Complete Java/Spring Boot Interview Preparation Guide 2025

📚 Study Plan Overview

- **Beginner Level**: 2-3 weeks preparation
- **Intermediate Level**: 3-4 weeks preparation
- **Advanced Level**: 4-6 weeks preparation

🟢 LEVEL 1: BEGINNER (0-2 Years Experience)

Core Java Fundamentals

Basic Concepts

1. What is the difference between JDK, JRE, and JVM?
2. What are the main features of Java (OOP principles)?
3. What are primitive data types in Java?
4. What is autoboxing and unboxing?
5. What are access modifiers in Java (private, protected, public, default)?
6. What is a package in Java?
7. What is a static keyword? Static variables and methods?
8. What is a Constructor in Java? Types of constructors?
9. Can we use static methods in a Constructor?

OOP Concepts

10. What is inheritance? Types of inheritance in Java?
11. What is polymorphism? Give examples
12. What is encapsulation and abstraction?
13. What is method overloading vs method overriding?
14. When to use Interface vs Abstract Class?
15. What is a Marker Interface? Why use it?

String Handling

16. What is the difference between String, StringBuilder, and StringBuffer?
17. Difference between creating String with literal and new operator

18. What is String pool and how does it work?
19. What are the advantages of String pool?
20. Why String is immutable?

Basic Collections

21. What is a Collection in Java?
22. Difference between Array and ArrayList?
23. What is the difference between ArrayList and LinkedList?
24. What is a HashMap? Basic operations?
25. What is the difference between HashSet and TreeSet?
26. What is an Iterator?
27. What is the difference between List and Set?
28. Array vs List
29. Set vs List

Exception Handling

30. What is an Exception? Types of exceptions?
31. What is the difference between checked and unchecked exceptions?
32. What is try-catch-finally block?
33. Can we use try without catch?
34. What is throw vs throws?
35. What are the use cases of creating user-defined exceptions?
36. How to handle user-defined exception?
37. What is a NullPointerException & how to prevent it?
38. What is a ClassCastException?
39. What is Error vs Exception?

Basic Control Structures

40. Explain the difference between break and continue statements
41. What is the difference between == and .equals()?

Spring Boot Basics

Core Concepts

42. What is Spring Boot?
43. What are the advantages of Spring Boot?
44. What is @SpringBootApplication annotation?
45. What is the difference between @Component, @Service, and @Repository?
46. What is dependency injection?
47. What is @Autowired?
48. What are Spring Boot starters?

49. What is application.properties file?
50. Difference between Spring and Spring Boot?
51. What is IOC container in Spring?

REST API Basics

52. What is HTTP? Common HTTP methods?
53. What are HTTP status codes (200, 404, 500)?
54. What is REST API?
55. What is JSON?
56. What is @RestController vs @Controller?
57. What is the process of creating a REST API (with example)?

🟡 LEVEL 2: INTERMEDIATE (2-4 Years Experience)

Advanced Core Java

Collections Deep Dive

58. Contract between hashCode() and equals() methods?
59. Explain the internal working of HashMap in Java
60. What happens on a HashMap collision?
61. What is the load factor in HashMap?
62. Difference between HashMap, LinkedHashMap, and TreeMap?
63. Difference between HashMap, LinkedHashMap, and ConcurrentHashMap
64. How can you convert a HashMap into an ArrayList?
65. What are the differences between Comparable and Comparator?
66. What is a ConcurrentHashMap? How does it work?
67. What's the difference between Hashtable and ConcurrentHashMap?
68. What is the difference between Vector and ArrayList?
69. HashMap vs HashTable
70. ArrayList vs LinkedList

Java 8 Features

71. What is a Functional Interface?
72. Explain Java 8 features (Lambdas, Streams, Optional)
73. Difference between map() vs flatMap() in Java 8?
74. What is Stream API and its advantages?
75. What is Stream pipeline?
76. Difference between intermediate and terminal operators in Stream API
77. Using Stream API, find the 2nd highest salary from employee objects list

78. Use of Stream API in projects
79. Lambda expressions vs Anonymous classes

Advanced OOP

80. What is a Functional Interface?
81. What are Sealed Classes (Java 17)?
82. What is Fail-Fast Iteration & how to handle it?
83. What is ConcurrentModificationException? Fail safe and fail fast iterators?
84. Final, finally and finalize()?
85. Autoboxing vs Unboxing
86. What is Cloneable? Deep clones vs shallow clones
87. Shallow Copy vs Deep Copy

Memory Management

88. Explain Java memory management
89. What is garbage collection in Java? Types of GC?
90. What is the finalize() method?
91. Stack vs Heap Memory Allocation
92. How to handle OutOfMemoryException?

Multithreading Basics

93. What is multithreading in Java? Thread lifecycle?
94. What is synchronization in Java? synchronized keyword?
95. What is the difference between wait() and sleep()?
96. What is the volatile keyword?
97. What is ThreadLocal?
98. What is a deadlock in Java? How can it be avoided?
99. Thread vs Runnable
100. What is the difference between Runnable and Callable?
101. Synchronized methods vs Synchronized blocks

Spring Boot Intermediate

Advanced Configuration

102. How does Spring Boot auto-configuration work?
103. What is the role of @Configuration and @Bean?
104. How do profiles work in Spring Boot (@Profile use case)?
105. What is @Transactional, and what's the benefit?
106. How do you write a global exception handler in Spring Boot?
107. Difference between application.properties and application.yml?
108. What is the use of CommandLineRunner and ApplicationRunner?

109. How does the embedded server (Tomcat) work in Spring Boot?
110. How do you override default auto-configurations?
111. How do you handle circular dependencies in Spring Boot?

Dependency Injection

112. What is constructor injection vs field injection vs setter injection?
113. What is @Qualifier and @Primary annotation?
114. If both @Qualifier and @Primary are used, which one will take precedence?
115. How to avoid bean creation failures in dependency injection?
116. @Autowired vs @Qualifier
117. @Primary vs @Qualifier

REST API Advanced

118. @RequestMapping vs @GetMapping
119. @PathVariable vs @RequestParam
120. @PostMapping vs @PutMapping
121. PUT vs PATCH
122. @ExceptionHandler vs @ControllerAdvice

Spring Boot Annotations

123. @ComponentScan vs @EnableAutoConfiguration
124. @Configuration vs @Bean
125. @Async vs @Scheduled
126. @Cacheable vs @CacheEvict

Database & JPA Basics

127. What is ORM?
128. What is JPA? Difference between JPA and Hibernate?
129. What is an Entity in JPA?
130. What are JPA annotations (@Entity, @Id, @GeneratedValue)?
131. What is the difference between persist() and merge()?
132. What are JPA relationships (@OneToMany, @ManyToOne, etc.)?
133. What is JPQL?
134. What is lazy loading vs eager loading?
135. What is the difference between save() and saveAndFlush()?
136. In Hibernate, what is the difference between get() and load() methods?

Testing Basics

137. What is JUnit? Basic annotations?
138. What is @Mockito? When to use @Mock?

139. What is the difference between @Mock, @MockBean, and @Spy?
140. How do you test REST controllers using MockMvc?
141. What is @SpringBootTest vs @WebMvcTest?

Security Basics

142. What is Authentication vs Authorization?
143. What is Spring Security?
144. How do you secure REST APIs using Spring Security?
145. What is BasicAuth vs JWT?
146. What are HTTP status codes: 401, 403, 404, 500, 502, 503?

🔴 LEVEL 3: ADVANCED (4+ Years Experience)

JVM Internals & Performance

147. How does the JVM work internally? (Class loader, memory areas, GC)
148. What are the different types of class loaders in Java?
149. What are strong, weak, soft, and phantom references in Java?
150. How does Java handle memory leaks despite having garbage collector?
151. What are different GC algorithms? When to use which?
152. How do you tune JVM performance?
153. What is JIT compilation?
154. What are JVM memory areas (Heap, Stack, Method Area, PC Register)?
155. What is metaspace in Java 8+?
156. How do you analyze heap dumps and thread dumps?

Advanced Concurrency

157. What is the Executor framework in Java?
158. What is a Future in Java?
159. What is a ThreadPool and why should you use it?
160. What are the real differences between ReentrantLock and synchronized?
161. How does volatile differ from synchronized?
162. What's the difference between thread safety and atomicity?
163. When would you use CountDownLatch vs CyclicBarrier?
164. What is the difference between busy-waiting vs blocking vs non-blocking calls?
165. How does CompletableFuture work internally?
166. How do you avoid thread starvation?
167. How would you write a thread-safe singleton?
168. What is the Producer-Consumer pattern? Implement with BlockingQueue?
169. What is ForkJoinPool and how it differs from regular thread pools?

170. How do Java Stream operations work internally (parallel vs sequential)?
171. What is a live-lock and how is it different from deadlock?
172. What is the Java Memory Model (JMM)?
173. How does a Semaphore work? Real-life examples?
174. What is a BlockingQueue? Types and use cases?

Advanced Spring Boot

175. How does Spring manage bean lifecycle? What are the hooks?
176. What is a proxy in Spring? JDK vs CGLIB proxies?
177. What is the difference between @Autowired, @Inject, and @Resource?
178. Can you inject a prototype bean into a singleton? How?
179. What is the default scope of a bean in Spring?
180. How do you create custom auto-configuration?
181. What is Spring Boot Actuator? Important endpoints?
182. How do you implement custom health checks?
183. What is @ConditionalOnProperty, @ConditionalOnClass?
184. How do you implement custom metrics and monitoring?
185. What are the Scopes in Spring?
186. Prototype vs Request Scope?

Advanced AOP

187. What is AOP and how is it implemented in Spring Boot?
188. Explain @Before, @After, @Around, @AfterReturning, @AfterThrowing advices
189. What is a Pointcut and how do you define it?
190. What is a JoinPoint and what data can it provide?
191. How do you create custom annotations and intercept them using AOP?
192. Real-world use cases of AOP (logging, auditing, security, caching)

Advanced JPA & Database

193. What is the N+1 select problem in JPA? How to fix it?
194. What are entity graphs and how do they help?
195. When would you use @Query vs derived queries vs Criteria API?
196. How does optimistic locking work in JPA? @Version annotation?
197. What is pessimistic locking? When to use it?
198. How do you handle pagination and sorting in Spring Data JPA?
199. What are database transactions? ACID properties?
200. What is connection pooling? HikariCP configuration?
201. How do you implement database migrations with Flyway/Liquibase?
202. What is database indexing? Query optimization strategies?

Advanced Spring Security

203. How do you implement OAuth2 and JWT in Spring Boot?
204. What is a stateless session? How does JWT help?
205. How do you implement two-factor authentication (2FA)?
206. How does CSRF protection work in Spring?
207. What is the difference between pre-auth and post-auth filters?
208. How does SecurityFilterChain work in Spring Boot 3+?
209. What is UserDetailsService and how do you implement it?
210. How do you implement role-based access control (RBAC)?
211. How do you implement method-level security?
212. How do you secure microservices communication?
213. How to implement OAuth authentication in Spring Boot?

Microservices Architecture

214. What is the difference between monolithic and microservices architecture?
215. How do microservices communicate with each other?
216. What is RestTemplate? How to use it?
217. What is an API Gateway? Why is it used?
218. What is service discovery?
219. How do you implement resilience patterns (Retry, Circuit Breaker, Bulkhead)?
220. What is the Circuit Breaker pattern? Libraries like Hystrix/Resilience4j?
221. How do you implement API rate limiting?
222. What happens when one microservice becomes slow? Isolation strategies?
223. How do you handle versioning of REST APIs?
224. How do you implement distributed logging and tracing (ELK, Zipkin)?
225. What's the difference between sync and async communication?
226. What is eventual consistency? CAP theorem?
227. How do you implement saga pattern for distributed transactions?
228. How do you handle data consistency across microservices?
229. Microservices architecture vs Monolithic architecture
230. What is producer and consumer applications?

Apache Kafka

231. What are the core components of Kafka?
232. What is a Partition in Kafka? How does partitioning work?
233. What is a Kafka Topic, Producer, Consumer?
234. What are Consumer Groups? How does rebalancing work?
235. What if a Kafka consumer keeps retrying endlessly? Dead letter queue?
236. How do you ensure message ordering in Kafka?
237. What is Kafka Connect? Use cases?
238. How do you handle exactly-once delivery semantics?
239. What is Kafka Streams? When to use it?
240. How do you monitor Kafka performance?

241. Explain Kafka and how it handles real-time message processing

Advanced Testing

242. What's the difference between unit, integration, and E2E testing?
243. How do you mock external REST APIs in tests?
244. What are TestContainers? How to use with Spring Boot?
245. How do you write parameterized tests in JUnit 5?
246. What is the role of @DataJpaTest, @WebMvcTest, @SpringBootTest?
247. How do you implement test data builders and object mothers?
248. What is contract testing? Pact framework?
249. How do you implement performance testing?
250. What is mutation testing?

🎯 EXPERT LEVEL (5+ Years / Architect Level)

System Design & Architecture

251. How do you design a scalable e-commerce system?
252. How do you implement distributed caching (Redis, Hazelcast)?
253. What is CQRS pattern? When to use it?
254. How do you implement event sourcing?
255. What is database sharding? Strategies?
256. How do you design for high availability?
257. What is load balancing? Different strategies?
258. How do you implement API Gateway patterns?
259. What is strangler fig pattern for legacy system migration?
260. How do you design monitoring and alerting systems?
261. How does Redis caching work, and when should it be used?

DevOps & Cloud

262. How do you containerize a Spring Boot app with Docker?
263. What is Docker compose? Multi-stage Docker builds?
264. How do you create a Jenkins pipeline for Java apps?
265. What is the difference between blue-green and rolling deployments?
266. How do you implement zero-downtime deployment?
267. What is infrastructure as code? Terraform basics?
268. How do you handle secrets management (Vault, K8s secrets)?
269. What is monitoring and observability? Prometheus, Grafana setup?
270. JAR vs WAR files
271. Maven vs Gradle

272. Continuous Integration vs Continuous Deployment

Kubernetes & Cloud Native

273. What is Kubernetes? Pods, Services, Deployments?
274. How do you deploy Spring Boot apps on Kubernetes?
275. What is a sidecar container? Use cases?
276. How do you handle configuration in K8s (ConfigMaps, Secrets)?
277. What is service mesh? Istio basics?
278. How do you implement autoscaling in K8s?
279. What is ingress controller?
280. How do you implement health checks in K8s?
281. What is Helm? Chart management?
282. Cloud-native patterns for Java applications?

Performance & Troubleshooting

283. How do you profile Java applications?
284. How to debug memory leaks?
285. How to analyze thread dumps?
286. Gateway Timeout vs Service Unavailable - troubleshooting?
287. How do you optimize database queries?
288. How to handle high CPU usage in production?
289. What tools do you use for application monitoring?
290. How do you implement distributed tracing?
291. Service Not Found – how to troubleshoot?
292. How to debug locally & remotely?

🔥 MODERN JAVA FEATURES (2024-2025)

Java 21+ Features

293. What are Virtual Threads vs Platform Threads?
294. How do Virtual Threads improve performance in I/O-heavy applications?
295. When should you NOT use Virtual Threads?
296. How to create Virtual Threads using Executors.newVirtualThreadPerTaskExecutor()?
297. How do Record Patterns simplify switch expressions?
298. What are the benefits of pattern matching for instanceof?
299. What are getFirst() and getLast() methods in collections?
300. How do String Templates improve string building over concatenation?

Reactive Programming

301. What is reactive programming?
302. Difference between Mono and Flux?
303. How to handle backpressure in reactive streams?
304. When to use reactive vs traditional Spring MVC?
305. How to chain multiple asynchronous operations with CompletableFuture?
306. Reactive programming vs Imperative programming

📝 SCENARIO-BASED QUESTIONS

Performance Issues

307. Your Spring Boot application takes 45 seconds to start up in production, but only 5 seconds locally. How do you investigate?
308. After deploying to production, your REST APIs are responding in 5+ seconds. What's your debugging approach?
309. Your application works fine with 100 users but crashes with OutOfMemoryError with 1000 users. How do you solve this?
310. Your microservice occasionally returns 503 Service Unavailable during peak traffic. What could cause this?

Security Scenarios

311. Users get logged out every 30 minutes despite JWT tokens being valid for 24 hours. What's happening?
312. Your API is being hit by 10,000 requests per second from the same IP. How do you implement rate limiting?
313. You need to implement SSO for 20 different applications. How do you design this?

Database Issues

314. Your e-commerce app takes 30 seconds to load product catalog with 10,000 products. How do you optimize?
315. You're getting 'LazyInitializationException' in production but not in tests. Why and how do you fix it?
316. Two users are trying to update the same product simultaneously. How do you handle this?

Microservices Challenges

317. You need to migrate a monolithic application with 2 million users to microservices. What's your strategy?
318. Service A depends on Service B, which depends on Service C. Service C is down. How do you prevent cascade failures?
319. You have an order that touches 5 different microservices. If one fails after others succeed, how do you handle this?

🎭 BEHAVIORAL & LEADERSHIP QUESTIONS

Technical Leadership

320. How do you evaluate new technologies for adoption?
321. Describe your approach to technical debt management
322. How do you handle disagreements in technical design?
323. What's your strategy for mentoring junior developers?

Problem Solving

324. Describe a complex technical problem you solved
325. How did your technical decisions impact business metrics?
326. Describe a time you prevented a major production issue
327. How do you prioritize technical improvements vs new features?

Communication & Collaboration

328. How do you explain technical concepts to non-technical stakeholders?
329. Describe a time you had to work with a difficult team member
330. How do you handle pressure and tight deadlines?

🚀 PREPARATION STRATEGY

Week 1-2: Core Java Mastery

- Focus on questions 1-100
- Practice basic coding problems
- Understand OOP principles thoroughly
- Master collections and exception handling

Week 3-4: Spring Boot Deep Dive

- Focus on questions 101-200
- Build sample REST APIs
- Practice dependency injection scenarios
- Understand Spring Security basics

Week 5-6: Advanced Concepts

- Focus on questions 201-300
- Study system design patterns
- Practice microservices scenarios
- Learn performance optimization

Week 7-8: Modern Features & Practice

- Study Java 21+ features
- Practice scenario-based questions
- Mock interviews with peers
- Review real project experiences

Final Week: Integration & Review

- Practice all levels together
- Focus on weak areas
- Prepare STAR format answers for behavioral questions
- Review company-specific requirements

📊 QUICK REFERENCE CHARTS

Key Annotations Comparison

Annotation	Purpose	Scope
@Component	Generic stereotype	Class level
@Service	Business logic	Class level
@Repository	Data access	Class level
@Controller	Web requests	Class level
@RestController	REST APIs	Class level

HTTP Status Codes

Code	Meaning	When to Use
200	OK	Successful request
201	Created	Resource created
400	Bad Request	Invalid input
401	Unauthorized	Authentication required
403	Forbidden	Access denied
404	Not Found	Resource doesn't exist
500	Internal Server Error	Server-side error

Collection Performance

Operation	ArrayList	LinkedList	HashMap
Add	O(1)	O(1)	O(1)
Get	O(1)	O(n)	O(1)
Remove	O(n)	O(1)	O(1)
Search	O(n)	O(n)	O(1)

💡 FINAL TIPS

Before Interview

1. Research the company's tech stack
2. Prepare 2-3 detailed project examples
3. Practice coding on whiteboard/shared screen
4. Review recent Java/Spring releases

During Interview

1. Think out loud while solving problems
2. Ask clarifying questions
3. Explain trade-offs and alternatives
4. Show willingness to learn

After Interview

1. Send thank you email within 24 hours
2. Address any gaps from the interview
3. Follow up professionally

Good luck with your interview preparation! 🚀

Remember: Consistent practice and understanding concepts deeply is more important than memorizing answers.