

Nouveauté

Ref. MICR-DEV

Développer des microservices avec Java et Spring

3 jours (21 heures)

Perfectionnement

Présentiel ou classe à distance

à partir de

2 205 € HT par personne (TVA en vigueur)

 Repas inclus en présentiel

Contenu mis à jour le 25/09/2026. Document téléchargé le 02/10/2026.

Objectifs

A l'issue de cette formation, vous serez capable de :

- Concevoir et développer un microservice avec Spring Boot
- Gérer la persistance des données dans un contexte distribué
- Mettre en place la communication inter-services
- Rendre les microservices résilients et sécurisés
- Assurer l'observabilité du système
- Conteneuriser et déployer sur Kubernetes
- Automatiser la livraison avec CI/CD et GitOps.

Les compétences visées par cette formation font l'objet d'une évaluation.

À qui s'adresse cette formation ?

Développeurs, Tech Leads et architectes.

Prérequis

Connaître les fondamentaux de l'architecture microservices, maîtriser Java 11+ avec des notions en développement Spring.

Programme

1 Pendant votre formation

Développer des microservices avec Java et Spring

Jour 1 – Matin

Fondements pour la construction d'un microservice Spring Boot

- Panorama de l'écosystème Spring pour les microservices : Spring Boot 3.x, Spring Cloud, Spring Security, Spring Data
- Initialisation de projet avec Spring Initializr
- Structure de projet recommandée (Hexagonal + DDD)
- Gestion des dépendances avec Maven / Gradle

Construire un microservice REST avec Spring Boot

- Création d'un service RESTful : @RestController, @Service, @Repository
- Validation des entrées : Bean Validation (Jakarta), @Valid, @ExceptionHandler
- Documentation API avec SpringDoc OpenAPI 3 / Swagger UI
- Versioning d'API : stratégies et implémentation
- Configuration externalisée : application.yml, profils Spring, Spring Cloud Config Server

Exemple de travaux pratiques (à titre indicatif)

- Créer le microservice order-service avec CRUD complet, validation et documentation OpenAPI

Jour 1 – Après-midi

Persistence et accès aux données

- Spring Data JPA avec PostgreSQL : Entities, Repositories, Projections
- Spring Data MongoDB pour les services orientés documents
- Spring Data Redis : cache distribué et sessions
- Migration de schéma avec Flyway ou Liquibase
- Implémentation du pattern Repository en accord avec DDD

Exemple de travaux pratiques (à titre indicatif)

- Ajouter la couche de persistence à order-service, gestion des migrations avec Flyway

Tests d'un microservice

- Tests unitaires : JUnit 5, Mockito, tests du domain isolé
- Tests d'intégration : @SpringBootTest, Testcontainers (PostgreSQL, Kafka en conteneur)
- Tests de l'API REST : MockMvc, RestAssured
- Contract testing avec Pact : définir et vérifier les contrats consumer / provider
- Couverture de code avec JaCoCo

Exemple de travaux pratiques (à titre indicatif)

- *Ecrire une suite de tests complète pour order-service avec Testcontainers*

Jour 2 – Matin

Communication synchrone avec Spring Cloud

- Spring Cloud Netflix Eureka : Service Registry et Discovery
- Spring Cloud LoadBalancer : load balancing côté client
- OpenFeign : clients HTTP déclaratifs typés
- Spring Cloud Gateway : API Gateway réactif – routes, filtres, rate limiting
- Configuration du BFF pattern avec Spring Cloud Gateway

Exemple de travaux pratiques (à titre indicatif)

- *Déployer Eureka Server, enregistrer order-service et product-service, les faire communiquer via OpenFeign*

Résilience avec Resilience4j

- Circuit Breaker : états (Closed, Open, Half-Open), configuration fine
- Retry avec backoff exponentiel
- Rate Limiter et Bulkhead (isolation des thread pools)
- TimeLimiter : timeout sur les appels distants
- Intégration avec Spring Boot Actuator et Micrometer pour les métriques

Exemple de travaux pratiques (à titre indicatif)

- *Implémenter Circuit Breaker entre order-service et inventory-service avec simulation de pannes*

Jour 2 – Après-midi

Communication asynchrone avec Spring et Kafka

- Spring Kafka : producers, consumers, consumer groups
- Configuration des topics : partitions, réplication, rétention
- Sérialisation / désérialisation : JSON avec Jackson, Apache Avro avec Schema Registry
- Spring Cloud Stream : abstraction des brokers (Kafka / RabbitMQ)
- Implémentation du Saga Pattern : Choreography-based avec événements domaine
- Implémentation de l'Outbox Pattern pour garantir la fiabilité des événements

Exemple de travaux pratiques (à titre indicatif)

- *Implémenter une saga de création de commande (order-service > inventory-service > payment-service) en choreography avec Kafka*

Sécurité avec Spring Security et OAuth2

- Spring Security OAuth2 Resource Server : validation des JWT
- Intégration avec Keycloak : configuration du realm, clients, roles
- Sécurisation des endpoints : @PreAuthorize, scopes OAuth2
- Communication sécurisée entre services : OAuth2 Client Credentials flow

- Propagation du contexte de sécurité dans les appels inter-services (Feign et token relay)

Exemple de travaux pratiques (à titre indicatif)

- Sécuriser le système avec Keycloak : authentification utilisateur et communication M2M sécurisée

Jour 3 – Matin

Observabilité avec Spring Boot Actuator et OpenTelemetry

- Spring Boot Actuator : endpoints de santé, info, métriques
- Micrometer, Prometheus et Grafana : dashboards de métriques applicatives
- Distributed Tracing avec Micrometer Tracing (intégration OpenTelemetry) et Jaeger
- Logging structuré avec Logback et JSON > ELK Stack
- Corrélation des logs avec les traces (Trace ID, Span ID dans les logs)

Exemple de travaux pratiques (à titre indicatif)

- Mettre en place la stack d'observabilité complète et tracer une requête de bout en bout

Conteneurisation et Kubernetes

- Dockerfile optimisé pour Spring Boot : multi-stage build, JRE minimal (Eclipse Temurin), layer caching
- Jib (Google) : build d'images Docker sans Dockerfile depuis Maven / Gradle
- Spring Boot Docker Compose : développement local avec services externes
- Kubernetes : Deployment, Service, ConfigMap, Secret, HPA (autoscaling horizontal)
- Helm Charts : templating et gestion des releases
- Health checks Kubernetes avec les endpoints Actuator (/health/liveness, /health/readiness)

Exemple de travaux pratiques (à titre indicatif)

- Conteneuriser tous les services et les déployer sur un cluster Kubernetes local (Kind ou Minikube) avec Helm

Jour 3 – Après-midi

CI/CD et bonnes pratiques

- Pipeline GitHub Actions : build > test > analyse SonarQube > build image > push registry > déploiement
- GitOps avec ArgoCD : déploiement déclaratif depuis Git
- Stratégie Canary Deployment avec Argo Rollouts
- Spring Cloud Contract : génération de stubs et tests de contrats
- Gestion des secrets en CI/CD : HashiCorp Vault et Spring Cloud Vault
- Architecture Decision Records (ADR) : documenter les choix d'architecture

Exemple de travaux pratiques (à titre indicatif)

- Pipeline CI/CD complet pour le système de commandes : de la PR au déploiement en production simulée

Financement

Plusieurs dispositifs de financement sont accessibles via les OPCO (Opérateurs de Compétences), organismes agréés par le ministère du Travail dont le rôle est d'accompagner, collecter et gérer les contributions des entreprises au titre du financement de la formation professionnelle.

Pour plus d'information, une équipe de gestionnaires SKOLAE Formation spécialisée vous accompagne dans le choix de vos formations et la gestion administrative.

[Nous contacter pour en savoir plus →](#)

Modalités pédagogiques

Formation délivrée en présentiel ou distanciel* (blended-learning, e-learning, classe virtuelle, présentiel à distance).

Le formateur alterne entre méthode** démonstrative, interrogative et active (via des travaux pratiques et/ou des mises en situation).

Variables suivant les formations, les moyens pédagogiques mis en oeuvre sont :

- Ordinateurs Mac ou PC (sauf pour certains cours de l'offre Management), connexion internet fibre, tableau blanc ou paperboard, vidéoprojecteur ou écran tactile interactif (pour le distanciel)
- Environnements de formation installés sur les postes de travail ou en ligne
- Supports de cours et exercices

En cas de formation intra sur site externe à M2I, le client s'assure et s'engage également à avoir toutes les ressources matérielles pédagogiques nécessaires (équipements informatiques...) au bon déroulement de l'action de formation visée conformément aux prérequis indiqués dans le programme de formation communiqué.

* nous consulter pour la faisabilité en distanciel

** ratio variable selon le cours suivi

[Accessibilité aux personnes en situation de handicap](#)

Détails de la session

Du 2 décembre 2026 au 4 décembre 2026 (3 jours / 21 heures)

Lille

Du 2 décembre 2026 au 4 décembre 2026

Lille

- 2 décembre 2026 9h00 – 16h00
- 3 décembre 2026 9h00 – 16h00
- 4 décembre 2026 9h00 – 16h00