Java Full Stack Developer - 100 Interview Questions & Answers (18+ Years Experience)

1. Q: What is circuit breaker in microservices?

A: It stops sending requests to a service that is likely to fail, improving system stability.

1. Q: What is the difference between HashMap and ConcurrentHashMap?

A: HashMap is not thread-safe. ConcurrentHashMap is thread-safe and allows concurrent reads and synchronized writes.

1. Q: Explain how Angular handles dependency injection.

A: Angular uses a hierarchical injector to create and inject services using constructor parameters.

1. Q: What are Angular lifecycle hooks?

A: ngOnInit, ngOnDestroy, ngAfterViewInit, etc. They allow you to act at different points in a component's life.

1. Q: Explain the Java Memory Model (JMM).

A: The JMM describes how threads interact through memory and what behaviors are allowed in concurrent execution.

1. Q: What is the difference between HashMap and ConcurrentHashMap?

A: HashMap is not thread-safe. ConcurrentHashMap is thread-safe and allows concurrent reads and synchronized writes.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: What is the difference between @Component, @Service, and @Repository in Spring?

A: @Component is a generic stereotype. @Service and @Repository are specialized for business and DAO layers, respectively.

1. Q: What is circuit breaker in microservices?

A: It stops sending requests to a service that is likely to fail, improving system stability.

1. Q: Explain the Java Memory Model (JMM).

A: The JMM describes how threads interact through memory and what behaviors are allowed in concurrent execution.

1. Q: What is circuit breaker in microservices?

A: It stops sending requests to a service that is likely to fail, improving system stability.

1. Q: How do you handle tight deadlines?

A: Break tasks into sprints, prioritize features, automate tests, communicate progress and blockers proactively.

1. Q: Explain how you secure REST APIs in Spring Boot.

A: Using Spring Security with OAuth2/JWT or Basic Auth. Endpoints can be secured using configuration in WebSecurityConfigurerAdapter.

1. Q: What is circuit breaker in microservices?

A: It stops sending requests to a service that is likely to fail, improving system stability.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: What is eventual consistency and where would you use it?

A: It's a consistency model used in distributed systems where updates propagate over time. Used in high-availability systems.

1. Q: Explain the Factory Design Pattern with an example.

A: It provides a way to create objects without exposing instantiation logic. Used in DAO creation.

1. Q: What is the difference between HashMap and ConcurrentHashMap?

A: HashMap is not thread-safe. ConcurrentHashMap is thread-safe and allows concurrent reads and synchronized writes.

1. Q: Explain the Java Memory Model (JMM).

A: The JMM describes how threads interact through memory and what behaviors are allowed in concurrent execution.

1. Q: Explain how Angular handles dependency injection.

A: Angular uses a hierarchical injector to create and inject services using constructor parameters.

1. Q: Explain the Java Memory Model (JMM).

A: The JMM describes how threads interact through memory and what behaviors are allowed in concurrent execution.

1. Q: What is the difference between HashMap and ConcurrentHashMap?

A: HashMap is not thread-safe. ConcurrentHashMap is thread-safe and allows concurrent reads and synchronized writes.

1. Q: What are Angular lifecycle hooks?

A: ngOnInit, ngOnDestroy, ngAfterViewInit, etc. They allow you to act at different points in a component's life.

1. Q: How do you handle tight deadlines?

A: Break tasks into sprints, prioritize features, automate tests, communicate progress and blockers proactively.

1. Q: What is the difference between @Component, @Service, and @Repository in Spring?

A: @Component is a generic stereotype. @Service and @Repository are specialized for business and DAO layers, respectively.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: How do you handle tight deadlines?

A: Break tasks into sprints, prioritize features, automate tests, communicate progress and blockers proactively.

1. Q: What is the difference between HashMap and ConcurrentHashMap?

A: HashMap is not thread-safe. ConcurrentHashMap is thread-safe and allows concurrent reads and synchronized writes.

1. Q: Explain the Java Memory Model (JMM).

A: The JMM describes how threads interact through memory and what behaviors are allowed in concurrent execution.

1. Q: What are Angular lifecycle hooks?

A: ngOnInit, ngOnDestroy, ngAfterViewInit, etc. They allow you to act at different points in a component's life.

1. Q: Explain how Angular handles dependency injection.

A: Angular uses a hierarchical injector to create and inject services using constructor parameters.

1. Q: How would you design a scalable e-commerce application?

A: Use microservices for inventory, payment, order. API Gateway, Load Balancer, DB sharding, cache with Redis.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: Explain the Factory Design Pattern with an example.

A: It provides a way to create objects without exposing instantiation logic. Used in DAO creation.

1. Q: When would you use the Strategy Pattern?

A: When you have multiple algorithms for a task. E.g., payment processing via different gateways.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: How do you identify performance bottlenecks in Java applications?

A: Use profilers like JVisualVM, analyze GC logs, DB query analysis, thread dumps.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: Explain how Angular handles dependency injection.

A: Angular uses a hierarchical injector to create and inject services using constructor parameters.

1. Q: What is the difference between @Component, @Service, and @Repository in Spring?

A: @Component is a generic stereotype. @Service and @Repository are specialized for business and DAO layers, respectively.

1. Q: What are Angular lifecycle hooks?

A: ngOnInit, ngOnDestroy, ngAfterViewInit, etc. They allow you to act at different points in a component's life.

1. Q: Explain the Java Memory Model (JMM).

A: The JMM describes how threads interact through memory and what behaviors are allowed in concurrent execution.

1. Q: What is the difference between @Component, @Service, and @Repository in Spring?

A: @Component is a generic stereotype. @Service and @Repository are specialized for business and DAO layers, respectively.

1. Q: What are Angular lifecycle hooks?

A: ngOnInit, ngOnDestroy, ngAfterViewInit, etc. They allow you to act at different points in a component's life.

1. Q: Explain how Angular handles dependency injection.

A: Angular uses a hierarchical injector to create and inject services using constructor parameters.

1. Q: Explain the Java Memory Model (JMM).

A: The JMM describes how threads interact through memory and what behaviors are allowed in concurrent execution.

1. Q: When would you use the Strategy Pattern?

A: When you have multiple algorithms for a task. E.g., payment processing via different gateways.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: Explain the Factory Design Pattern with an example.

A: It provides a way to create objects without exposing instantiation logic. Used in DAO creation.

1. Q: Explain the Factory Design Pattern with an example.

A: It provides a way to create objects without exposing instantiation logic. Used in DAO creation.

1. Q: Explain the Factory Design Pattern with an example.

A: It provides a way to create objects without exposing instantiation logic. Used in DAO creation.

1. Q: Explain the Java Memory Model (JMM).

A: The JMM describes how threads interact through memory and what behaviors are allowed in concurrent execution.

1. Q: How do you handle tight deadlines?

A: Break tasks into sprints, prioritize features, automate tests, communicate progress and blockers proactively.

1. Q: Explain how you secure REST APIs in Spring Boot.

A: Using Spring Security with OAuth2/JWT or Basic Auth. Endpoints can be secured using configuration in WebSecurityConfigurerAdapter.

1. Q: How do you identify performance bottlenecks in Java applications?

A: Use profilers like JVisualVM, analyze GC logs, DB query analysis, thread dumps.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: How do you handle tight deadlines?

A: Break tasks into sprints, prioritize features, automate tests, communicate progress and blockers proactively.

1. Q: Explain the Java Memory Model (JMM).

A: The JMM describes how threads interact through memory and what behaviors are allowed in concurrent execution.

1. Q: When would you use the Strategy Pattern?

A: When you have multiple algorithms for a task. E.g., payment processing via different gateways.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: When would you use the Strategy Pattern?

A: When you have multiple algorithms for a task. E.g., payment processing via different gateways.

1. Q: What is the difference between @Component, @Service, and @Repository in Spring?

A: @Component is a generic stereotype. @Service and @Repository are specialized for business and DAO layers, respectively.

1. Q: How would you design a scalable e-commerce application?

A: Use microservices for inventory, payment, order. API Gateway, Load Balancer, DB sharding, cache with Redis.

1. Q: What is eventual consistency and where would you use it?

A: It's a consistency model used in distributed systems where updates propagate over time. Used in high-availability systems.

1. Q: Explain how Angular handles dependency injection.

A: Angular uses a hierarchical injector to create and inject services using constructor parameters.

1. Q: Explain the Java Memory Model (JMM).

A: The JMM describes how threads interact through memory and what behaviors are allowed in concurrent execution.

1. Q: When would you use the Strategy Pattern?

A: When you have multiple algorithms for a task. E.g., payment processing via different gateways.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: How do you identify performance bottlenecks in Java applications?

A: Use profilers like JVisualVM, analyze GC logs, DB query analysis, thread dumps.

1. Q: Explain how Angular handles dependency injection.

A: Angular uses a hierarchical injector to create and inject services using constructor parameters.

1. Q: How do you handle tight deadlines?

A: Break tasks into sprints, prioritize features, automate tests, communicate progress and blockers proactively.

1. Q: Explain the Factory Design Pattern with an example.

A: It provides a way to create objects without exposing instantiation logic. Used in DAO creation.

1. Q: When would you use the Strategy Pattern?

A: When you have multiple algorithms for a task. E.g., payment processing via different gateways.

1. Q: Explain how Angular handles dependency injection.

A: Angular uses a hierarchical injector to create and inject services using constructor parameters.

1. Q: What is the difference between @Component, @Service, and @Repository in Spring?

A: @Component is a generic stereotype. @Service and @Repository are specialized for business and DAO layers, respectively.

1. Q: Explain the Factory Design Pattern with an example.

A: It provides a way to create objects without exposing instantiation logic. Used in DAO creation.

1. Q: What are Angular lifecycle hooks?

A: ngOnInit, ngOnDestroy, ngAfterViewInit, etc. They allow you to act at different points in a component's life.

1. Q: When would you use the Strategy Pattern?

A: When you have multiple algorithms for a task. E.g., payment processing via different gateways.

1. Q: What are Angular lifecycle hooks?

A: ngOnInit, ngOnDestroy, ngAfterViewInit, etc. They allow you to act at different points in a component's life.

1. Q: Explain the Factory Design Pattern with an example.

A: It provides a way to create objects without exposing instantiation logic. Used in DAO creation.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: What is eventual consistency and where would you use it?

A: It's a consistency model used in distributed systems where updates propagate over time. Used in high-availability systems.

1. Q: What is eventual consistency and where would you use it?

A: It's a consistency model used in distributed systems where updates propagate over time. Used in high-availability systems.

1. Q: How do you handle tight deadlines?

A: Break tasks into sprints, prioritize features, automate tests, communicate progress and blockers proactively.

1. Q: When would you use the Strategy Pattern?

A: When you have multiple algorithms for a task. E.g., payment processing via different gateways.

1. Q: How would you design a scalable e-commerce application?

A: Use microservices for inventory, payment, order. API Gateway, Load Balancer, DB sharding, cache with Redis.

1. Q: What are Angular lifecycle hooks?

A: ngOnInit, ngOnDestroy, ngAfterViewInit, etc. They allow you to act at different points in a component's life.

1. Q: Explain the Java Memory Model (JMM).

A: The JMM describes how threads interact through memory and what behaviors are allowed in concurrent execution.

1. Q: How would you design a scalable e-commerce application?

A: Use microservices for inventory, payment, order. API Gateway, Load Balancer, DB sharding, cache with Redis.

1. Q: Tell me about a time you resolved a team conflict.

A: Facilitated a meeting where each party expressed their concerns. Reached consensus by aligning goals with business needs.

1. Q: How do you identify performance bottlenecks in Java applications?

A: Use profilers like JVisualVM, analyze GC logs, DB query analysis, thread dumps.

1. Q: Explain how Angular handles dependency injection.

A: Angular uses a hierarchical injector to create and inject services using constructor parameters.

1. Q: How do you identify performance bottlenecks in Java applications?

A: Use profilers like JVisualVM, analyze GC logs, DB query analysis, thread dumps.

1. Q: Explain how you secure REST APIs in Spring Boot.

A: Using Spring Security with OAuth2/JWT or Basic Auth. Endpoints can be secured using configuration in WebSecurityConfigurerAdapter.

1. Q: How do you handle tight deadlines?

A: Break tasks into sprints, prioritize features, automate tests, communicate progress and blockers proactively.