AMBIENT-IT

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

Updated on January 29, 2026

Register

Red Hat Fuse Training

4 days (28 hours)

Overview

Our Red Hat Fuse training course will enable you to design, deploy, and monitor modern integration flows, from simple processes (file to database) to the publication of containerized REST APIs, ready to be executed in a cloud-native platform such as OpenShift.
cloud-native platform such as OpenShift.

You will learn how to use Apache Camel, integration components, Enterprise Integration Patterns (EIP), and how to create robust and secure routes.

You will learn how to create Camel microservices in Spring Boot, containerize them with Docker, integrate them into a CI/CD pipeline, and deploy them in OpenShift with integrated monitoring. Particular attention will be paid to the modularity, maintainability, and continuous integration of business flows.

By the end of the training, you will be able to develop and operate an agile, interoperable, scalable, and DevOps-compatible integration architecture.

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

Objectives

- Understand the architecture of Red Hat Fuse and its Apache Camel core
- Create and deploy integration routes: REST, file, database, API
- Use Camel components (HTTP, JMS, Kafka, JDBC, etc.)
- Containerize and execute your integrations with Spring Boot and Docker
- Deploy your flows in OpenShift and integrate them into a CI/CD pipeline
- Monitor your integrations via Hawtio, Prometheus, and Grafana

Target audience

- DevOps
- Backend developers
- Technical architects
- Application administrators
- Any technical team looking to build a distributed, lightweight, cloud-ready ESB

Prerequisites

- Good foundation in Java and command line
- Basic knowledge of HTTP, APIs, files, and databases
- Knowledge of containerization (Docker) appreciated
- Knowledge of Kubernetes/OpenShift useful but not required

Our Red Hat Fuse training program

Introduction to Red Hat Fuse and modern integration

- The challenges of application integration in a hybrid IT environment
- Fuse vs. traditional ESBs vs. modern iPaaS
- Red Hat Fuse architecture and integrated open source components
- Apache Camel, the backbone of Fuse
- Use cases: DevOps, cloud, APIs, microservices
- Overview of complementary tools: AMQ, 3scale, OpenShift

Getting started with Apache Camel

- Structure of a Camel route: endpoint? processor? output
- Supported formats (Java DSL, XML DSL, YAML)
- Introduction to Enterprise Integration Patterns (EIP)
- Setting up a simple Maven Camel project
- Tools: IntelliJ, VS Code, Fuse Tooling, hawtio
- Workshop: Creating a local Camel route to read a CSV file, transform it, and write JSON output

Connectors and data transformation

- Camel components (HTTP, JMS, Kafka, JDBC, File, etc.)
- Adding a connector to a database
- Using Processors (Java) and beans
- Simple transformations: JSON to XML, CSV to Java object
- REST calls and structured formats (Jackson, JAXB, etc.)

Routing, logic, and error handling

- Content-Based Router, Choice, Splitter, Aggregator
- Conditional processing, filtering, transformation
- Error handling: Try-Catch, Dead Letter Channel
- Redelivery policy and circuit breaker
- Distributed business logic in flows

Creating REST APIs with Camel

- Defining an exposed REST service with Camel (REST DSL)
- Mapping routes to HTTP endpoints
- Swagger/OpenAPI: automatic document generation
- Parameter management (path, query, headers, body)
- Workshop: Creating a CRUD REST API exposed by Fuse with JSON? SQL transformation

Integration with databases

- Read/write access via JDBC or JPA
- Object/column mapping, batch queries
- Triggered by polling or events
- Integration logs, SQL error handling
- Synchronization between two databases

Containerization with Spring Boot and Docker

- Running Camel in a standalone Spring Boot project
- Configuration via application.properties and Spring profile
- Creating a Docker image of the Camel microservice
- Best practices: healthcheck, secrets, ports
- Workshop: Containerizing a Camel route in Spring Boot + Docker build + local REST test

Cloud-native deployment with OpenShift

- OpenShift concepts: pods, routes, S2I, ConfigMaps
- Deploying Camel microservices on OpenShift
- Integration with CI/CD pipelines (Git + auto build)
- Creating a public HTTP route + monitoring
- Workshop: Deploying a Camel microservice on OpenShift with dynamic configuration

Security, authentication, and filtering

- Securing a REST API with Basic Auth or OAuth2
- Header management, tokens, and input validation
- Exposure via HTTPS on OpenShift
- Controlled access to internal databases or services
- Logging and masking of sensitive data

Monitoring and supervision of routes

- Supervision with hawtio: status, metrics, logs
- Prometheus + Grafana integration
- Error detection, timeout, latency
- Audit logs and rejected messages
- Alerting via OpenShift or Prometheus

Distributed integration scenarios

- Microservices orchestration via Fuse
- Asynchronous calls, JMS or Kafka queues
- Saga or compensation patterns
- Event processing (EDA)
- Inter-system synchronization (API? Queue? DB)

DevOps best practices and industrialization

- Camel project structuring: modules, profiles
- Flow versioning, unit testing (Camel Test, JUnit)
- Integration with GitLab CI / Jenkins / ArgoCD
- Packaging in Helm charts / K8s templates
- Rollback and scaling strategies

Target companies

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

Positioning at the start of training

The positioning at the start of the training complies with Qualiopi quality criteria. Upon final registration, the learner receives a self-assessment questionnaire that allows us to assess their estimated level of proficiency in different types of technologies, as well as their expectations and personal objectives for the upcoming training, within the limits imposed by the selected format. This questionnaire also allows us to anticipate certain connection or internal security issues within the company (intra-company or virtual classroom) that could be problematic for the monitoring and smooth running of the training session.

Teaching methods

Practical training: 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.

Validation

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

Certification

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