April 11, 2017 NO COMMENTS

Spring Boot Interview Questions and Answers

« Previous Next »

In this article, I have collected best top 20 most frequently asked spring boot interview questions and answers with explanation. You can find more detail about Spring Boot in my previous Spring Boot tutorial and also microservices architecture with Spring Boot.

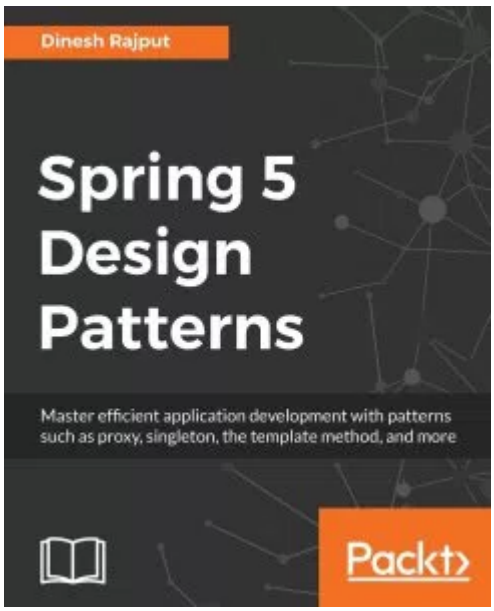
Spring Boot Interview Questions

1. What is Spring Boot?
2. What are the advantages of using Spring Boot?
3. What are the disadvantages of using Spring Boot?
4. Why is it "opinionated"?
5. How does it work? How does it know what to configure?
6. How are properties defined? Where?
7. What is the difference between an embedded container and a WAR?
8. What embedded containers does Spring Boot support?
9. What does @EnableAutoConfiguration do? What about @SpringBootApplication?
10. What is a Spring Boot starter POM? Why is it useful?
11. Spring Boot supports both Java properties and YML files. Would you recognize and understand them if you saw them?
12. Can you control logging with Spring Boot? How?

13. How to reload my changes on Spring Boot without having to restart server?
14. What is Actuator in Spring Boot?
15. How to run Spring boot application to custom port?
16. How to implement security for Spring boot application?
17. What is the configuration file name used by Spring Boot?
18. How to implement Spring web using Spring boot?
19. How to configure database using Spring boot?
20. What is YAML?

Spring 5 Design Pattern Book

You could purchase my **Spring 5 book** that is with title name "**Spring 5 Design Pattern**". This book is available on the **Amazon** and **Packt** publisher website. Learn various **design patterns** and **best practices** in Spring 5 and use them to solve common design problems. You could use author discount to purchase this book by using code- "**AUTHDIS40**".



Spring Boot Interview Questions and Answers

1. What is Spring Boot?

First of all Spring Boot is not a framework, it is a way to ease to

create stand-alone application with minimal or zero configurations. It is approach to develop spring based application with very less configuration. It provides defaults for code and annotation configuration to quick start new spring projects within no time. It leverages existing spring projects as well as Third party projects to develop production ready applications. It provides a set of Starter Pom's or gradle build files which one can use to add required dependencies and also facilitate auto configuration.

Spring Boot automatically configures required classes depending on the libraries on its classpath. Suppose your application want to interact with DB, if there are Spring Data libraries on class path then it automatically sets up connection to DB along with the Data Source class.

2. What are the advantages of using Spring Boot?

- It is very easy to develop Spring Based applications with Java or Groovy.
- It reduces lots of development time and increases productivity.
- It avoids writing lots of boilerplate Code, Annotations and XML Configuration.
- It is very easy to integrate Spring Boot Application with its Spring Ecosystem like Spring JDBC, Spring ORM, Spring Data, Spring Security etc.
- It follows "Opinionated Defaults Configuration" Approach to reduce Developer effort
- It provides Embedded HTTP servers like Tomcat, Jetty etc. to develop and test our web applications very easily.
- It provides CLI (Command Line Interface) tool to develop and test Spring Boot (Java or Groovy) Applications from command prompt very easily and quickly.
- It provides lots of plugins to develop and test Spring Boot Applications very easily using Build Tools like Maven and Gradle
- It provides lots of plugins to work with embedded and in-memory Databases very easily.

3. What are the disadvantages of using Spring Boot?

It is very tough and time consuming process to convert existing or legacy Spring Framework projects into Spring Boot

Applications. It is applicable only for brand new/Greenfield Spring Projects.

4. Why is it "opinionated"?

It follows "Opinionated Defaults Configuration" Approach to reduce Developer effort. Due to opinionated view of spring boot, what is required to get started but also we can get out if not suitable for application.

• Spring Boot uses sensible defaults, "opinions", mostly based on the classpath contents.

• For example

– Sets up a JPA Entity Manager Factory if a JPA implementation is on the classpath.

– Creates a default Spring MVC setup, if Spring MVC is on the classpath.

• Everything can be overridden easily

– But most of the time not needed

5. How does it work? How does it know what to configure?

• Auto-configuration works by analyzing the classpath

– If you forget a dependency, Spring Boot can't configure it

– A dependency management tool is recommended

– Spring Boot parent and starters make it much easier

• Spring Boot works with Maven, Gradle, Ant/Ivy

– Our content here will show Maven

6. How are properties defined? Where?

In spring boot, we have to define properties in the application.properties file exists in classpath of application as follow.

Example: configure default DataSource bean

```
database.host=localhost
database.user=admin
```

7. What is the difference between an embedded container and a WAR?

There is no force to go container less

– Embedded container is just one feature of Spring Boot

• Traditional WAR also benefits a lot from Spring Boot

– Automatic Spring MVC setup, including DispatcherServlet

– Sensible defaults based on the classpath content
– Embedded container can be used during development

8. What embedded containers does Spring Boot support?
Spring Boot includes support for embedded Tomcat, Jetty, and Undertow servers.
By default the embedded server will listen for HTTP requests on port 8080.

9. What does @EnableAutoConfiguration do? What about @SpringBootApplication?
@EnableAutoConfiguration annotation on a Spring Java configuration class
– Causes Spring Boot to automatically create beans it thinks you need
– Usually based on classpath contents, can easily override

```
@Configuration
@EnableAutoConfiguration
public class MyAppConfig {
public static void main(String[] args) {
SpringApplication.run(MyAppConfig.class, args);
}
}
```

@SpringBootApplication was available from Spring Boot 1.2
It is very common to use @EnableAutoConfiguration, @Configuration, and @ComponentScan together.

```
@Configuration
@ComponentScan
@EnableAutoConfiguration
public class MyAppConfig {
...
}
```

With @SpringBootApplication annotation

```
@SpringBootApplication
public class MyAppConfig {
...
}
```

10. What is a Spring Boot starter POM? Why is it useful?
Starters are a set of convenient dependency descriptors that you can include in your application. The starters contain a lot of the dependencies that you need to get a project up and running quickly and with a consistent, supported set of managed transitive dependencies.

The starter POMs are convenient dependency descriptors that can be added to your application's Maven. In simple words, if you are developing a project that uses Spring Batch for batch processing, you just have to include spring-boot-starter-batch that will import all the required dependencies for the Spring Batch application. This reduces the burden of searching and configuring all the dependencies required for a framework.

11. Spring Boot supports both Java properties and YML files. Would you recognize and understand them if you saw them?
spring boot application java property file name is application.properties
spring boot application YML file name is application.yml

12. Can you control logging with Spring Boot? How?
Yes, we can control logging with spring boot.

Customizing default Configuration for Logging:
By adding logback.xml file to the application we can override the default logging configuration providing by the Spring Boot. This file place in the classpath (src/main/resources) of the application for Spring Boot to pick the custom configuration.

Spring Boot can control the logging level
– Just set it in application.properties
• Works with most logging frameworks
– Java Util Logging, Logback, Log4J, Log4J2

```
logging.level.org.springframework=DEBUG
logging.level.com.acme.your.code=INFO
```

13. How to reload my changes on Spring Boot without having to restart server?
Include following maven dependency in the application.

```
<dependency>
 <groupId>org.springframework</groupId>
 <artifactId>springloaded</artifactId>
 <version>1.2.6.RELEASE</version>
</dependency>
```

Automatic restart

Applications that use spring-boot-devtools will automatically restart whenever files on the classpath change. This can be a useful feature when working in an IDE as it gives a very fast feedback loop for code changes. By default, any entry on the classpath that points to a folder will be monitored for changes.

```
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-devtools</artifactId>
<optional>true</optional>
</dependency>
```

This can be achieved using DEV Tools. With this dependency any changes you save, the embedded tomcat will restart. Spring Boot has a Developer tools (DevTools) module which helps to improve the productivity of developers. One of the key challenge for the Java developers is to auto deploy the file changes to server and auto restart the server. Developers can reload changes on Spring Boot without having to restart my server. This will eliminates the need for manually deploying the changes every time. Spring Boot doesn't have this feature when it has released it's first version. This was a most requested features for the developers. The module DevTools does exactly what is needed for the developers. This module will be disabled in the production environment.

14. What is Actuator in Spring Boot?

pring Boot Actuator is a sub-project of Spring Boot. It adds several production grade services to your application with little effort on your part. There are also has many features added to your application out-of-the-box for managing the service in a production (or other) environment. They're mainly used to expose different types of information about the running application – health, metrics, info, dump, env etc.

15. How to run Spring boot application to custom port ?

In application.properties, add following property.

```
server.port = 8181
```

16. How to implement security for Spring boot application ?

Add spring security starter to the boot application

```
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-security</artifa
  </dependency>
```

17. What is the configuration file name used by Spring Boot?

The configuration file used in spring boot projects is application.properties. This file is very important where we would over write all the default configurations. Normally we have to keep this file under the resources folder of the project.

18. How to implement Spring web using Spring boot?

Web Application Convenience

• Boot automatically configures
– A DispatcherServlet & ContextLoaderListener
– Spring MVC using same defaults as @EnableWebMvc
• Plus many useful extra features:
– Static resources served from classpath
• /static, /public, /resources or /META-INF/resources
– Templates served from /templates
• If Velocity, Freemarker, Thymeleaf, or Groovy on classpath
– Provides default /error mapping
• Easily overridden
– Default MessageSource for I18N

19. How to configure datasource using Spring boot?

• Use either spring-boot-starter-jdbc or spring-boot-starterdata-jpa and include a JDBC driver on classpath
• Declare properties

```
spring.datasource.url=jdbc:mysql://localhost/test
spring.datasource.username=dbuser
spring.datasource.password=dbpass
spring.datasource.driver-class-name=com.mysql.jdbc.
```

– Spring Boot will create a DataSource with properties set
– Will even use a connection pool if the library is found on the classpath!

20. What is YAML?

Yaml Ain't a Markup Language
– Recursive acronym
• Created in 2001
• Alternative to .properties files
– Allows hierarchical configuration
• Java parser for YAML is called SnakeYAML
– Must be in the classpath
– Provided by spring-boot-starters

YAML for Properties

• Spring Boot support YAML for Properties
– An alternative to properties files
application.properties

```
database.host = localhost
database.user = admin
```

application.yml

```
database:
    host: localhost
    user: admin
```

• YAML is convenient for hierarchical configuration data
– Spring Boot properties are organized in groups
– Examples: server, database, etc

Spring Boot Related Topics

1. Introduction to Spring Boot
2. Essentials and Key Components of Spring Boot
3. Spring Boot CLI Installation and Hello World Example
4. Spring Boot Initializr Web Interface
5. Spring Boot Initializr With IDEs
6. Spring Boot Initializr With Spring Boot CLI
7. Installing Spring Boot
8. Developing your first Spring Boot application
9. External Configurations for Spring Boot Applications
10. Logging Configuration in Spring Boot
11. Spring Boot and Spring MVC
12. Working with SQL Databases and Spring Boot
13. MySQL Configurations
14. Spring Data JPA using Spring Boot Application
15. Spring Boot with NoSQL technologies
16. Spring Cache Tutorial
17. Spring Security Tutorial with Spring Boot
18. Spring Boot and MongoDB in REST Application
19. Complete Guide for Spring Boot Actuator
20. Microservices with Spring Boot

« Previous Next »

INTERVIEW QUESTIONS

SPRING BOOT

Related Posts