101 Spring Framework interview questions to hire top developers

Questions

1. What is Spring Framework and why do developers use it?

2. Can you explain Inversion of Control (IoC) in simple terms?

3. What is Dependency Injection (DI) and how does it relate to IoC?

4. Name a few advantages of using the Spring Framework?

5. What are Spring Beans?

6. How do you define a Spring Bean in a configuration file?

7. What is an Application Context in Spring?

8. Explain the difference between ApplicationContext and BeanFactory.

9. What are the different ways to configure Spring Beans?

10. What is component-scanning in Spring?

11. How does Spring handle Aspect-Oriented Programming (AOP)?

12. What are Spring modules you have heard of?

13. What is Spring MVC?

14. Explain the basic flow of a request in Spring MVC.

15. What is a DispatcherServlet?

16. What are Spring Controllers?

17. How do you handle form submissions in Spring MVC?

18. What are the different types of Dependency Injection?

19. Can you describe how Spring simplifies database interactions?

20. How can you define scopes for beans in Spring, and why is it important?

21. What's the big deal about Spring Framework? Like, why do developers even bother using it?

22. Imagine you're building a Lego castle. How does Spring help you organize all the Lego blocks (components)?

23. Can you explain dependency injection in simple terms? Pretend I'm a kid who loves building things.

24. What's the difference between getBean() and @Autowired in Spring?

25. What's an ApplicationContext? Is it just a fancy name for something simple?

26. Why would I choose Spring MVC over writing servlets directly?

27. Explain what an 'aspect' is in Spring AOP, as if you were describing a superpower.

28. If a Spring bean is a singleton, what does that *really* mean? Does it mean only one can exist, ever?

29. What is the difference between constructor injection and setter injection? When might you choose one over the other?

30. Describe the lifecycle of a Spring bean from creation to destruction.

31. What's the purpose of the @Component annotation in Spring?

32. How can you define configuration metadata in Spring? What are the different ways?

33. Explain the concept of 'inversion of control' (IoC) in Spring.

34. What are some advantages of using Spring Boot?

35. How does Spring simplify database interactions? What's a JdbcTemplate?

36. If you had to explain Spring Data JPA to a non-technical person, how would you do it?

37. What are the different scopes available for a Spring bean?

38. How do you handle exceptions in a Spring MVC application?

39. What are some benefits of using Spring's transaction management?

40. Explain how you can validate user input in a Spring MVC application.

41. How can you create RESTful APIs using Spring?

42. What are some of the modules in the Spring Framework?

43. What is the purpose of Spring Security?

44. What are some advantages of using annotations in Spring?

45. How does Spring's AOP module help in separating cross-cutting concerns, and can you provide a real-world example where it's particularly useful?

46. Explain the difference between @Component, @Service, and @Repository annotations in Spring. Why would you choose one over the others?

47. Describe the purpose of Spring's DispatcherServlet. How does it process incoming HTTP requests?

48. What are Spring Boot Actuators and how do they aid in monitoring and managing a Spring Boot application?

49. Can you explain how Spring Security works and how you would implement authentication and authorization in a Spring application?

50. How does Spring Data JPA simplify database interactions? Explain with an example.

51. What is Spring's transaction management, and how does it ensure data consistency in a multi-threaded environment?

52. Explain the role of BeanPostProcessors in Spring. When would you need to implement a custom BeanPostProcessor?

53. How does Spring's dependency injection (DI) work? What are the advantages of using DI over traditional object creation?

54. Describe the difference between constructor injection and setter injection. What are the pros and cons of each approach?

55. What is Spring's Bean lifecycle? Explain the different stages a bean goes through from creation to destruction.

56. How would you handle exceptions in a Spring MVC application? Explain different approaches with code examples.

57. What are Spring profiles and how can you use them to configure different environments (e.g., development, testing, production)?

58. Explain the use of Spring's RestTemplate. How would you consume a RESTful API using RestTemplate?

59. How does Spring support integration testing? What are some best practices for writing integration tests in Spring?

60. What are message queues, and how does Spring integrate with message queue systems like RabbitMQ or Kafka?

61. Explain the concept of reactive programming. How does Spring WebFlux enable reactive web applications?

62. How can you secure a Spring Boot application using OAuth 2.0?

63. What is Spring Cloud, and what are some of its key components for building microservices?

64. Describe the purpose and usage of Spring's @Async annotation. What are the considerations when using asynchronous methods?

65. How do you implement caching in a Spring application? What are some different caching providers you can use with Spring?

66. What is the role of Spring's ApplicationContext? How does it differ from a BeanFactory?

67. Explain the purpose of Spring Data REST. How does it simplify the creation of RESTful APIs for data repositories?

68. How would you implement a custom validator in Spring? When would you need a custom validator?

69. What are Spring WebSockets, and how do they enable real-time communication in web applications?

70. Explain the concept of Spring Batch. What are the key components involved in a batch processing job?

71. How can you configure Spring to use different data sources based on the environment?

72. How does Spring Boot's auto-configuration work, and how can you customize or disable it?

73. Explain Spring Data JPA and how it simplifies database interactions. What are some common best practices when using it?

74. Describe the different types of Spring AOP pointcuts and provide examples of when you might use each.

75. How can you implement security in a Spring Boot application using Spring Security?

76. What are Spring Cloud Config and Spring Cloud Gateway, and how do they help with microservices architecture?

77. Explain how Spring's transaction management works, including different transaction propagation levels.

78. How do you handle exceptions in Spring MVC, and what are the pros and cons of different approaches?

79. Describe the role of Spring's ApplicationContext and how it differs from a BeanFactory.

80. Explain how Spring Batch helps in processing large volumes of data. What are some key components?

81. How can you monitor and manage a Spring Boot application in production?

82. What are the advantages and disadvantages of using Spring WebFlux over Spring MVC for building reactive applications?

83. Describe your experience with testing Spring applications, including unit, integration, and end-to-end tests.

84. Explain how Spring integrates with messaging systems like Kafka or RabbitMQ.

85. How do you handle caching in Spring, and what are some common caching providers?

86. Describe how you would implement a custom Spring Boot starter.

87. How does Spring's dependency injection work, and what are the different ways to configure it (annotations, XML, Java config)?

88. Explain how to use Spring's RestTemplate or WebClient to consume RESTful APIs.

89. What are the different scopes of Spring beans, and when would you use each scope?

90. Describe your experience with Spring Cloud's service discovery and load balancing components (e.g., Eureka, Ribbon).

91. How can you optimize the performance of a Spring Boot application?

92. Explain how to configure and use Spring's internationalization (i18n) support.

93. Describe your experience with Spring Data REST and how it simplifies the creation of RESTful APIs from JPA repositories.

94. How would you implement a custom health indicator in Spring Boot?

95. Explain how to use Spring's TaskExecutor and TaskScheduler for asynchronous task execution.

96. Describe your experience with securing Spring applications using OAuth 2.0 and JWT.

97. How can you implement distributed tracing in a Spring Cloud microservices environment?

98. Explain how to use Spring's integration testing support to test your application components in isolation.

99. Describe your experience with Spring's WebSocket support for building real-time applications.

100. How would you handle versioning of REST APIs built with Spring MVC or Spring WebFlux?