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

Updated on 09/10/2025

Sign up

Spring Reactor Training

3 days (21 hours)

Overview

Spring Reactor is the reactive library of the Spring ecosystem. Based on the Reactive Streams specification, it provides non-blocking APIs (Mono and Flux) for building asynchronous, scalable and resilient applications.

Our Spring Reactor training course will enable you to master stream modeling, orchestration and error handling, while integrating WebFlux, reactive clients and data access via R2DBC or MongoDB Reactive.

You'll learn how to design, test and observe high-performance reactive pipelines: Schedulers, backpressure, StepVerifier, traceability and CI/CD integration.

At the end of the course, you'll know how to develop robust WebFlux APIs, optimize latency and throughput, secure your processing and gradually migrate your services from blocking models.

As with all our training courses, this one covers the latest stable version of the Reactor ecosystem, with a resolutely practical and operational approach.

Objectives

- Understand the principles of reactive programming
- Master Mono, Flux and Schedulers
- Build high-performance Spring WebFlux APIs
- Access data via R2DBC / MongoDB Reactive
- Test with StepVerifier and instrument observability
- Industrialize : BOM, CI/CD, security and performance

Target audience

- Java / Kotlin developers
- Technical leaders and application architects

Prerequisites

- Good command of Java (or Kotlin)
- Knowledge of Spring Boot and REST APIs
- Notions of automated testing

Our Spring Reactor training program

[Day 1 - Morning]

Fundamentals of reactive programming

- Why reactive programming: non-blocking I/O, elasticity and resilience
- Reactive Streams specification: Publisher, Subscriber, Subscription
- Introducing Project Reactor and its ecosystem
- Differences with Future/CompletableFuture, RxJava, blocking code
- Choosing Reactor: use cases and limitations
- Practical workshop: starting a Maven/Gradle project and executing a first flow

[Day 1 - Afternoon]

Handling flows: Mono and Flux

- Creating Mono and Flux (factory methods, fromIterable, interval)
- Key operators: map, flatMap, filter, reduce, collectList
- Backpressure and request strategy (request, limitRate)
- Error handling: onErrorResume, onErrorReturn, retryWhen
- Introduction to StepVerifier for flow testing
- Practical workshop: transformation pipelines and unit tests with StepVerifier

Concurrency and scheduling

- Schedulers: boundedElastic, parallel, single, immediate
- publishOn vs subscribeOn: thread propagation
- Cold/Hot publishers, ConnectableFlux, autoConnect
- Combine streams: merge, concat, zip, combineLatest
- Debug, checkpoint(), reactive logs

- Practical workshop: optimizing a CPU/I/O pipeline with Schedulers

[Day 2 - Morning]

Spring WebFlux: the reactive framework

- WebFlux vs. Spring MVC: servlets vs. Netty, end-to-end non-blocking
- Annotated and functional controllers (RouterFunction, HandlerFunction)
- Reactive WebClient: timeouts, retry, backoff
- Jackson reactive serialization and validation
- SSE and WebSocket management
- Practical workshop: creating a reactive REST API + WebClient

[Day 2 - Afternoon]

Reactive data access

- R2DBC: reactive SQL drivers, transactions and limits
- MongoDB Reactive: reactive streams, tailable cursors
- Reactive repository patterns, pagination and backpressure
- Reactive cache and circuit breaker (e.g. Resilience4j)
- Context propagation strategies (tracing, security)
- Practical workshop: connecting WebFlux to R2DBC or Mongo Reactive

Reactive architecture patterns

- Fan-out/Fan-in, timeout, fallback, bulkhead
- Idempotency and in-stream retries
- Real-time processing: SSE, WebSocket, Kafka (reactor-kafka)
- End-to-end backpressure: client to base
- Observability: Micrometer, Tracing, zipkin/otel
- Practical workshop: real-time service (WebSocket + backpressure)

[Day 3 - Morning]

Quality, testing and industrialization

- Unit testing and test scheduler, virtual time
- WebFlux, WebTestClient slice tests
- Contracts, Testcontainers (Mongo/R2DBC)
- BlockHound: detection of blocking calls
- CI/CD integration, BOM Reactor, versioning
- Practical workshop: CI pipeline with reactive testing and BlockHound

[Day 3 - Afternoon] Performance,

tuning and security

- Memory, GC, batch sizes, prefetch
- Netty tuning (pools, timeouts), reactor.netty basics
- Rate limiting, circuit breaker, hedging
- Reactive security with Spring Security (non-blocking authN/authZ)
- Profiling & replays (JFR and reactive events)
- Practical workshop: WebFlux API profiling and remediation

Migration, best practices and runbook

- Migrating from Spring MVC/RxJava to WebFlux/Reactor
- Anti-patterns: block(), subscribe() wildcards, parallel() abuse
- Responsive code review guidelines, PR checklist
- Runbook: SLO/SLI, timeouts, error budgets
- Feature flags and canary deployment strategies
- Practical workshop: migration plan from MVC API to WebFlux

Companies concerned

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

Positioning on entry to training

Positioning at the start of training complies with Qualiopi quality criteria. As soon as registration is finalized, the learner receives a self-assessment questionnaire which enables us to assess his or her estimated level of proficiency in different types of technology, as well as his or her expectations and personal objectives for the forthcoming 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 training: 60% hands-on, 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.

Certification

A certificate will be awarded to each trainee who has completed the entire course.