



Java and Spring Boot Training Course: REST APIs, JPA and Microservices



Java and Spring Boot Training Course: REST APIs, JPA and Microservices

Introduction:

Java and Spring Boot training covering REST APIs, JPA and microservices is a 5-day course for application developers, back-end engineers and technical leads on enterprise service teams, ending with a tested REST microservice packaged as a container image. Organisations running business systems on Java often carry services built on older idioms, hand-wired configuration and untested data access, so changes are slow and outages follow releases. Nominees already write code in an object-oriented language at work and spend each day in hands-on labs building, securing and testing Spring Boot services. CoreConcept Training Center delivers this Java and Spring Boot course.

Course Objectives:

- Write maintainable Java code using records, sealed types, pattern matching, the collections framework, lambdas and streams
- Set up Maven or Gradle builds and structure Spring Boot applications with dependency injection, auto-configuration and environment profiles
- Build REST endpoints backed by JPA entities and Spring Data repositories on a relational database, with validated input and consistent error responses
- Protect service endpoints with Spring Security and cover service and web layers with JUnit tests and Mockito mocks
- Apply microservice patterns for resilient service calls, health checks and metrics, and package a service as a container image
- Deliver a tested REST microservice with its source repository, test report and container image

Target Audience:

- Application developers responsible for building and maintaining business services in Java
 - Back-end engineers responsible for data access, integration logic and service endpoints on enterprise systems
 - Developers responsible for moving services from older Java code or other languages onto Spring Boot
 - Technical leads responsible for code reviews, build configuration and service design choices within delivery teams
 - Integration and platform engineers responsible for running, monitoring and containerising Java services
-

Course Outline:

Day 1: Modern Java Language, Object-Oriented Design and Collections

- Java SE Records, Sealed Interfaces and Pattern Matching for Switch
- Encapsulation, Interfaces and Composition in Domain Class Design
- Collections Framework Lists, Sets and Maps with Generics
- Lambda Expressions and Stream Pipelines for Filtering and Grouping
- Checked and Unchecked Exceptions with Try-With-Resources Handling

Day 2: Maven, Gradle and Spring Boot Application Architecture

- Maven and Gradle Build Files, Dependencies and Lifecycle Phases
- Spring Initializr Project Setup with Starter Dependencies
- Inversion of Control Container and Constructor Dependency Injection
- Auto-Configuration, Component Scanning and Bean Scopes in Spring Boot
- Externalised Configuration with Profiles and Application Properties Files

Day 3: REST Controllers, JPA Persistence and Input Validation

- Spring MVC REST Controllers with Request Mappings and DTOs
- JPA Entity Mapping with Hibernate for Relational Tables
- Spring Data JPA Repositories, Derived Queries and JPQL
- Transaction Boundaries and Lazy Loading Pitfalls in Hibernate Sessions
- Jakarta Bean Validation Constraints and Global Exception Handler Advice

Day 4: Spring Security, Automated Tests and Microservice Patterns

- Spring Security Filter Chain, Password Encoding and Role Rules
- JUnit Unit Tests with Mockito Mocks for Service Classes
- Spring Boot Test Slices for Web and Repository Layers
- Service-to-Service HTTP Calls with Timeouts, Retries and Circuit Breakers
- Actuator Health Endpoints and Micrometer Metrics for Service Observability

Day 5: Lab Build of a Tested REST Microservice in Containers

- Microservice Brief, Domain Model and Endpoint Contract Scoping
 - Lab Implementation of Entities, Repositories and REST Controllers
 - Lab Integration of Validation, Security Rules and Error Responses
 - Container Image Packaging with Dockerfile and Layered Spring Boot JARs
 - Tested REST Microservice Completion with Test Report and Code Review
-

Skills You Will Gain:

- Modern Java Idioms
- Spring Dependency Injection
- REST Controller Development
- Object-Relational Mapping
- Input Validation and Error Handling
- Service Endpoint Security
- Mock-Based Unit Testing
- Container Image Packaging

Why Attend This Course:

- Hand the development lead a tested REST microservice with source code, test report and container image as a reference template for new services
- Choose between Maven and Gradle, eager and lazy fetching, and blocking calls or fallbacks when designing a Java service
- Prevent N+1 query slowdowns, unhandled exceptions and unsecured endpoints that cause outages and rework after release
- Share starter project layouts, exception handler patterns and Mockito test templates with colleagues on other Java teams

Conclusion:

Back at work, the participant hands the development lead or software architect a tested REST microservice with its source repository, test report and container image. The team uses it as a reference template when deciding how new Java services are structured, how they reach the database, how failures are reported to callers and how they are packaged for deployment. After the first service built from the template reaches a test environment, the team should review test coverage, query counts per request, error response consistency and health check results.

Frequently Asked Questions FAQ:

What should participants know before the Java and Spring Boot course?

Participants should already write code in Java or another object-oriented language and be comfortable with basic SQL and the command line. A laptop able to run a Java development kit, an IDE and a container runtime is needed for the daily labs.

How does this Java and Spring Boot course differ from a full-stack web development or API management course?

This course works on the server side of Java services: language features, Spring Boot, JPA, security, tests and microservice patterns. Full-stack courses add browser interfaces in JavaScript frameworks, and API management courses cover gateways, developer portals and API governance rather than writing service code.

Why do Java teams use Spring Boot for REST microservices?

Spring Boot supplies starter dependencies, auto-configuration and an embedded web server, so a Java service runs as a single executable file with little setup. Its Actuator adds health checks and metrics, which suits microservices that are deployed, monitored and scaled independently.

What do participants take back from the Java and Spring Boot course?

Participants take back a tested REST microservice with its source repository, JUnit and Mockito test suite, test report and container image, plus starter build files, an exception handler pattern and a security configuration to reuse on the next Java service.
