Spring Boot Interview Questions and Answers

Basic Level

1. What is Spring Boot, and how is it different from the Spring Framework?

Spring Boot simplifies the configuration of Spring applications, while Spring Framework requires detailed configuration. Spring Boot includes an embedded server, auto-configuration, and opinionated defaults.

2. What are the main features of Spring Boot?

Key features include:

- Auto-configuration
- Embedded servers (Tomcat, Jetty, Undertow)
- Opinionated 'starter' dependencies
- Spring Boot CLI
- Actuator for monitoring and metrics

3. What are Spring Boot Starters?

Starters are dependency descriptors that simplify adding related dependencies to your project, e.g., spring-boot-starter-web for web applications.

4. What is the purpose of the application.properties file?

It is used to configure Spring Boot application properties such as server ports, logging, database configurations, etc.

5. Explain the @SpringBootApplication annotation.

It is a convenience annotation that combines @Configuration, @EnableAutoConfiguration, and @ComponentScan.

6. How does Spring Boot manage dependencies?

Spring Boot uses a set of curated dependencies to ensure compatibility and simplify version management. It uses Maven or Gradle to manage dependencies.

7. What is auto-configuration in Spring Boot?

Auto-configuration attempts to automatically configure Spring applications based on the dependencies on the classpath. It reduces the need for manual configuration.

8. What are Spring Profiles?

Profiles allow you to define different configurations for different environments, such as dev, test, or prod, making it easier to manage environment-specific configurations.

9. How do you run a Spring Boot application?

Spring Boot applications can be run using:

- Command-line java -jar <application>.jar
- Maven mvn spring-boot:run
- From an IDE using main() method.

10. What is the role of an embedded server in Spring Boot?

Spring Boot provides embedded servers like Tomcat, Jetty, and Undertow, allowing the application to run standalone without needing a separate application server.

Intermediate Level

11. What is Spring Boot Actuator, and why is it used?

Actuator provides production-ready features such as health checks, metrics, and monitoring through endpoints like /actuator/health.

12. How do you configure Spring Boot to connect to a database?

Configure database properties in application.properties or application.yml and include the required JDBC or JPA dependencies.

13. What is a Spring Boot initializer?

It is a web-based tool provided by Spring (start.spring.io) to generate Spring Boot projects with dependencies and configurations.

14. What are the different ways to deploy a Spring Boot application?

Deploy as a JAR file with an embedded server or as a WAR file to an external server (by extending SpringBootServletInitializer).

15. Explain the difference between @Component, @Service, and @Repository.

All are Spring-managed beans, but they are used for different purposes:

- @Component: Generic component for any Spring-managed bean.
- @Service: Specialized component for the service layer.
- @Repository: Specialized component for data access and handling exceptions.

16. What is the @RestController annotation?

It is a combination of @Controller and @ResponseBody and is used to create RESTful web services.

17. How does Spring Boot handle exceptions?

Spring Boot allows global exception handling using @ControllerAdvice and @ExceptionHandler.

18. How to enable HTTPS in a Spring Boot application?

Configure the keystore properties in application.properties and set the appropriate server port and protocol.

19. What is Spring Data JPA?

Spring Data JPA simplifies data access with JPA (Java Persistence API) by providing repository interfaces that can be extended for common database operations without writing implementations.

20. What is the use of the @Transactional annotation?

It ensures that the methods execute within a transaction and rolls back the transaction in case of any runtime exceptions.

Advanced Level

21. What is the use of @ConfigurationProperties in Spring Boot?

This annotation is used to bind external configurations (like properties files) to Java objects, allowing for structured configuration.

22. How does Spring Boot manage logging?

Spring Boot uses SLF4J with Logback as the default logging implementation. You can customize logging using the application.properties file.

23. What are custom starters in Spring Boot?

Custom starters are a way to encapsulate dependencies and auto-configurations that can be reused across multiple applications.

24. What is the role of the SpringBootServletInitializer class?

It is used to configure the application when deployed as a WAR file on an external servlet container.

25. How do you secure a Spring Boot application?

Use Spring Security to secure a Spring Boot application. You can configure authentication, authorization, and access control mechanisms via annotations or configuration classes.

26. What is the difference between @RequestParam and @PathVariable?

@RequestParam extracts query parameters from the URL, while @PathVariable extracts values from the URI path.

27. What are the steps to integrate Spring Boot with Hibernate?

Add dependencies for Spring Data JPA and Hibernate.

Configure the datasource in application.properties.

Annotate entities with @Entity and repositories with @Repository.

28. What is Spring Cloud, and how does it relate to Spring Boot?

Spring Cloud provides tools to build distributed systems and microservices. It builds on top of Spring Boot for configuration and deployment simplicity.

29. What is @EnableAutoConfiguration, and how does it work?

This annotation enables Spring Boot's auto-configuration feature, automatically configuring beans that it thinks you might need based on the dependencies present in your project.

30. How does Spring Boot handle circular dependencies?

Spring Boot handles circular dependencies through proxies or lazily initialized beans. However, it is best to avoid circular dependencies altogether by refactoring the code.

31. What is the difference between @SpringBootTest and @WebMvcTest?

@SpringBootTest loads the full application context for integration testing, while @WebMvcTest is used for testing the web layer only.

32. How do you configure custom health indicators in Spring Boot Actuator?

Create a bean implementing HealthIndicator interface and define the logic for checking the health of a specific component.

33. How can you optimize Spring Boot application performance?

Ways to optimize include using caching (@Cacheable), connection pooling, async processing (@Async), and minimizing unnecessary autowiring.

34. How does Spring Boot support reactive programming?

Spring Boot integrates with the Spring WebFlux module, enabling reactive programming using Project Reactor's Mono and Flux types.

35. How can you implement microservices using Spring Boot?

Use Spring Cloud along with Spring Boot to manage microservices architecture. Tools like Eureka (service discovery), Zuul (API Gateway), and Hystrix (circuit breaker) are commonly used in Spring Cloud.