Spring boot interview questions for 10 years experience

Table of Contents [hide]

In this post, we will see top 15 Spring Boot interview questions with answers.If you want to read more about Spring boot, you can go through Spring boot tutorial.
If you are looking for below queries then this post will help you as well.

- Spring boot interview questions for 3 years experience
- Spring boot interview questions for 5 years experience
- Spring boot interview questions for 7 years experience

Here is the list of Spring Boot interview questions.

1. What is Spring boot?

Spring Boot makes it easier for you to create production ready applications in no time. It is an opinionated view to create Spring application quickly. It follows convention over configuration. In simple terms, it comes with default configurations for most of the Spring projects, you don't need to do much to bootstrap any spring application.

2. Why did you use Spring boot in your application?

As discussed earlier, Spring boot makes it easier for you to create Spring application, it can save a lot of time and efforts.

For example: Let's say you want to create Spring boot project with activeMQ. You can simply use "spring–boot–starter–activemq" as artifact Id, it will take all the defaults and create Spring application with ActiveMQ configured. Let's say you don't want to use inbuilt activeMQ, you can simply override "spring.activemq.broker-url" in application.properties to use external ActiveMQ.

3. Can you list advantages of Spring boot?

Advantages of Spring boot are:

- It increases productivity as you can create Spring application quickly.
- It provides a lot of starter project for easy maven integration.You don't have to worry about version mismatch.
- You can quickly create using sample project using spring boot initializer

4. What are disadvantages of Spring boot?

If you want to convert your old spring application to Spring boot application, it may not be straight forward and can be time consuming.

5. How can you override default properties in Spring boot Project?

Spring boot provides a lot of properties which can be overridden by specifying them in application.properties.

For example: You want to specify prefix and suffix in Spring MVC applications. You can simply do it by putting below properties in application.properties.

spring.mvc.view.prefix: /WEB-INF/
spring.mvc.view.suffix: .jsp

5. How can you run Spring boot application on custom port?

You can simply put server.port properties in application.properties.

For example:server.port=8050

6. What is Spring boot starter and how it is useful?

Spring boot comes with a lot of starters which is set of convenient dependency descriptors which you can include in your pom.xml.

For example: Let's say you want to work Spring MVC application, you can simply include "**spring–boot–starter–web**" as dependency in pom.xml .

7. Can we use Spring boot with applications which are not using Spring?

No, it is not possible as of now. Spring boot is limited to Spring applications only.

8. What is name of the configuration file which you use in Spring boot?

Configuration file used in Spring boot projects is application.properties. It is very important file as it is used to override all default configurations.

9. What is DevTools in Spring boot?

Spring boot comes with DevTools which is introduced to increase the productivity of developer. You don't need to redeploy your application every time you make the changes.Developer can simply reload the changes without restart of the server. It avoids pain of redeploying application every time when you make any change. This module will be disabled in production environment.

10. What is actuator in Spring boot?

Spring boot actuator provides restful web services end points which you can simply use and check various metrics. For example:
/metrics : This restful end point will show you metrics such as free memory, processors, uptime and many more properties,

Spring boot actuator will help you to monitor your application in production environment.Restful end points can be sensitive, it means it will have restricted access and will be shown only to authenticated users. You can change this property by overriding it in application.properties.

11. How can you implement Spring security in Spring boot application?

Implementation of Spring security in Spring boot application requires very little configuration. You need to add **spring-boot-starter-security** starter in pom.xml.You need to create Spring config class which will extend WebSecurityConfigurerAdapter and override required method to achieve security in Spring boot application.

You can read more about Spring boot security example.

12. Have you used @SpringBootApplication annotation in Spring boot project?

@SpringBootApplication annotation was introduced in Spring Boot 1.2.0. This annotation is equivalent to declaring these 3 annotations.

- @Configuration
- @EnableAutoConfiguration
- @ComponentScan

For example:

When you create your main class with Spring boot, you have to use below annotations before Spring boot 1.2.0.

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

But after Spring boot 1.2.0, you just need to use @SpringBootApplication annotation which will cover above 3 annotations

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

13. What are embedded containers which are supported by Spring Boot?

configure external ActiveMQ?

Spring Boot comes with embedded ActiveMQ.We need to use "spring–boot–starter–activemq" dependency in pom.xml and it will take care of all defaults and will configure ActiveMQ in the project.

If you want to configure external ActiveMQ then you need to just put "spring.activemq.broker-url" in application.properties and provide the URL of external ActiveMQ.

15. How can you configure logging in Spring boot application?

Spring Boot comes with support for Java Util Logging, Log4J2 and Logback and it will be pre-configured as Console output.
Hence,You can simply specify logging.level in application.properties.

```
logging.level.spring.framework=Debug
```

It will set Spring framework logs to debug level.
Let's say you want to put logs to the file.You can specify logger.file in application.properties.

```
logging.file={java.io.tmpdir}/application.log
```

If you want to do logging configuration explicitly, You can also create logback.xml in main/java/resources folder and specify logging configuration in the file. Spring Boot will pick this file and configure logging accordingly.

16. Which embedded servers are used in a Spring Boot application

Spring Boot provides an embedded HTTP server so that you can get started quickly and has support for Tomcat, Jetty, and Undertow.

17. What are the different ways you can leverage to run and deploy a Spring Boot application

Spring Boot apps can be run as standalone because they include an embedded server enabling it to be run from the IDE or the command line.

If you want to deploy the application traditionally, you can deploy a `war` file to an external server like Tomcat, JBoss, and WebSphere, and it can work just like you used it in the past.

For example, if we have a Tomcat deployed somewhere on your corporate network, you can deploy that Spring Boot app with the `.war` file extension.

As a war file, you only have your code included, and there is no need to have the embedded server because now you are deploying it in a traditional sense.

There is already a Tomcat server installed running elsewhere, and we are simply deploying our war file to that server.

18. Does Spring Boot replace Spring MVC, Spring REST, etc...

No, Instead, Spring Boot uses those technologies in the background. Spring Boot uses Spring Core, Spring AOP, Spring MVC, Spring REST, and other technologies in the background, so there is no competition among them. Spring Boot is mainly about configuration, and once you do your configs, then you can do the regular Spring coding just as you have `@Component`, `@Controller`, `@Autowired` etc.

19. Does Spring Boot code run faster than regular Spring code

No, behind the scenes, Spring Boot uses the same code as Spring Framework, and its sole purpose is about making it easier to get started minimizing configuration.

20. What is Spring Boot starter parent?

This is a special starter that provides Maven defaults. In your `pom.xml` you will have an entry for parent, and you will give the actual groupId artifactId and version which is included in the `pom.xml` directly when using the Spring initializer.

```
<parent>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-parent</artifactId>
  <version>2.5.3</version>
  <relativePath/> <!-- lookup parent from repository -->
</parent>
```

The maven defaults defined in the starter parent are default compiler level which is Java 8, UTF-8 source encoding, and other features out there.

To override a default, set it as a property.

```
<properties>
    <java.version>11</java.version>
</properties>
```

For Spring Boot starter dependencies, there is no need to list the version as they inherit the version from the starter parent, which is great for maintenance and also helps to make sure that all the dependencies that you are using are compatible.

Dependency management by using the version on parent only and all Spring Boot starter dependencies extend version from the parent.

Provides default configuration of the Spring Boot plugin as shown below.

```
<build>
    <plugins>
      <plugin>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-maven-plugin</artifactId>
      </plugin>
    </plugins>
  </build>
```

22. What are some of the endpoints exposed by Spring Boot Actuator?

All the endpoints are prefixed `/actuator` and each request returns a JSON response from the server to authorized users only.

`/health` checks the status of your application to see if your app is up or down.

`/info` gives you information about your application but is empty by default, and you need to customize the info endpoint by updating the `application.properties` with your app info.

```
info.app.name = My Super Cool App
info.app.description = An app that does magic
info.app.version = 2.1.0
```

`/auditevents` inspect events for your application.

`/beans` return a list of all beans registered in the Spring application context.

`/mappingd` returns a list of all @RequestMapping paths.

23. How can you expose all Spring Boot actuator endpoints over HTTP

```
management.endpoints.web.exposure.include=*
```

24. How do you inject custom application properties?

Your app needs to be configurable to prevent hard coding of values by reading values from a properties file.

By default, Spring Boot reads information from a standard properties file located at `src/main/resources/application.properties`.

You can define any custom properties in this file, and your Spring Boot app can access properties using `@Value` annotation.

25. What logger does Spring Boot use?

Spring Boot provides support for Logback, Log4j2, and Java Util Logging in its default configuration but uses the Apache Commons logging for all internal logging.

26. How do you integrate Hibernate and JPA with Spring Boot?

Spring Boot will automatically configure your data source for you based on entries from the Maven POM file. In your POM file, you give a reference to your JDBC Driver, Spring Data (ORM) and setup database connection properties in `application.properties`, and Spring Boot will use this information to create a data source for you.

27. What is Spring Boot auto data source configuration?

Based on the configurations provided, Spring Boot automatically creates Beans for `DataSource` and `EntityManager`, and then you can inject these into your data access objects (DAO). EntityManager is a class from the Java persistence API (JPA). In Spring Boot, Hibernate is the default implementation of JPA. The EntityManager is similar to `HibernateSessionFactory`, and EntityManager can serve as a wrapper for a Hibernate `Session` object.

28. What is JPA?

JPA is a standard API for Object-to-Relational-Mapping and acts as a specification that defines a set of interfaces and requires an implementation to be usable.

29. Which are the JPA vendor implementations?

One of the implementations is `Hibernate`, just like java coding, they take those interfaces and provide an implementation of those interfaces. Another implementation is `EclipseLink`, which also have their implementation of the JPA specification. There are other implementations, but Hibernate is the most popular implementation of the JPA specification.

30. What are the benefits of JPA?

By having a standard API, you are not locked to vendor implementation, so you can maintain portable, flexible code by coding to JPA specifications. Theoretically, you can switch vendor implementations. For example, if Hibernate stops supporting their products, you can switch to EclipseLink without vendor lock-in because you are coding to the actual JPA standard API.

EntityManager, and standard JPA API, and finally Spring Data JPA which is the most common nowadays.

32. What is JPQL?

JPQL stands for Jakarta Persistence Query Language which is a query language for JPA and has the following syntax.

```
SELECT a FROM Author a ORDER BY a.firstName, a.lastName
```

33. What are the benefits of Spring Data JPA?

When you create a DAO and plug in your entity type, and primary key, Spring Boot will give you CRUD implementations for free, which help us minimize our boiler-plate DAO code. Depending on the use case, you can get more than a 70% reduction in code.

34. What is JpaRepository?

Spring Data provides an interface called `JpaRepository` and exposes methods, some of which are inherited from parents, and these are CRUD methods that you can use. Some of these methods include `findAll()`, `findById()`, `save()`, `deleteById()` and others.

35. How does Spring Data REST work in the background? Spring Data REST scans your project for `JpaRepository` and exposes REST APIs for each entity type for your JpaRepository.

In the code below, if you have a custom repository that extends JpaRepository and an entity type of type customer, It will expose a customer's endpoint.

By default, Spring Data REST will create endpoints based on entity type by making use of the simple pluralized form, meaning the first character of the entity type is lowercase then just adds an "s" to the entity.

The entity type is Customer, and it will create the `/customers` endpoint, and that's the basic approach for how they will expose these REST endpoints.

```
public interface CustomerRepository extends JpaRepository<Customer, Long>{

}
```

36. What is HATEOAS, and how does it apply in Spring Data REST?

HATEOAS stands for Hypermedia as the Engine of Application State and Spring Data REST is HATEOAS compliant, meaning that it provides information to access REST interfaces and you can also think of it as meta-data for REST data. For a collection, meta-data includes page size, total elements, and pages.

37. Which are the advanced features of Spring Data REST?

38. What is Thymeleaf?

Thymeleaf is a Java templating engine commonly used to generate HTML views on web pages apps. However, it is a general-purpose templating engine meaning you can use it outside of web apps.

39. Where is the Thymeleaf template processed?

In a web app, the Thymeleaf template is processed on the server, results included in HTML, and returned to the browser.

40. Which are the few Thymeleaf use cases?

We can make use of an email template where when a customer signs up for service; then we will send them a welcome email.

We can also use a CSV template to generate a monthly CSV then upload it to Google drive.

We can use a pdf template to generate a travel confirmation PDF then send it via an email attachment.

That's all about Spring boot interview questions for 10 years experience.