Séminaire Microservices, l'état de l'art

Durée 1 jour(s) (MICROSERVICES-ETAT-01)

Comprendre les principes et les enjeux des Microservices

Description

Formation animée en présentiel

Ce séminaire permet de comprendre le pourquoi des Architectures Microservices en présentant les enjeux et les différents cas d'utilisation.

Formation disponible en mode "formation à distance"

En inter-entreprises, l'outil de visio-conférence privilégié est Microsoft Teams

En intra-entreprises, on privilégie Zoom mais Microsoft Teams est également proposé

Objectifs

- Savoir si les architectures Microservices peuvent répondre à ses besoins
- Appréhender les différentes techniques pour découper son code métier
- Comprendre ce qu'apporte la méthode de conception logicielle Domain Driven Design (DDD)
- Connaître les principales stratégies de migration vers une cible Microservices
- Comprendre les implications d'une Architecture Microservices pour les exploitants

Public

- Développeur confirmé
- Architecte

Prérequis

Aucun

Répartition

100% Théorie, 0% Pratique

Evaluations des acquis

L'évaluation des acquis de la formation se fera en séance au travers d'ateliers, d'exercices et/ou de travaux pratiques. Dans le cas d'une formation officielle éditeur, veuillez nous consulter afin que nous vous fassions part des modalités d'évaluation.

A l'issue de la formation, vous sera transmis une évaluation à chaud de l'action de formation qui vous permettra de nous faire part de vos retours quant à votre expérience apprenant avec Zenika.

Ressources pédagogiques

Les ressources pédagogiques proviennent de productions des équipes Zenika et/ou de la documentation éditeur dans le cas d'une formation "Officielle". Les documents sont en français ou en anglais.

RQTH et ma formation Zenika

Si vous êtes sujet à un handicap, prenez contact avec nos équipes pour que nous puissions définir ensemble comment nous pourrons aménager la session afin que vous puissiez vivre une expérience en formation inchangée.

Programme

Introduction

- L'émergence des Microservices, l'aboutissement d'une maturité d'ingénierie logicielle
  - Limitations des monolithes
  - Méthodologies Agiles et pratiques DevOps
  - Méthode de conception DDD (Domain Driven Design)
  - Multiplication de la mise en place des architectures orientées service (SOA)
  - Industrialisation des processus d'intégration continue et de déploiement continu
- Première approche des architectures Microservices
- Adoption des Microservices chez les géants du Web

Des fondations de l'Architecture Logicielle

- L'architecture logicielle, clé de voûte des applications
- Facteurs de qualité pour le succès d'une architecture
  - Gestion maîtrisée des dépendances, faible couplage et forte cohésion
  - Principes DRY, KISS et YAGNI
  - Flexibilité, Sécurité, Scalabilité, Robustesse et Résilience
- Techniques de collaboration entre composants
  - Request-Reply (Synchrone ou Asynchrone)
  - Request-Reply with events
  - Request-Reply with commands/queries (CQRS)
  - Publish-Subscribe (Asynchrone)
  - Publish-Subscribe with events
  - Publish-Subscribe with commands/queries (CQRS)
- Gestion des invariants
  - Cohérence transactionnelle (invariant strict)
  - Cohérence éventuelle ('Eventual consistency')
- Renforcement du couplage faible avec les Web APIs REST
  - Fonctionnement et contraintes
  - Le véritable usage REST à travers le modèle de maturité de Richardson
  - HATEOAS vs 'Pragmatic REST'
- Architectures à base d'événements (Event-driven Architecture)

Les caractéristiques des Microservices

- Isolation de capacité Métier
- Unité d'exécution autonome
- Adaptation aux nouvelles organisations Client/Fournisseur
- Des équipes orientées Produit
  - Regroupement de compétences (Métier, DEV, QA, Ops, etc)
  - Renforcement de la culture DevOps et de la communication entre les équipes
- Tolérance avancée aux pannes ('Fail-Safe')
  - Circuit Breaker
  - Cloisons

Construction d'une Architecture Microservices

- Savoir commencer petit et évoluer graduellement
- Les différentes pistes de décomposition
  - Perspectives Métier
  - Analyse des modes de communication
  - Trouver les verbes et les noms
- Utilisation de DDD
  - Trouver les limites des services avec les Bounded Context (BC)
  - Consistance éventuelle par des Domain Events

    - Exploitation des différents patterns d'intégration du DDD Stratégique
- Choisir comment stocker les données
    - Centralisation vs décentralisation des données
    - Représentation des données adaptée à son usage
    - Gestion des données partagées
- Choix du style de collaboration
    - REST request/reply vs Publish-Subscribe Messaging
    - Combinaison des styles
- Event-Driven Microservices
    - Utilisation des architectures Event-Driven
    - Consistance des données et Event Sourcing
    - Séparation des requêtes et des commandes avec CQRS

Mise en place d'une Architecture Microservices

- Mécanisme d'inscription
    - Service Registry
    - Self-Registration
    - Third-Party Registration
- Mécanisme de découverte
    - Client-Side Discovery
    - Server-Side Discovery
- Composition des Microservices
    - Aggregator
    - Dumb & Smart Proxy
    - Chained
    - Branch
- Automatisation de la construction et Intégration Continue (CI process)

Stratégies de migration d'un système existant

- Identifier les Bounded Context (BC)
- Les autres approches

Déploiement

- Exploration des différents possibilités de déploiement
    - Multiple Service Instances per Host
    - Single Service Instance per Host
    - Service Instance per VM
    - Service Instance per Container
- Exemple de l'utilisation de Docker pour limiter la complexité opérationnelle
- Bénéfices des plateformes Cloud
- Conséquences sur un processus de livraison et déploiement continu (CD process)

Monitoring

- Observabilité du service et du système
- Importance des logs et mécanisme de corrélation
- Service Metrics
- Exemples de solutions techniques