After creating an account on the Pivotal site, you can select the desired training. After you make the payment, if you choose an online training, about a month later, you will receive through the mail an official training kit consisting of the following:

- A pair of conference headphones (usually Logitech) for use during the training to hear your trainer talk and so you can ask questions.⁹
- A professional webcam (usually Logitech) for use during the training, so that your trainer and colleagues can see you, thus simulating the classroom experience.¹⁰
- A Spring study guide containing the printed version of the slides your tutor will be using during the training. (This might also be in electronic form, from consideration of the environment.)
- A Spring Study Lab book containing explanations and instructions for the practical exercises you will do during the training. (This might also be in electronic form, from consideration of the environment.)
- A Pivotal official flash drive containing the following:
 - Jdk installer
 - Sources necessary during the training. Each study lab has a small Spring application attached to it with missing configuration and code, and the student’s task is to complete it in order to have a working application. The same model is used in the code associated with this book.
 - An installer of the most recent stable version of the Spring Tool Suite. The version on the flash drive is mandatory for the course, because the installer sets up a local Maven repository with all the needed dependencies and a full eclipse project configuration with the lab sources. The STS also has an internal tcServer to run the web lab applications.
 - An html or PDF version of the Spring Study Lab.

If you decide against an online training course, you will not receive the headphones and the webcam. The training kit and the rest of the materials will be given to you when you arrive at the location where the training is taking place. After the training, you will receive a free voucher that is required to schedule the certification exam at an approved exam center near you. Basically, the voucher or voucher code being given to you is the proof that you have attended the official Spring Web Training.

! The exam duration is ninety minutes and consists of fifty questions. There are both single and multiple-answer questions. The latter are quite explicit, telling you how many correct answers you are expected to select. The questions in the book are actually more difficult, because you will not be told the number of correct options you must select. But you will be given a complete explanation of the answers in the appendix.

⁹Depending on the area and the training center, this item is optional.

¹⁰Depending on the area and the training center, this item, too, is optional.

The questions will cover (approximately) the following topics:

- Spring overview container, IoC and dependency injection
- SpEL and Spring AOP
- Spring JDBC, Transactions, ORM
- Spring MVC and the web layer
- Spring Security
- Spring Messaging and REST
- Spring Testing

The passing score for the exam is **76%**. This means that **38** correct answers are needed in order to pass. Most of the questions will present you a piece of Java code or configuration and ask you what it does, so make sure you understand the code attached to this book and write your own beans and configurations in order to understand the framework better. The good news is that all the code in the exam can be found in the sources you are given when you attend the official training. Other questions will present you with assertions about Spring Web and will require you to select the correct or the invalid statement.

If you read this book, understand all the examples, solve the practice exercises, and then attend the official training, my recommendation is that you take the certification exam as soon as possible. Do not allow for too much time to pass between finishing the training and taking the exam, because we are all human after all and information can be forgotten. Also, **the certification voucher is valid for only a year**. You can retake the exam again if you fail the first time, but it will cost you about \$150.

How to Use This Book as a Study Guide

This book was written in such a way as to guide you step by step through the wonderful technology that is Spring. It follows the same learning curve as the official training and focuses on the same topics that are required for the certification exam, since those are also the most needed in real production applications. The topics that are not needed for the certification exam are marked, so you know that you can skip them, although if you are truly interested in Spring, you will definitely not do so.

The main differences are in the tools used for the practical examples, which will be covered shortly.

How Is This Book Structured?

This book has eight chapters and an appendix. The official Spring guide has sixteen chapters, but for the purposes of the book, related topics are wrapped up together. For example, the official study guide has four separate chapters that cover dependency injection, Spring Core fundamentals, and configuration. This book has only a single big chapter about these topics: **Chapter 2—Bean LifeCycle and Configuration**.

The list of chapters and a short description of each are presented in Table 1-1.

Table 1-1. List of topics by chapter

| Chapter | Topic | Details |
|---------|----------------------------------------|----------------------------------------------------------------------------------------|
| 1 | Book overview | Introduction to Spring history, technologies and tools used for practice |
| 2 | Bean LifeCycle and Configuration | Basic Spring core concepts, components, and configuration |
| 3 | Testing Spring Applications | How Spring applications can be tested, most used testing libraries and test principles |
| 4 | Aspect Oriented Programming | AOP concept, problems that it solves and how it is supported in Spring |
| 5 | Data Access | Advanced Spring Data access using JDBC, Hibernate and Spring Data JPA |
| 6 | Spring Web | Basic introduction of Spring MVC |
| 7 | Spring Advanced Topics | Remoting, Messaging, Web Services with REST |
| 8 | Spring Microservices with Spring Cloud | Introduction to Spring Microservices and what they can be used for |
| A | Appendix | Two mock exams, answers to review questions, and other comments |

How Each Chapter Is Structured

The introductory chapter, the one you are reading now, covers the basics of Spring that every developer using this book should know: what Spring is, how it has evolved, how many official Spring projects there are, the technologies used to build and run the practical exercises, how you can register for the exam to become a Certified Spring Professional, and so on. This chapter is the exception. It is structured differently from the others, because it was designed to prepare you for what it will be coming next.

The remaining chapters are designed to cover a Spring Module and associated technologies that will help you build a specific type of Spring application. Each chapter is split into a few sections, but in a nutshell, a chapter is organized as follows:

- Basics
- Configuration
- Components
- Summary
- Quick quiz
- Practical exercise

The longer chapters deviate from this structure, introducing small practice exercises after key sections, since solving these exercises will help you to check your understanding and solidify your knowledge of the presented components.

Code that is irrelevant to Spring understanding will not always be quoted in this book, but it is available to you in the book's practice project.

Conventions

! This symbol appears in front of paragraphs to which you should pay particular attention.

** This symbol appears in front of a paragraph that is an observation or an execution step that you can skip.

? This symbol appears in front of a question for the user.

... This symbol represents missing code that is not relevant for the example.

CC This symbol appears in front of a paragraph that describes a **Convention over Configuration** practice in Spring, a default behavior that helps the developer reduce his or her work.

[random text here] When you have text surrounded by square brackets, this means that the text between the brackets should be replaced by a context-related notion.

Downloading the Code

This book has code examples and practical exercises associated with it. There will be missing pieces of code that you will have to fill in to make applications work and test your understanding of Spring Web. I recommend that you go over the code samples and do the exercises, since similar pieces of code and configurations will appear in the certification exam.

The following downloads are available:

- Source code for the programming examples in the practice section using XML configuration
- Source code for the programming examples in the practice section using Java configuration

You can download these items from the Source Code area of the Apress website <http://www.apress.com>.

Contacting the Author

More information about Iuliana Cosmina can be found at <http://ro.linkedin.com/in/iulianacosmina>. She can be reached at <mailto:iuliana.cosmina@gmail.com>.

Follow her personal coding activity on <https://github.com/iuliana>.

Recommended Development Environment

If you decide to attend the official course, you will notice that the development environment recommended in this book differs considerably from the one used at the course. A different editor was recommended, a different application server, and even a different build tool. The reason for this was to improve and expand your experience as a developer and to offer a practical development infrastructure. Motivation for each choice will be mentioned on the corresponding sections.

Recommended JVM



Java 8, the official JVM from Oracle. Download the JDK matching your operating system from

<http://www.oracle.com> and install it.

! It is recommended that you set the `JAVA_HOME` environment variable to point to the directory where Java 8 was installed (the directory in which the JDK was unpacked) and add `%JAVA_HOME%\bin` for Windows, `$JAVA_HOME/bin` for Unix-based operating systems, to the general path of the system. The reason behind this is to ensure that other development applications written in Java will use this version of Java and prevent strange incompatibility errors during development.


! Verify that the version of Java the operating system sees is the one you just installed by opening a terminal (Command Prompt in Windows, and any type of terminal you have installed on MacOs and Linux) and typing:

```
java -version
```

You should see something similar to this:

```
java version "1.8.0_74"
Java(TM) SE Runtime Environment (build 1.8.0_74-b02)
Java HotSpot(TM) 64-Bit Server VM (build 25.74-b02, mixed mode)
```

Recommended Project Build Tool

 **Gradle 2.x** ** The sources attached to this book can be compiled and executed using the Gradle wrapper, which is a batch script on Windows and a shell script for other operating systems. When you start a Gradle build via the wrapper, Gradle will be automatically downloaded and used to run the build; thus you do not need to install Gradle as stated previously. Instructions on how to do this can be found by reading the public documentation at http://www.gradle.org/docs/current/userguide/gradle_wrapper.html.

A good practice is to keep code and build tools separate, but for this study guide, it was chosen to use the wrapper to make setting up the practice environment easy by skipping the Gradle installation step and also because the recommended source code editor uses the wrapper internally.

If you decide to use Gradle outside the editor, you can download the binaries only (or if you are curious, you can download the full package, which contains binaries, sources, and documentation) from their official site <https://www.gradle.org/>, unpack them, and copy the contents somewhere on the hard drive. Create a `GRADLE_HOME` environment variable and point it to the location where you have unpacked Gradle. Also add `%GRADLE_HOME%\bin` for Windows, `$GRADLE_HOME/bin` for Unix-based operating systems, to the general path of the system.

Gradle was chosen as a build tool for the sources of this book because of the easy setup, small configuration files, flexibility in defining execution tasks, and the fact that the Pivotal Spring team currently uses it to build all Spring projects.

! Verify that the version of Gradle the operating system sees is the one you just installed by opening a terminal (Command Prompt in Windows, and any type of terminal you have installed on MacOS and Linux) and typing

```
gradle -version
```

You should see something similar to this:

```
-----  
Gradle 2.11  
-----
```

```
Build time:    2016-02-08 07:59:16 UTC  
Build number: none  
Revision:     584db1c7c90bdd1de1d1c4c51271c665bfcb978  
  
Groovy:       2.4.4  
Ant:          Apache Ant(TM) version 1.9.3 compiled on December 23 2013  
JVM:         1.8.0_74 (Oracle Corporation 25.74-b02)  
OS:          -- whatever operating system you have --
```

The text above being displayed is confirmation that Gradle commands can be executed in your terminal; thus Gradle was installed successfully.

The reason Gradle was used to build the projects for this book is its simplicity. Gradle was presented recently as the modern open source polyglot build automation system, and the Gradle team now also helps you analyze your builds in order to prove it. Gradle now offers the possibility of registering a receipt on their site that will be used to connect to the site and generate the build statistics. The project of this book will use this new service provided by the Gradle team to keep you informed on how the project grows with each chapter.

If you enter <https://gradle.com/demo/> and follow the instructions in the demo, you can build your project and get a set of statistics about how healthy your project and team are. In the first step of conception, the project is quite simple, so the initial automatic build does not provide much information. In Figure 1-2, you can see the statistics for the project in the initial phase. As you can see, the build takes one second, no tests are run, and the JVM needed a maximum of 910 MB of memory to run this build.



Ran for 1 sec
 Started today at 11:43:59 PM +02:00
 Gradle 2.11

1 sec

+02:00 Started on **Feb 17, 2016** at **11:43:59 PM** Finished on **Feb 17, 2016** at **11:44:00 PM**

EET Started on **Feb 17, 2016** at **11:43:59 PM** Finished on **Feb 17, 2016** at **11:44:00 PM**

| | | |
|---------------------------------------------------|--------|-----|
| :00-ps-core:compileJava UP-TO-DATE | 226 ms | 91% |
| :00-ps-core:processResources UP-TO-DATE | 5 ms | 2% |
| :00-ps-core:jar UP-TO-DATE | 4 ms | 1% |
| :01-ps-start-practice:processResources UP-TO-DATE | 3 ms | 1% |
| :01-ps-start-practice:jar UP-TO-DATE | 3 ms | 1% |
| :00-ps-core:processTestResources UP-TO-DATE | 1 ms | |
| :00-ps-core:test UP-TO-DATE | 1 ms | |
| :01-ps-start-practice:classes UP-TO-DATE | 1 ms | |

See all tasks

| | |
|------------------------|-----|
| Daemon ⓘ | On |
| Parallel ⓘ | Off |
| Refresh dependencies ⓘ | Off |
| Re-run tasks ⓘ | Off |
| Continuous ⓘ | Off |

See 4 more inactive switches

| | | |
|---------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------|
|  Java 1.8 |  Mac OS X 10.10.5 |  8 cores |
|  JVM max mem 910 MB |  8 max workers |  English (United States) |

Figure 1-2. Gradle.com build statistics

Recommended IDE



The recommended IDE to use in this study guide is IntelliJ IDEA. The reason for this is that it is the most intelligent Java IDE. IntelliJ IDEA offers outstanding framework-specific coding assistance and productivity-boosting features for Java EE, and Spring also includes support for Maven and Gradle. It is the

perfect choice to help you focus on learning Spring, and not how to learn to use an IDE. It can be downloaded from the JetBrains official site <https://www.jetbrains.com/idea/>. It is also light on your operating system and quite easy to use.

Since Spring Boot will be used to run the web applications in the project attached to the book, you can use the community edition in order to build and solve the TODOs in the project. But if you are looking for a professional experience in development with Java and Spring, you can try working with the Ultimate Edition, which has a trial period of thirty days. The figures with code being run, launchers being created, and other IDE-related details in this book are made using an IntelliJ IDEA Ultimate version.

I believe that an IDE should be so easy to use and so intuitive that you can focus on what really matters: the solution you are implementing. But in case you are already familiar with a different Java editor, you can go ahead and use it as long as it supports Gradle.

The Project Sample

The project attached to this book is called **Pet Sitter**. As you have probably figured out, it is a proof of concept for an application designed to help pet owners find people to take care of their pets while they are on vacation or are forced to leave the pet alone for some reason. Here is what this project should provide:

- A user should have a secured account to access the application. The type of the account can be:
 - OWNER = user that is only looking for a pet sitter for its pet(s)
 - SITTER = user that is only looking to provide pet sitter services
 - BOTH = both of the above
 - ADMIN = account with special privileges that can manage other users' activities on the site.
- A user account of type OWNER can have one or more pet instances associated with it.
- Each pet must have an RFID¹¹ microchip implanted, and the barcode should be provided to the application.
- A user account of type OWNER is able to create a request for a pet sitter for more than one interval.
- A user account of type SITTER can reply to requests by creating response objects that will be approved or rejected by the owner of that request.
- A user account of type BOTH can act as an OWNER and as a SITTER.
- Admin accounts can deactivate other types of accounts for inactivity.
- Pet sitters and owners can rate each other by writing a review of their experience. The results are stored in a rating field attached to the user account.

The project is a multimodule Gradle project. Every module is a project that covers a specific Spring Topic. The projects suffixed with `practice` are missing pieces of code and configuration and are the ones that need to be solved by you to test your understanding of Spring Web. The projects suffixed with `solution` are a proposal resolution for the tasks. Some projects are suffixed with `sample` to tell you that they contain a sample of code or configuration that you are to analyze and pay special attention to.

In Figure 1-3, the structure of the Pet Sitter project as it is viewed in IntelliJ IDEA is depicted. Each module name is prefixed with a number, so no matter what IDE you use, you will always have the modules in the exact order in which they were intended to be used.

¹¹Radio-frequency identification (RFID) uses electromagnetic fields to automatically identify and track tags attached to an object, or pet in this case.

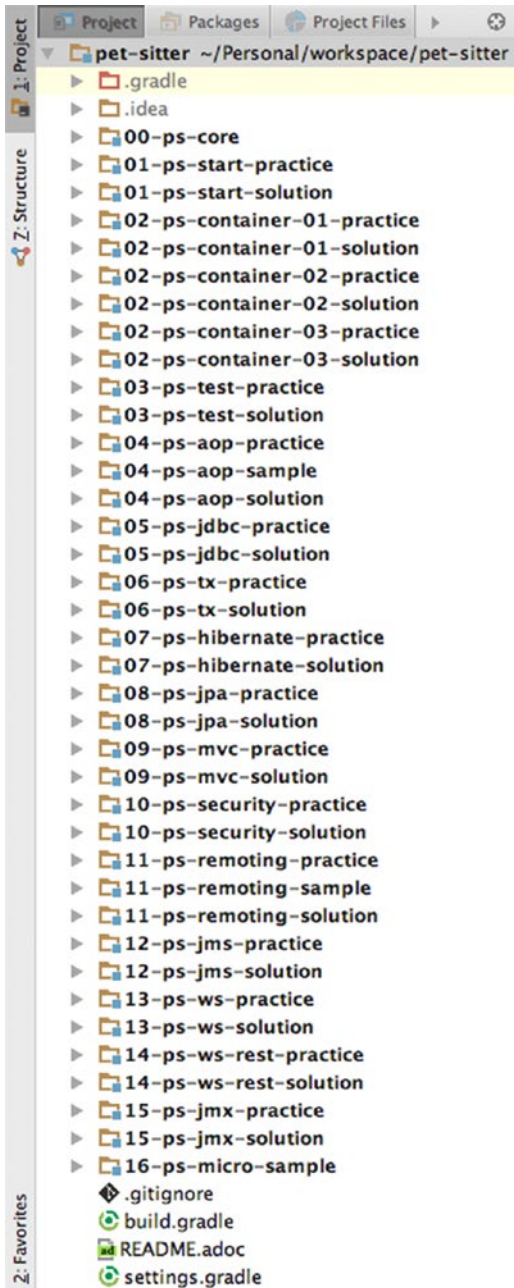


Figure 1-3. Pet Sitter modules

The **00-ps-core** contains the entity classes that map on database tables, enumerations, and other utility classes that are referenced from other modules. As the name of the project implies, this is the core project, the base tier. The other projects are implementation of service tiers that are built upon it. The Pet Sitter was designed with the multitier architecture in mind, and the abstract internal layer structure is depicted in Figure 1-4.

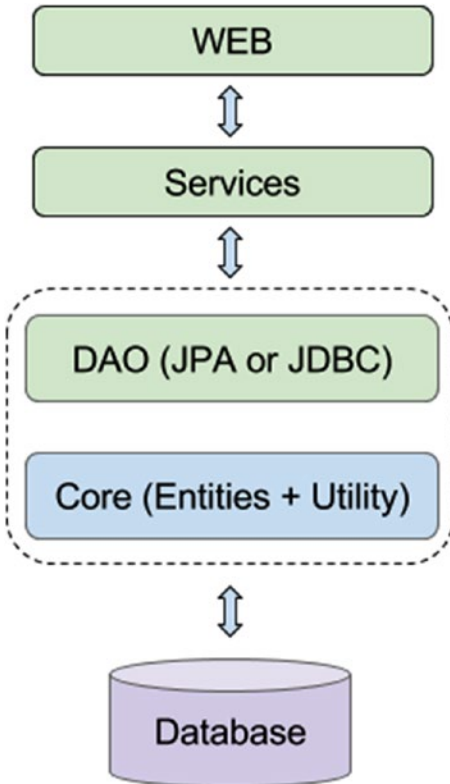


Figure 1-4. Pet Sitter application layers

The entities have common fields used by hibernate to identify uniquely each entity instance (`id`) and fields used to audit each entity instance (`createdAt` and `modifiedAt`) and keep track of how many times an entity was modified (`version`). These fields have been grouped in the `AbstractEntity` class to avoid having duplicated code. Other classes are enumerations used to define different types of objects and other utility classes (for conversion and serialization). The contents of the **00-ps-core** project are depicted in Figure 1-5.

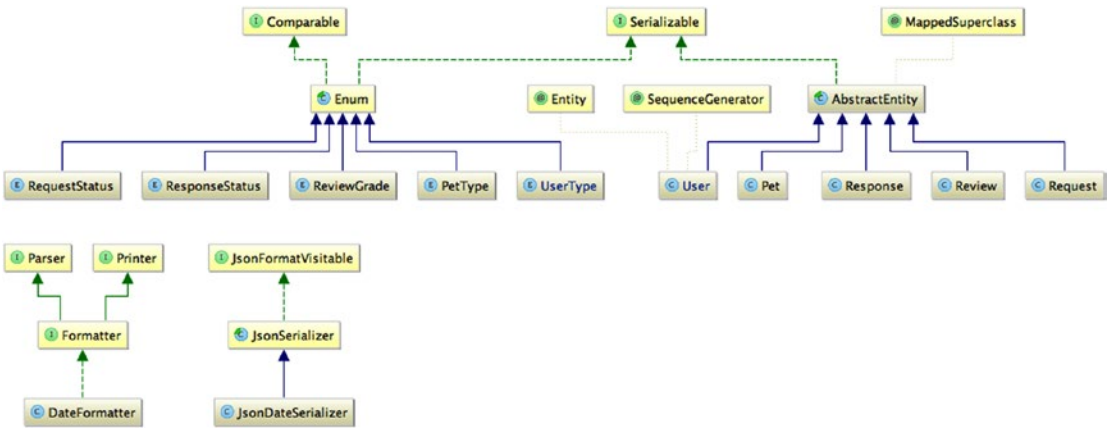


Figure 1-5. Pet Sitter 00-ps-core project contents

The class hierarchy, class members, and relationships between classes can be analyzed in Figure 1-6.

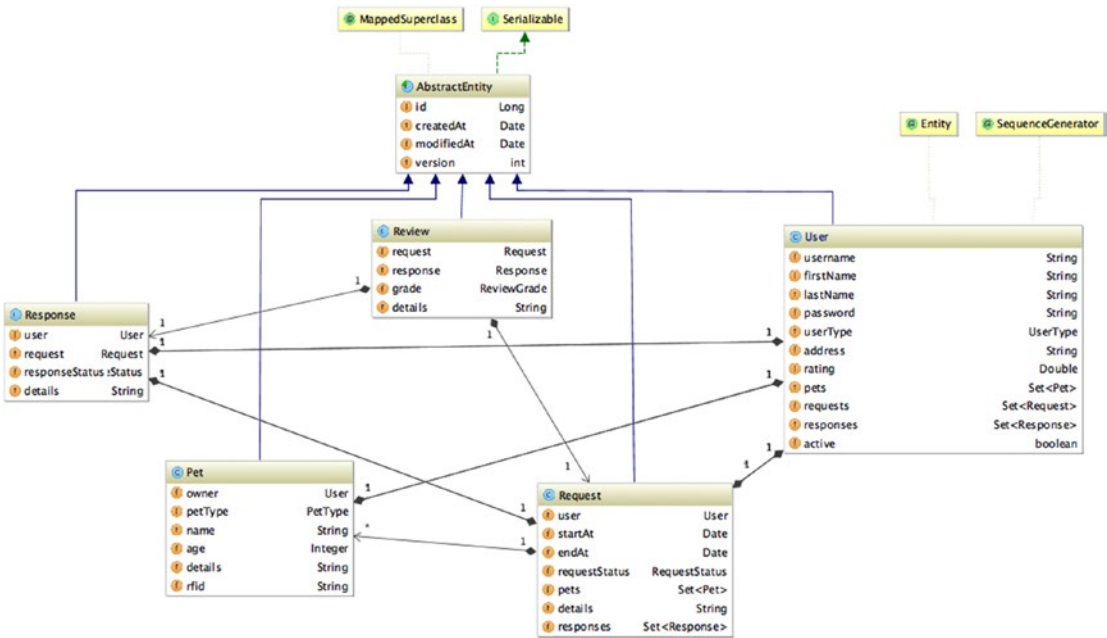


Figure 1-6. Pet Sitter entity class hierarchy

The UML diagram in Figure 1-7 describes the general functionality of the application. The RequestDispatcher and Controller are part of the web tier and are included here because the **09-ps-mvc-*** and **10-ps-security-*** projects also have a simple web tier in place, and basic notions of Spring Web are part of the certification exam.

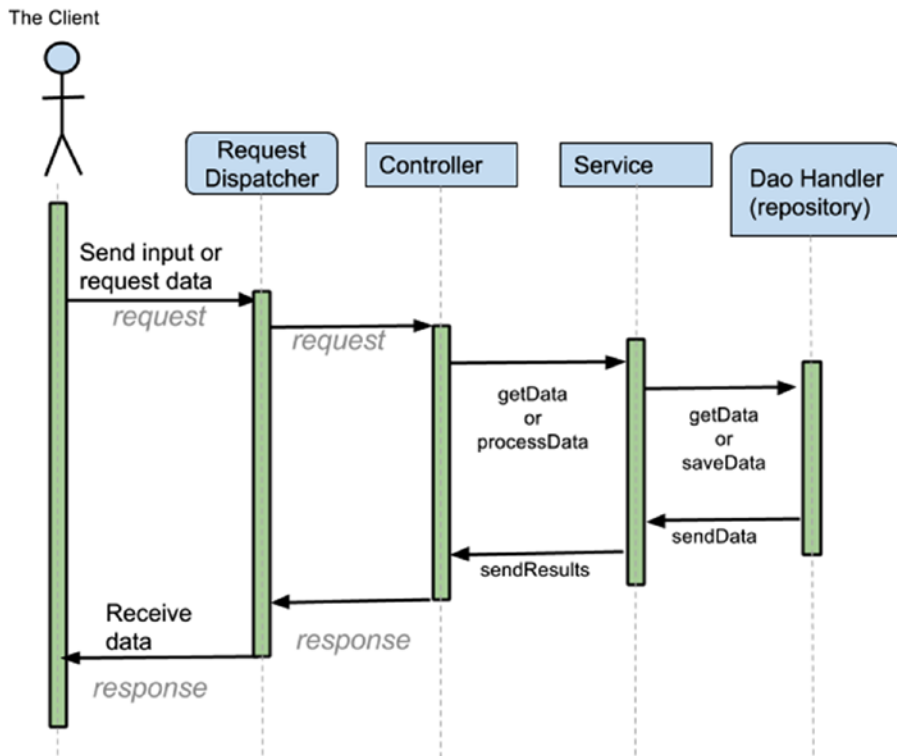


Figure 1-7. UML diagram describing the general behavior of the Pet Sitter application

This chapter does not have any practice and sample code attached to it, so more information regarding the setup of the project and how it is built and executed will be provided in the following chapters.

CHAPTER 2



Spring Bean LifeCycle and Configuration

The Spring Framework provides an easy way to create, initialize, and connect objects into competent, decoupled, easy to test enterprise-ready applications. Every software application consists of software components that interact, that collaborate and depend on other components to successfully execute a set of tasks. Each software component provides a service to other components, and linking the customer and the provider component is the process known as *Dependency Injection*. Spring provides a very simplistic way to define the connections between them in order to create an application.

Before Spring entered the picture, defining connections between classes and composing them required for development to be done following different design patterns, such as *Factory*, *Abstract Factory*, *Singleton*, *Builder*, *Decorator*, *Proxy*, *Service Locator*, and even reflection.¹ Spring was built in order to make *Dependency Injection* easy. This software design pattern implies that clients delegate the dependency resolution to an external service. The client is not allowed to call the injector service, which is why this software pattern is characterized by *Inversion of Control* behavior, also known as the *Don't call us, we'll call you!* principle.

The software components that Spring uses to build applications are called *beans* and are nothing more than *Plain Old Java Objects* (POJOs) that are being created, initialized, assembled, and managed by the Spring *Inversion of Control* container. The order of these operations and the relationships between objects are provided to the Spring IoC container using XML configuration files prior to Spring version 2.5. Starting with 2.5, a small set of annotations was added for configuring beans, and with Spring 3, Java configuration using annotations was introduced.

This chapter covers everything a developer needs to know in order to configure a basic Spring application using XML and Java Configuration. The Java annotations that represent the intermediate step between configurations using XML and full Java Configuration will also be covered.

Old Style Application Development

In the most competent development style, a Java application should be composed of POJOs, simple Java objects each with a single responsibility. In the previous chapter, the entity classes that will be used throughout the book were introduced along with the relationships between them. In order to manage this type of object at the lowest level, the *dao (repository) layer* of the application, classes named repositories will be used. The purpose of these classes is to retrieve, update, create, and delete entities from the storage support, which usually is some type of database.

¹If you are interested in more books about Java Design Patterns, you can check out this book from Apress: <http://www.apress.com/9781484218013?gtmf=s>.