+33 (0)1 48 10 09 46
formation@ambient-it.net
140 Avenue Jean Lolive 93500 Pantin - France

Updated 06/06/2025

Sign up

Spring Boot training

3 days (21 hours)

PRESENTATION

Our Spring Boot, Developing microservices training course will teach you to master one of the most powerful frameworks for creating microservices. Spring Boot has established itself as the Java benchmark for rapidly developing, testing and delivering applications.
cloud-native.

Discover our advanced training course for mastering microservices architectures, designed to enable you to efficiently develop, secure and deploy robust, scalable microservices.

Understand the essentials: single responsibility, autonomy and service isolation. Clearly identify the tangible benefits in terms of scalability, resilience and operational agility.

Explore the main frameworks on the market: Spring Boot, Quarkus, Micronaut and Helidon. You'll be able to compare these technologies and select the one that best meets the specific needs of your projects. Our training program will teach you why Spring Boot has become the 1?? microservices framework on the market, by comparing it with its direct competitors. You will then learn the skills needed to design, secure and deliver a Spring Boot microservice in complete autonomy.

Master the art of secure, resilient inter-service communications with practices such as CQRS, Event Sourcing, as well as advanced resilience patterns like Circuit Breaker or Retry, for maximum robustness.

Finally, delve deeper into modern security and monitoring strategies: JWT authentication, OAuth2, orchestration with Docker and Kubernetes, as well as full integration into an automated DevOps pipeline. Become an expert in the design and optimal operation of high-performance microservices architectures.

Like all our training courses, it will run on the latest version of the tool: Spring Boot.

3.4.

OBJECTIVES

- Understand microservices architecture and identify its advantages
- Analyze different frameworks and identify the most appropriate for microservice deployment
- Distinguishing between different containers (Docker, Azure) and their advantages for microservices
- Securing communication between microservices
- Developing a microservice

TARGET AUDIENCE

- Developers, architects

Prerequisites

- Knowledge of Java

Program of our Spring Boot Training: Developing microservices

Fundamentals and key principles of a Microservices architecture

- Clear definition of microservices (single responsibility, autonomy, isolation)
- Bounded Context and Domain Driven Design (DDD)
- Key benefits: scalability, resilience, upgradeability
- Communication patterns (synchronous REST/gRPC, asynchronous Event-Driven)
- API exposure with API Gateway (Zuul, Kong, Traefik equivalent)
- Centralized configuration management (e.g. Config Server, Hashicorp Vault)
- Practical workshop: breaking down a monolithic application into microservices by business domain.

Micro-services architecture and benefits

- Monolith vs. microservices: decoupling, scalability, resilience
- Bounded Contexts (DDD) & "Single Responsibility Service" principle
- Distributed transactions: sagas, CQRS, possible coherence
- Comparison of different microservices frameworks and why Spring Boot
- Participatory workshop: benchmarking different frameworks

State of the art on the main Microservice frameworks

- Spring Boot / Spring Cloud (Java, complete, mature)
- Quarkus (native Java, lightweight, fast)
- Micronaut (Java/Kotlin, very powerful)
- Helidon (native Java, lightweight, modular)
- Impact on cold-start serverless and costs
- Spring ecosystem (Data, Security, Actuator) as a factor of choice
- Participatory workshop: compare two POCs and justify Spring Boot

Advanced interaction management and Microservices resilience (CQRS & Event Sourcing)

- CQRS (Command Query Responsibility Segregation): command/query separation
- Event Sourcing: manage status and history via business events
- Advanced resilience patterns (Circuit Breaker, Retry, Bulkhead, Timeout)
- Inter-service communication (Ribbon client-side load balancing, Feign API abstraction)
- Service Discovery: concepts and tools (Eureka, Consul, DNS-based)
- Distributed tracing (Zipkin, Jaeger) for observability and debugging
- Participative workshop: Designing a microservice using CQRS and Event Sourcing (e.g. order management in an e-commerce)

Operations, security and advanced monitoring in microservices architecture

- Advanced security patterns (OAuth2, JWT, mutual TLS between microservices)
- Authorization strategies and inter-department access management
- Leader Election and coordination management (Spring Cloud Cluster equivalent, Zookeeper)
- Automated deployment and orchestrators (Docker, Kubernetes, Azure Container Apps)
- Advanced monitoring and alerting strategies (Prometheus, Grafana, ELK Stack)
- DevOps best practices: Continuous Integration & Delivery (CI/CD)
- Participative workshop: interactive quiz and practical scenario on best practices for security and monitoring in a microservices architecture.

Spring Boot express start

- Spring Initializr, starters, auto-configuration
- Actuator: health-checks, metrics, custom endpoints
- Hello-World" REST demo & unit tests
- Practical workshop: implementing a basic "Catalog" microservice

Contract-first & API management

- OpenAPI 3 specification, controller / DTO generation
- Server- and client-side JSON-Schema validation
- Spring Cloud Gateway: routing, lightweight circuit breaker, rate-limiting
- Practical workshop: writing an OpenAPI contract and scaffolding the code

Docking a Spring Boot microservice

- Dockerfile multi-stage, Buildpacks, slim images
- Environment variables, secrets and logs STDOUT
- Probes liveness, readiness, startup
- Practical workshop: building, tagging and pushing images on Docker Hub

Orchestration on Azure AKS

- AKS concepts: node pool, Managed Identity, auto-scaling
- Helm Charts, deployment Blue/Green, rollback
- Pipeline GitHub Actions: build ? push ? deploy
- Practical workshop: deploying the "Catalog" service on AKS

Securing Spring Boot APIs

- Spring Security 6: OAuth 2.1 / OpenID Connect, JWT stateless
- CORS policy, secure headers, Key Vault for secrets
- Practical workshop: protecting the Catalog API with JWT & RBAC

Secure inter-departmental communication

- Mesh Istio service: strict mTLS, mutual authentication
- Breaker & retry circuit with Resilience4j
- Tracing Zipkin/Jaeger, correlation-id
- Practical workshop: injecting Istio sidecars and activating mTLS

Build, Test, Deploy

- Unit & contract testing in CI
- Build image, push, Helm deployment, HPA autoscaling
- Grafana / Prometheus monitoring, Slack alerts
- Practical workshop: final delivery of the microservice and retrospective.

Companies concerned

This course is aimed at both individuals and companies, large or small, wishing to train their teams in a new advanced computer technology, or to acquire specific business knowledge or modern methods.

Positioning on entry to training

Positioning on entry to training complies with Qualiopi quality criteria. As soon as

On final registration, the learner receives a self-assessment questionnaire which enables us to assess his or her estimated level of proficiency in different types of technology, and his or her expectations and personal objectives for the forthcoming training course, within the limits imposed by the selected format. This questionnaire also enables us to anticipate any connection or security difficulties within the company (intra-company or virtual classroom) which could be problematic for the follow-up and smooth running of the training session.

Teaching methods

Practical course: 60% Practical, 40% Theory. Training material distributed in digital format to all participants.

Organization

The course alternates theoretical input from the trainer, supported by examples, with brainstorming sessions and group work.

Validation

At the end of the session, a multiple-choice questionnaire verifies the correct acquisition of skills.

Sanction

A certificate will be issued to each trainee who completes the course.