JAVA FULL STACK

Basic Java Full Stack Interview Questions (1-25)

1. **What is Java?**

   - Java is a high-level, object-oriented programming language developed by Sun Microsystems. It is used for developing applications that can run on any device or operating system.

2. **What is the difference between JDK, JRE, and JVM?**

   - JDK (Java Development Kit) is a full-featured SDK for Java development. JRE (Java Runtime Environment) provides libraries and JVM (Java Virtual Machine) for running Java applications. JVM is responsible for converting bytecode into machine code.

3. **What is the role of a Full Stack Developer?**

   - A Full Stack Developer is proficient in both front-end and back-end development. They work on everything from the user interface to the database and server-side logic.

4. **What is the difference between a GET and POST request in HTTP?**

   - A GET request retrieves data from the server, while a POST request submits data to the server.

5. **What is the use of Spring Framework?**

   - The Spring Framework is a comprehensive programming and configuration model for Java applications, providing support for dependency injection, aspect-oriented programming, and more.

6. **What is the difference between abstract classes and interfaces in Java?**

   - Abstract classes can have both abstract and concrete methods, while interfaces can only have abstract methods (though Java 8 allows default methods).

7. **What is the difference between ArrayList and LinkedList?**

   - ArrayList is backed by a dynamic array and provides fast access to elements, whereas LinkedList is backed by a doubly linked list, offering faster insertion and deletion.

8. **What is the difference between HashMap and TreeMap?**

   - HashMap is an unordered collection that does not maintain any order, while TreeMap is a sorted map based on the natural ordering of keys or a custom comparator.

9. **What is the use of the 'final' keyword in Java?**

   - The 'final' keyword is used to define constants, prevent method overriding, and prevent inheritance of a class.

10. **What is a constructor in Java?**

    - A constructor is a special method used to initialize objects. It is called when an object of a class is created.

11. **What is the difference between '==' and 'equals()' in Java?**

    - '==' compares references (memory addresses), while 'equals()' compares the actual content of two objects.

12. **What is polymorphism in Java?**

    - Polymorphism allows objects of different classes to be treated as objects of a common superclass. It enables one interface to be used for a general class of actions.

13. **What is the purpose of the 'static' keyword in Java?**

    - The 'static' keyword allows a method or variable to be associated with the class rather than with instances of the class.

14. **What is the Java Collections Framework?**

- The Java Collections Framework provides a set of interfaces and classes that implement commonly used data structures, such as lists, sets, and maps.

15. **What is the difference between 'String' and 'StringBuilder' in Java?**

    - String is immutable, meaning its value cannot be changed after initialization, while StringBuilder is mutable, allowing for modification of its value.

16. **What is multithreading in Java?**

    - Multithreading in Java allows concurrent execution of two or more parts of a program for maximum CPU utilization.

17. **What is the use of the 'super' keyword in Java?**

    - The 'super' keyword is used to refer to the superclass of the current object, typically to access superclass methods or constructors.

18. **What is exception handling in Java?**

    - Exception handling in Java is the mechanism to handle runtime errors using try, catch, and finally blocks.

19. **What is a Java package?**

    - A package in Java is a namespace that organizes classes and interfaces, making code modular and easy to manage.

20. **What is a session in web applications?**

    - A session is a mechanism that allows the server to maintain state across multiple HTTP requests from the same user.

21. **What is the use of 'synchronized' keyword in Java?**

- The 'synchronized' keyword is used to ensure that a method or block of code is accessed by only one thread at a time, ensuring thread safety.

22. **What is RESTful API?**

    - A RESTful API is an architectural style for designing networked applications, using HTTP methods (GET, POST, PUT, DELETE) for communication.

23. **What is the use of the 'this' keyword in Java?**

    - The 'this' keyword refers to the current instance of a class and is used to access instance variables and methods.

24. **What is the difference between a Stack and a Queue?**

    - A Stack is a last-in, first-out (LIFO) data structure, while a Queue is a first-in, first-out (FIFO) data structure.

25. **What are annotations in Java?**

    - Annotations are metadata added to Java code, which can be used by compilers or runtime environments to perform actions.

Intermediate Java Full Stack Interview Questions (26-50)

26. **What is Spring Boot?**

    - Spring Boot is a framework for building production-grade, stand-alone Spring applications with minimal configuration.

27. **What is the difference between Spring and Spring Boot?**

    - Spring provides a comprehensive set of features for enterprise-level applications, while Spring Boot simplifies the setup process with conventions over configuration.

28. **What is the Spring MVC architecture?**

- Spring MVC is a request-response model consisting of the Model, View, and Controller components. It separates concerns and allows for cleaner application design.

29. **What are microservices in Java?**

- Microservices are an architectural style where an application is divided into small, loosely coupled, independently deployable services.

30. **What is Hibernate in Java?**

- Hibernate is an ORM (Object Relational Mapping) framework that simplifies database interactions by mapping Java objects to database tables.

31. **What is the difference between @RequestMapping and @GetMapping in Spring?**

- @RequestMapping is used to map HTTP requests to handler methods of MVC controllers, while @GetMapping is a shortcut for @RequestMapping with the GET method.

32. **What are the different types of HTTP methods?**

- The different HTTP methods are GET (retrieve), POST (create), PUT (update), DELETE (remove), PATCH (partial update), and OPTIONS (request options).

33. **What is the use of the @Autowired annotation in Spring?**

- @Autowired is used to inject dependencies into Spring beans automatically, eliminating the need for explicit bean definitions.

34. **What is the purpose of the 'bean' in Spring?**

- A bean in Spring is an object managed by the Spring IoC container, and it can be injected into other objects to enable loose coupling.

35. **What is Dependency Injection in Spring?**

    - Dependency Injection (DI) is a design pattern where an object's dependencies are provided by an external source rather than the object itself.

36. **What is the difference between @Component, @Service, and @Repository in Spring?**

    - All three annotations are used to mark beans, but @Component is generic, @Service is for service-layer beans, and @Repository is for DAO-layer beans.

37. **What is the role of the controller in the Spring MVC framework?**

    - The controller handles user input, calls service methods, and returns a model and view for rendering a response.

38. **What is a REST controller in Spring?**

    - A REST controller is a type of controller in Spring MVC that handles HTTP requests and responses with RESTful services, typically returning JSON or XML.

39. **What is the difference between a List and a Set in Java?**

    - A List is an ordered collection that allows duplicates, while a Set is an unordered collection that does not allow duplicates.

40. **What is the role of the @Entity annotation in Hibernate?**

    - @Entity marks a Java class as a persistent entity to be mapped to a table in the database.

41. **What is the use of @Transactional annotation in Spring?**

    - @Transactional is used to ensure that a method or class is executed within a transactional context, providing ACID properties for database operations.

42. **What is JWT (JSON Web Token)?**

    - JWT is an open standard used to securely transmit information between parties as a JSON object, typically used for authentication.

43. **What is the purpose of a sessionFactory in Hibernate?**

    - A sessionFactory is used to create and manage Hibernate sessions, which are responsible for interacting with the database.

44. **What is the difference between @PathVariable and @RequestParam in Spring?**

    - @PathVariable is used to extract values from URI templates, while @RequestParam is used to extract query parameters from the request URL.

45. **What is a join in SQL and how is it used in Hibernate?**

    - A join in SQL combines rows from two or more tables based on related columns. Hibernate supports join operations using HQL (Hibernate Query Language) or Criteria API.

46. **What is the significance of @EnableAutoConfiguration in Spring Boot?**

    - @EnableAutoConfiguration is used to automatically configure Spring application context based on the project dependencies.

47. **What is Spring Security?**

    - Spring Security is a framework for securing Java applications, providing authentication and authorization capabilities.

48. **What is the difference between @PreAuthorize and @Secured annotations?**

  - @PreAuthorize allows for method-level security with expressions, while @Secured is a simpler way to specify roles that can access the method.

49. **What is a Spring Boot Actuator?**

    - Spring Boot Actuator provides production-ready features such as monitoring, health checks, and metrics for Spring Boot applications.

50. **How does Java handle memory management?**

    - Java handles memory management through automatic garbage collection, which reclaims unused memory.

Advanced Java Full Stack Interview Questions (51-75)

51. **What is the role of a Full Stack Developer in a microservices architecture?**

    - Full Stack Developers design, develop, and integrate both frontend and backend components of microservices-based applications.

52. **How do you manage state in a Spring Boot microservices application?**

    - State management can be done using session management, JWT tokens, or external distributed caches like Redis.

53. **What is the role of an API Gateway in a microservices architecture?**

    - An API Gateway handles request routing, load balancing, security, and rate limiting for microservices.

54. **How do you ensure data consistency in distributed systems?**

- Data consistency in distributed systems can be achieved using eventual consistency, CAP theorem, and distributed transactions.

55. **What is the use of Spring Cloud in microservices?**

    - Spring Cloud provides tools for building and deploying microservices-based applications, including service discovery, configuration management, and circuit breakers.

56. **What is the difference between synchronous and asynchronous communication in microservices?**

    - Synchronous communication involves direct calls between services, while asynchronous communication uses message queues for decoupled interaction.

57. **What is Hystrix in microservices architecture?**

    - Hystrix is a library used for fault tolerance and latency management in microservices, allowing for fallback mechanisms and circuit breakers.

58. **How do you implement versioning in RESTful APIs?**

    - Versioning in RESTful APIs can be done using URI paths, request parameters, or headers.

59. **What is the difference between monolithic and microservices architecture?**

    - In monolithic architecture, the entire application is a single unit, while microservices architecture breaks the application into smaller, loosely coupled services.

60. **What is the role of a Service Registry in a microservices architecture?**

    - A Service Registry keeps track of available services and their instances, enabling service discovery for dynamic communication between microservices.

61. **What are the advantages of using Spring Boot over traditional Spring?**

    - Spring Boot simplifies the development process by providing automatic configuration, embedded servers, and convention over configuration.

62. **What is API Gateway in Spring Cloud?**

    - The API Gateway in Spring Cloud provides a unified entry point to microservices, handling routing, security, load balancing, and monitoring.

63. **What are the different types of joins in SQL?**

    - The different types of joins are INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL JOIN, each serving different purposes in combining data from multiple tables.

64. **What is database normalization?**

    - Database normalization is the process of organizing a database to reduce redundancy and improve data integrity.

65. **What is the purpose of a load balancer in microservices?**

    - A load balancer distributes incoming traffic across multiple instances of a service to ensure high availability and scalability.

66. **What is the role of Eureka in Spring Cloud?**

    - Eureka is a service discovery tool that helps microservices register themselves and discover other services.

67. **What is the difference between a REST API and a SOAP API?**

    - REST is lightweight, uses HTTP methods, and is stateless, while SOAP is a protocol that is more complex and typically uses XML for message exchange.

68. **What is the role of a circuit breaker in microservices?**

    ○ A circuit breaker prevents service failures from cascading through the system, allowing for fallback mechanisms and reducing the impact of failures.

69. **What is the difference between a monolithic and modular Java application?**

    ○ A monolithic application is a single, tightly-coupled unit, while a modular application is broken into smaller, independent modules for easier maintainability.

70. **What is Cloud Foundry and how does it relate to Java?**

    ○ Cloud Foundry is an open-source platform-as-a-service (PaaS) for deploying and managing applications, including Java-based ones.

71. **What is Docker and how does it benefit Java development?**

    ○ Docker is a platform for developing, shipping, and running applications in containers. It helps in packaging Java applications and their dependencies for consistent environments.

72. **What is Kubernetes and how does it help in deploying Java applications?**

    ○ Kubernetes is an open-source system for automating containerized application deployment, scaling, and management.

73. **What is the difference between a primary key and a foreign key?**

    ○ A primary key uniquely identifies records in a table, while a foreign key establishes a relationship between two tables.

74. **What are the advantages of using NoSQL databases like MongoDB over relational databases?**

    - NoSQL databases are more flexible in schema design, handle unstructured data, and offer better scalability for large datasets.

75. **What is the role of a message broker in microservices?**

    - A message broker facilitates asynchronous communication between services by routing messages through queues and topics.

Technical Java Full Stack Interview Questions (76-100)

76. **What is the difference between a thread and a process in Java?**

    - A process is a self-contained execution environment, while a thread is a smaller unit of a process that shares the same resources.

77. **How do you implement pagination in a Spring Boot application?**

    - Pagination can be implemented using Spring Data JPA's `Pageable` interface to retrieve a subset of records from the database.

78. **What are Java's access modifiers and how do they work?**

    - Java provides four access modifiers: public, protected, private, and default (no modifier), which control the visibility of classes and members.

79. **How do you implement an authentication system in a Spring Boot application?**

    - Authentication in Spring Boot can be implemented using Spring Security, with support for OAuth2, JWT, or basic authentication.

80. **What are some common performance optimizations you can make in Java?**

    - Performance can be improved by using efficient data structures, avoiding unnecessary object creation, leveraging multi-threading, and using JVM tuning.

81. **What is Spring Boot DevTools?**

    - Spring Boot DevTools provides development-time features like hot swapping, automatic restarts, and live reload for a better development experience.

82. **What is the purpose of the @Configuration annotation in Spring?**

    - @Configuration is used to indicate that a class contains Spring bean definitions, typically used for defining application configuration.

83. **What is the difference between the Spring Bean lifecycle and the Java Bean lifecycle?**

    - The Java Bean lifecycle is simpler, based on getter/setter methods. In Spring, the Bean lifecycle is managed by the Spring container with additional hooks like initialization and destruction methods.

84. **How do you handle database migrations in Spring Boot?**

    - Database migrations can be handled using tools like Liquibase or Flyway, which provide version-controlled SQL scripts to manage database changes.

85. **What is the purpose of the Spring Batch framework?**

    - Spring Batch is used for batch processing jobs, such as reading, processing, and writing large volumes of data in a transactional manner.

86. **What is the significance of the 'volatile' keyword in Java?**

- The 'volatile' keyword ensures visibility of changes to a variable across threads, preventing optimization by the JVM.

87. **How do you handle cross-origin resource sharing (CORS) in Spring Boot?**

- CORS can be configured in Spring Boot using `@CrossOrigin` annotations or global configuration with `WebMvcConfigurer`.

88. **How do you implement logging in a Spring Boot application?**

- Logging can be implemented using libraries like Logback or SLF4J, and configuration can be done in `application.properties` or `application.yml`.

89. **What is the use of Spring Cloud Config?**

- Spring Cloud Config provides a centralized configuration management solution for distributed applications, allowing for dynamic configuration changes.

90. **What is the role of Redis in Java-based applications?**

- Redis is an in-memory data store that can be used for caching, session management, and message brokering in Java applications.

91. **What are the main differences between SQL and NoSQL databases?**

- SQL databases are relational, use structured data, and support ACID transactions, while NoSQL databases are non-relational, handle unstructured data, and offer scalability.

92. **What is an SQL injection, and how do you prevent it in a Java application?**

- SQL injection is a security vulnerability where malicious SQL statements are executed in a database query. It can be prevented by using parameterized queries and ORM frameworks like Hibernate.

93. **What is the use of OAuth2 in a Java application?**

    - OAuth2 is an authorization framework that allows third-party services to access user resources without exposing user credentials.

94. **What is the role of Docker in Java Full Stack development?**

    - Docker allows developers to containerize Java applications, ensuring consistency across different environments and simplifying deployment.

95. **How do you handle asynchronous processing in Spring Boot?**

    - Asynchronous processing can be implemented in Spring Boot using `@Async` annotation or by leveraging message queues like RabbitMQ.

96. **What is the purpose of Spring Boot's embedded Tomcat server?**

    - Spring Boot's embedded Tomcat server allows applications to run as stand-alone applications without needing a separate servlet container.

97. **How do you optimize database queries in a Java Full Stack application?**

    - Database queries can be optimized by indexing, using query caching, and avoiding N+1 select problems.

98. **What is the difference between eager and lazy loading in Hibernate?**

    - Eager loading fetches related entities immediately, while lazy loading fetches them only when needed, reducing initial loading time.

99. **How do you ensure security in a Java Full Stack application?**

    ○ Security can be ensured using encryption, input validation, secure authentication, authorization, and protection against common threats like XSS and CSRF.

100. **What is a WebSocket and how is it used in Full Stack applications? -** A WebSocket is a protocol for full-duplex communication between client and server, used in real-time applications like chat apps or live updates.