

Full-Stack Web Development with Angular, Spring Boot and Docker

Intermediate development and Linux deployment workshop

FA-0545 · 5 days

Software Development · DevOps, Cloud & Infrastructure

Build a guided full-stack application while reviewing Java, TypeScript, Angular, Spring Boot, containers and the Linux skills needed for deployment.

Introduction

This five-day intermediate workshop brings Angular and a Java/Spring Boot backend together in a guided web-application exercise. It reviews the relevant language fundamentals, develops a REST API with persistence, introduces Docker workflows and covers the Linux tasks needed to run the application in a test environment.

The course is intensive and assumes prior programming experience. Language basics are a focused review, not a complete introduction to two languages. Linux administration and ancillary services are treated selectively or as demonstrations so that the main emphasis remains on application development and deployment.

Use a supported RHEL or Rocky Linux training environment and mutually compatible Java, Spring Boot, Node.js and Angular versions. Legacy CentOS Linux is not the default platform. Activities take place on approved training machines and isolated test servers, not production systems.

Learning outcomes

The course teaches participants to:

- Use Linux command-line tools, file permissions, package-management and remote-access workflows for application support.

- Run and connect Docker containers using port mappings, environment variables, networks and volumes.
- Apply Java object-oriented and collection concepts to a layered backend.
- Build and exercise REST endpoints with Spring Boot and Spring Data JPA.
- Create TypeScript classes and Angular components and connect the frontend to the backend.
- Prepare and troubleshoot a guided test deployment, including configuration, logging and basic security considerations.

Prerequisites

- Programming experience in at least one language; familiarity with JavaScript, HTML and CSS is strongly recommended for the Angular portion.
- Comfort with a command line and basic networking, including SSH.
- Administrative access to an approved training machine where required, plus an isolated deployment/test server.
- A compatible Java JDK, build tool, Node.js/npm, Angular CLI and container runtime; versions are checked against the selected framework releases.

Training outline

1. Day 1 — Linux and Container Foundations

Linux Skills for the Training Application

- Command line and Bash; create/search/manage files and directories; executable files and environment variables.
- Inspect hardware/software, users, permissions, compression and file-transfer workflows.
- Networking basics, interface information, hostname/IP/DNS queries, curl, ping and wget.
- Packages, repositories and patch management using the tools appropriate to the chosen supported distribution.
- Secure remote access with SSH; distinguish insecure legacy remote-access protocols.
- Introduce operating-system hardening, web-service and log-management roles.

Docker Workflows

- Run containers in foreground and background and attach or execute commands for troubleshooting.
- Expose ports, configure environment variables, names and hostnames.
- Create bridge networks and communicate between containers.

- Persist data with volume mappings; practise with an isolated database container.

2. Day 2 — Java Review and Spring Application Structure

Java for the Backend

- Expressions, variables, types, casting and operators; conditionals and loops.
- Methods, parameters, return values and variable arguments.
- Classes, constructors, encapsulation, abstraction, inheritance, composition and polymorphism.
- Strings, wrapper classes, BigDecimal and immutability; date/time APIs.
- Arrays and collection choices: lists, sets, queues and maps.
- Lambda expressions, streams, common functional interfaces and method references.
- Checked/unchecked exceptions, custom exceptions and try-with-resources; file/directory APIs.

Spring Foundations

- Create a project with Spring Initializr and understand Maven's role.
- Compare tight coupling with interface-based loose coupling and dependency injection.
- Organise a flow across application layers; understand framework terminology and the roles of Spring modules and projects.
- Use a small illustrative application to trace object creation and dependency wiring.

3. Day 3 — Spring Boot, REST and Persistence

- Create a Spring Boot project and a simple API; review starters, auto-configuration and embedded servers such as Tomcat.
- Distinguish Spring Framework, Spring MVC and Spring Boot; introduce Actuator and DevTools.
- Design GET, POST, PUT and DELETE operations for an illustrative course-management resource.
- Review JDBC, Spring JDBC, JPA and Spring Data JPA; map an entity and create a repository.
- Use H2 for local exercises, seed sample data and connect the repository to REST endpoints.
- Configure a MySQL training container and connect it to the application; review persistence settings and environment configuration.

4. Day 4 — TypeScript and Angular

TypeScript Review

- Types and variables, loops and arrays; hands-on coding.

- Classes, accessors, parameter properties, inheritance, abstract classes and interfaces.
- Compiler options in tsconfig.json and module import/export.

Angular Application

- Create an Angular workspace with the CLI and inspect its project structure.
- Build components and apply conditional display, formatting and Bootstrap styling.
- Review compatible third-party integrations.
- Connect the Angular frontend to Spring Boot REST endpoints.

5. Day 5 — Full-Stack Integration and Test Deployment

- Complete and review the guided frontend/backend integration.
- Prepare the application and its container/database configuration for a Linux test environment.
- Use command-line and remote-access skills to run the test deployment and inspect logs.
- Discuss secure transport (TLS, historically called SSL), access permissions, environment settings and basic deployment hardening.
- Review troubleshooting steps, limitations and the additional engineering needed before any production rollout.

Supplementary Linux Administration Demonstrations

- Overview of network-interface tools and bonding, local package repositories and advanced package management.
- Time synchronisation using an NTP implementation such as chrony.
- Roles of web servers, central logging, mail services and directory services such as OpenLDAP.
- These source topics are contextual demonstrations or follow-on practice, not a promise of complete Linux administration coverage within the five-day development workshop.