Pivotal

Spring Web Developer 4.2 Certification Study Guide

COMPLETED: DECEMBER 2015

UPDATED & REVISED: AUGUST 2017

Table of Contents

Overview

This guide is designed to help you prepare for the Spring Web Developer 4.0.a certification exam. Please beware it should not be used if you have attended a Spring Web course that was using Spring 3.x (that certification exam is called Spring Web 3 and there is a dedicated certification guide that goes with it dated April 2011). There is also a dedicated guide, if you took Spring Web V4.0, which is dated March 2015.

The certification exam is based on the 4-day Spring Web 4.0.a training and the materials provided with it are the ideal source to use for preparation. Of course as with any certification the most valuable part, besides recognition, is the learning process. Hence we encourage you to take time to experiment and follow your curiosity when questions arise.

A 4-day course contains a lot of material. To help you focus your efforts and to know when you're ready we've put together this guide. The guide contains a list of topics and a list of further resources. Topics are organized by subject area, where each topic contains a description of what you should make sure you know.

The list of topics can be used as a check-list. The training materials can be used as a point of reference and as a learning ground. The list of resources is where you can go further for getting answers.

One possible way to prepare is to do the following for each training module:

- Review the slides, making notes of questions
- Work through the lab
- Review the list of topics that matches to the module by subject area
- Use the lab to experiment with anything you need to spend more time on
- Use the provided list of resources to look for further answers
- Reading (at least partially) the reference documentation
- Memorize the "big pictures", tables, overviews, …

Read the chapter on Spring MVC in the online documentation – but only the material covered in the course is relevant

Of course there are many more ways to organize your efforts. Pair up with someone else planning to take the exam or review all presentations for a given subject area before going through the labs. If possible, review actual applications at your work to test your knowledge.

Please keep in mind that you are expected to have good working knowledge of all the topics listed. Most of the questions will be very general, however you will be asked a few advanced questions.

Logistics

When you are ready to test and validate your product knowledge, please visit pivotal.io/training/certification to *purchase* an industry-recognized Pivotal certification exam.

Pivotal partners with Exams Local to remotely proctor our exams. Our certification exams may be taken from a location of your choosing anywhere in the world provided you can meet the basic system and test environment requirements and have a valid form of photo identification.

For help with your exam purchase, exam registration process, credentials verification, or other questions related to our certification program, process, and procedures, please visit pivotal.io/training/faq/certification.

The Exam

The exam itself is a computer-based exam. The exam software first gives you some general instructions: how to navigate, how to mark a question, and so forth - please read it carefully.

Once you have agreed that you want to start, you have 90 minutes to answer 50 multiple-choice questions. You must answer 38 questions correctly (76%) in order to pass the exam.

Basic exam technique applies: read each question *carefully* and answer the question that was asked *not* what you *thought* was asked.

Exam FAQ

1. *Is there anything in the exam which was not covered in the course?*

   No.

2. *Do I have to remember class names and method signatures?*

   No. We think that this is why you are using an IDE - for us it's much more important that you've understood the concepts rather than learning method signatures. However you should recognize key Spring Web classes in code examples.

3. *Do I have to write, complete or rearrange source code?*

   No. But you should be able to read a snippet of code and understand what it's doing. This might be an example of a class implementing a Controller and you will then see a couple of related questions. We do not ask you questions on things an IDE can do for you, like checking if the code will compile.

4. *Do I have to know any other APIs like JavaScript or JSP in Detail?*

   No. Of course you should be able to read, understand and use JavaScript or JSP wherever it is necessary but this is not an exam about either.

5. *Are the advanced slides also part of the exam?*

   No. Only the content presented before each chapter lab slide will be on the exam. Any course content presented after any chapter lab will not be on the exam. No content from the optional chapters will be on the exam.

   EXCEPTION: Please note that the material after the lab-slide in the *Getting Started* section (entitled “Simplifying Configuration”) *is* required for the exam.

Topics

The following is a list of topics, each of which is likely to have questions on the exam. The topics are organized by subject area but don't necessarily correspond exactly to sections of the course.

Several of the bullet points below overlap, asking the same (or a related) question in a different way. The answer to more than one bullet point question may be the *same* as the answer to another question just before or after. *Don't let this confuse you*.

Spring Web Overview

- Basic facts about what Spring Web is, what products it consists of and how they relate to each other in terms of dependencies.
- How Spring applications can be loaded in a Servlet container independent of what framework is used (Spring MVC, Struts, etc.) to develop the web layer.
- What is the role of the ContextLoaderListener?
- How does the application context get initialized and from what files?

TIP:

> An easy way to explore project dependencies is to start the Spring Tool Suite, open a maven pom.xml file for a project that uses Spring MVC, then click on the "Dependency Graph" (or the "Dependency Hierarchy") tab.
>
> Alternatively, browse https://github.com/spring-projects/spring-mvc-showcase (sample applications that highlight Spring MVC features).

SPRING MVC SERVLET CONFIGURATION

- How to configure DispatcherServlet in web.xml or using pure Java (servlet-3 style)
- Understand @EnableMvc and the WebMvcConfigurerAdapter class
- Where do the ContextLoaderListener and the DispatcherServlet can find their configuration?
- What features were introduced in Spring 4 to manage static resources?
- What is the Resource Handling Chain?

Tip:

> An important concept to keep in mind is that the DispatcherServlet looks for certain types of Spring MVC "infrastructure" beans in its Spring configuration. Examples of such types of beans include MessageSource, HandlerMapping, ViewResolver, and others. The exam will test your understanding of the purpose of these Spring MVC beans including what their purpose is, how they are discovered (by bean class type or by bean id), what implementations are configured by default, and so on.
>
> Understanding Spring MVC "infrastructure" beans and what they do/provide is an important step to learning Spring MVC. One way to learn is to open each type listed in the sections below, review the class-level Javadoc, and explore available subtypes (in Eclipse/STS use the Ctrl+T shortcut to open a class).
>
> Another Eclipse/STS exercise is to use Ctrl+Space in Spring configuration files on bean property names in order to see the list of available properties. These properties describe how a given infrastructure type can be customized.

HANDLERMAPPING

- The purpose of the HandlerMapping strategy and the details of configuring it.
- How the RequestMappingHandlerMapping and the ControllerClassNameHandlerMapping implementations work and what they would do in the case of specific incoming requests?
- Which is the default handler mapping?

HANDLERADAPTER

- The purpose of the HandlerAdapter strategy and the details of configuring it.

HANDLERINTERCEPTOR

- The purpose of the HandlerInterceptor strategy and the details of configuring it
- What points in the request lifecycle can be intercepted, what data is available at each point?

MESSAGESOURCE

- The purpose of the MessageSource strategy and the details of configuring it.
- How does Spring recognize this bean, by name or by type?

THE MVC NAMESPACE

- What does the *mvc* namespac provide, what you can configure?
- What do *mvc:annotation-driven, mvc:interceptor, mvc:resource-chain* and *mvc:view-controller* actually do?

CONVENTION OVER CONFIGURATION

- How is Spring MVC configured by default?
- What conventions are available to reduce explicit coding?
- What features, introduced by Spring 3, must be enabled by specifying @EnableMvc or *mvc:annotation-driven*?

Spring MVC Programming Model Essentials

The basics of the annotation-based programming model in Spring MVC, configuring @Controller-annotated classes, writing request-handling methods, and testing them.

@REQUESTMAPPING ANNOTATION

- The purpose of the annotation, what can be annotated, and what options (or attributes) it provides.
- How a URL breaks down (web application context, servlet name, path info) as well as which part of the URL is used in Spring MVC for request mapping purposes.

Tip:

In the exam you may be given basic examples of URL's and annotated controller methods. You will be expected to predict what methods will be invoked. The best way to experiment is to try it out. Use the course labs, and check the Spring MVC logging information on each request!

Also examine the logs for output from HandlerMapping beans. HandlerMapping beans construct URL mappings during startup and log that information. The log level for org.springframework.web must be set to DEBUG.

Below are a couple of specific scenarios to experiment with.

Scenario 1: a *RequestMappingHandlerMapping* with one Controller and a single method annotated with *@RequestMapping("/foo")*.

Questions: What URL's reach the method successfully? Does adding an extension to the URL (.pdf, .xml) make a difference? How about more segments (/foo/bar)? Passing anything (/other)? Try switching to *@RequestMapping(method=RequestMethod.GET)*, does it still work?

Scenario 2: Similar to scenario 1 but involving a *ControllerClassNameHandlerMapping*, a controller (FooController) and a method (bar) annotated with *@RequestMapping(method=RequestMethod.GET)*

REQUEST HANDLING METHODS

- How to write methods to handle requests
- What annotations can be used?
- What the signature of the method can be – input argument types, return types?
- What happens if the method returns void?

- **Tip:**

  In addition to the annotations listed below, the JavaDoc of the @RequestMapping annotation is a good place to start for information on how input arguments and return values are interpreted on @RequestMapping-annotated methods.

@REQUESTPARAM AND @PATHVARIABLE ANNOTATIONS

- The purpose of the annotation
- What can be annotated?
- What options (or attributes) does it provide?
- Can it handle optional parameters?

@MODELATTRIBUTE ANNOTATION

- The purpose of the annotation, what can be annotated, what options (or attributes) it provides?
- What is the default name given to a model attribute object if the attribute name is left unspecified?
- What is the default attribute name given to a model attribute object that is an array or a collection?

OTHER ANNOTATIONS

- How do you access request headers or cookies?
- What does `@Value("#{request.requestURL}")` do?

Spring MVC Views

- The basics of how views work.
- What do they do?
- How are they typically instantiated through the process of view resolution?
- What is a logical view name?
- What is the default logical view name selected if a controller method does not specify it (method returns void or null).

VIEW RESOLVERS

- The purpose of the ViewResolver strategy, and the details of configuring it?
- You should be familiar with the several different view-resolvers covered on the course.
- How do ViewResolver chains work?
- How they can be used to render multiple content types – for example to re-use the same controller method to render HTML, PDF, or XML depending on the content type requested by the client.
- Understanding `<mvc:view-resolvers>`
- Content negotiation can be specified in three different ways – how?

Spring MVC Form Processing

- The basics of working with forms such as how to configure data binding through the *@ModelAttribute* annotation and how data binding is used to populate the form object from request parameter values.
- Can data binding be used on a POST or can it also be done on a GET request (consider search forms vs. forms updating data).
- How can a request handling method get access to the results of data binding?

@SESSIONATTRIBUTES ANNOTATION

- The purpose of the *@SessionAttributes* annotation, what can be annotated, and what options it provides.

Tip:

> The @SessionAttributes annotation provides more than one way to specify what objects should be added to the HTTP Session. Be sure to check them. A good way to learn what options any Spring annotation provides is the Javadoc of the annotation.

- Understand the lifecycle of attributes specified by *@SessionAttributes* – how long they remain around, when and how they can be removed from the HTTP Session.
- How does *@SessionAttributes* work when it's used with multiple controllers – for example is the data stored globally to the HTTP session (e.g. user preferences) or is it per-controller (e.g. account editing).

@INITBINDER ANNOTATION AND DATA BINDING CUSTOMIZATIONS

- The purpose of the *@InitBinder* annotation, what can be annotated, and what options it provides.
- What customizations can be applied to the data binding mechanism?

- What error codes are generated automatically during data binding?
- What error codes can be used to customize the errors generated during data binding?

SPRING FORM TAG LIBRARY

- The purpose and the value provided by the Spring form tags, and how to use them.
- How are error-messages handled? In the Controller? Using form tags?

FORMATTERS AND VALIDATION

- Understand the Spring stateless formatters (introduced by Spring 3.0)
- What can they do, where are they used?
- How do they interact with the formatting annotations?
- How can the form model object be validated?
- How do Spring forms leverage JSR 303 bean validation?
- How can error messages be customized?

Spring MVC REST

- What does REST stand for?
- What is a resource?
- What are safe operations?
- What are idempotent operations? Why is idempotency important?
- Is REST scalable and/or interoperable?
- What are the advantages of the *RestTemplate*?
- Which HTTP-Methods does REST use?
- What is an *HttpMessageConverter*? How are they configured?
- Is REST stateless?
- What does *@RequestMapping* do?
- Is *@Controller* a stereotype? Is *@RestController* a stereotype?
- What is the difference between *@Controller* and *@RestControler*?
- When do you need *@ResponseBody*? Or *@RequestBody*?
- What does *@PathVariable* do?
- What is the HTTP status code for a delete statement? What about for a create?
- What does CRUD mean? Which HTTP methods do you use for each CRUD operation?
- Is REST secure? What can you do to secure it?
- Where do you need *@EnableWebMVC*? What about *<mvc-annotation-driven>*?
- Name some common HTP response codes. When do you need *@ResponseStatus*?
- Does REST work with transport layer security (TLS)?
- Do you need Spring MVC in your classpath?

Exceptions

- How can exceptions be handled in the MVC framework?
- What does a *ControllerAdvice* class do?
- How do you set the HTTP status of a response in the event of an exception (to avoid the default 500 error)
- Would you understand the configuration of a *HandlerExceptionResolver* if you saw it?

- What can a *SimpleMappingExceptionResolver* be used for?
- How can a RESTful request return an error?

Securing Web Applications With Spring Security

- What web.xml configuration is required to enable Spring Security and what the mechanism Spring Security uses to protect web applications.
- What Spring Security related configuration is needed to secure a web application.
- How should URL patterns be configured?

Tip:

> Pay special attention to the order in which URL patterns are provided. Does it matter if you put the more general (e.g. */accounts/*)* or the more specific (e.g. */accounts/edit*) pattern first?

- How method level security can be added to an application.
- How authentication and authorization relate to each other – for example does the choice of authentication affect authorization?

Testing Web Applictions

- How can the MVC layer be tested?
- How do we test Controller *logic*?
- How do we test if a Controller works properly inside the MVC framework?
- What framework artifacts/components does Spring's Mock MVC allow you to test? (Refer to *Testing Each Layer* slide)
- If presented with the code for a Mock MVC test would you understand what it is doing? – we *don't* expect you to know the API from memory.

Spring Boot

- What is Spring Boot? What are the advantages of using Spring Boot?
- Why is it “opinionated”?
- How does it work? How does it know what to configure?
- What things affect what Spring Boot sets up?
- How are properties defined? Where?
- Would you recognize common Spring Boot annotations and configuration properties if you saw them in the exam?
- What is the difference between an embedded container and a WAR?
- What embedded containers does Spring Boot support?
- What does `@EnableAutoConfiguration` do? What about `@SpringBootApplication`?
- What is a Spring Boot starter POM? Why is it useful?
- Spring Boot supports both Java properties and YML files. Would you recognize and understand them if you saw them?
- Can you control logging with Spring Boot? How?

Spring MVC and Websockets

- What is a web-socket?
- What is STOMP?
- How does Spring MVC support web-sockets? What annotation(s) are involved?
- How do you perform messaging with Spring MVC web-sockets?
- What is a User Destination?
- Why would you use *SimpTemplate*?

Tip:

If you know how to use JMS or AMQP with Spring, messaging over a web-socket using SimpTemplate is similar.

Resources

This section contains a list of resources relating for learning.

- The best place to go for help is at StackOverflow – look for existing discussions or start your own, take advantage of one of the best parts of Spring: its community.
- A list of Spring topics from StackOverflow can also be found at Spring Questions.
- Spring Blog – point your favorite RSS reader or come back every so often for detailed, quality posts by Spring developers. The STS Dashboard automatically takes updates from here.
- Reference Documentation – add bookmarks in your browser to the reference documentation pages for Spring (namely the chapter on Spring MVC), and Spring Security.
- The Spring Framework API Javadoc.
- The Spring IO website has many Spring Guides.
- Spring By Example – a great repository with complete code samples and the ability to contribute your own samples. Code also available on Github.
- It's hard to single out individual web sites, there are so many, but we had to name a few they would include InfoQ, JavaZone@DZone, JavaWorld among many others.
- There are also many books and even the best become out of date quite quickly. Publishers like (e.g. Apress, Manning) provide early access to book chapters in PDF as they are being written. Many of the well-known books have chapters on Spring MVC.
  - Spring books at Apress.
  - The are several Spring books published by Spring Books at Manning – use their search option.
- Spring Projects JIRA – most likely *not* the first place to come to in the beginning but overall a great learning resource when looking up information on very specific issues, new features or potential bugs. You can read comments, leave comments, as well as vote. Sometimes discussions on the community forums result in the creation of issues in JIRA.

Tools

Spring Tool Suite

Spring project templates, tutorials, bean diagrams, code completion and more. STS is updated often and it's free.

Cloud Foundry

Pivotal's open *Platform as a Service* (PaaS) is a great place to run Spring and Java applications (among others). Pivotal Web Services is our public cloud offering and a trial account is free for 60 days and pretty cheap thereafter. If you want to run Spring applications *and* make them available to friends, customers or colleagues world-wide, PWS is easy to use and requires no hosting by you.

Conclusion

When you worked through this guide and know all the answers, we are pretty confident that you should pass the certification. It's recommended to do it as soon as possible and we wish you good luck with it.

Thank you again for choosing Pivotal as your education partner and good luck with your projects.

If you have encountered any errors, have any suggestions or enquiries please don't hesitate to contact your trainer or send an email to education@pivotal.io.

