

Updated on 06/03/2026

Sign up

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

You will learn to design, test, and monitor high-performance reactive pipelines: Schedulers, backpressure, StepVerifier, traceability, and CI/CD integration.

By the end of the course, you will be able to develop robust WebFlux APIs, optimize latency and throughput, secure your processing, and gradually migrate your services away from blocking models.

Like all our courses, this one will introduce you to **the latest stable version** of the technology and its new features.

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 implement observability

- Scaling: BOM, CI/CD, security, and performance

Target Audience

- Java / Kotlin developers
- Technical leads and application architects

Prerequisites

- Proficiency in Java (or Kotlin)
- Knowledge of Spring Boot and REST APIs
- Basic understanding of automated testing

Spring Reactive 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
- Introduction to Project Reactor and its ecosystem
- Differences from Future/CompletableFuture, RxJava, and blocking code
- Choosing Reactor: use cases and limitations
- Hands-on workshop: starting a Maven/Gradle project and running your first stream

[Day 1 – Afternoon]

Working with streams: 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 testing streams
- Hands-on 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
- Combining streams: merge, concat, zip, combineLatest
- Debug, checkpoint(), reactive logs
- Hands-on 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
- Reactive Jackson serialization and validation
- SSE and WebSocket management
- Hands-on 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 caching and Circuit Breaker (e.g., Resilience4j)
- Context propagation strategies (tracing, security)
- Hands-on workshop: Integrating WebFlux with R2DBC or Mongo Reactive

Reactive architecture patterns

- Fan-out/Fan-in, timeout, fallback, bulkhead
- Idempotence and retries in streams
- Real-time processing: SSE, WebSocket, Kafka (reactor-kafka)
- End-to-end backpressure: from client to database
- Observability: Micrometer, Tracing, Zipkin/Otel
- Hands-on workshop: real-time service (WebSocket + backpressure)

[Day 3 – Morning]

Quality, testing, and industrialization

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

Migration, best practices, and runbook

- Migrating from Spring MVC/RxJava to WebFlux/Reactor
- Anti-patterns: block(), uncontrolled subscribe(), excessive use of parallel()
- Reactive code review guidelines, PR checklist
- Runbook: SLO/SLI, timeouts, error budgets
- Feature flag strategies and canary deployments
- Hands-on workshop: migration plan from an MVC API to WebFlux

Target Audience

This training is intended for both individuals and companies, large or small, seeking to train their teams in new advanced IT technologies or to acquire specific domain knowledge or modern methodologies.

Assessment upon enrollment

The pre-training assessment complies with Qualiopi quality standards. Upon final registration, the learner receives a self-assessment questionnaire that allows us to evaluate their estimated proficiency in various types of technologies, as well as their expectations and personal goals regarding the upcoming training, within the limits imposed by the selected format. This questionnaire also enables us to anticipate certain connection issues or

internal security issues within the company (intra-company or virtual classroom) that could pose challenges for the monitoring and smooth running of the training session.

Teaching Methods

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

Organization

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

Assessment

At the end of the session, a multiple-choice questionnaire is used to verify that the skills have been properly acquired.

Certification

A certificate will be issued to each trainee who has completed the entire training program.