La Programmation Réactive avec RxJava

Durée 2 jour(s) (RXJAVA-02)

Découverte et implémentation de la programmation réactive en Java

Description

Java SE a introduit dans sa version 8 une style de programmation plus fonctionnel avec les expressions lambda et l'API stream. Cela permet de tirer parti des avantages du paradigme de programmation fonctionnelle en complément de la programmation orientée objet. Un certain nombres de librairies ont vu le jour pour exploiter de nouvelles possibilités, et notamment le concept de _reactive streams_. Les _reactive streams_ permettent, comme leur nom l'indique, de réagir à des flux de données dont la volumétrie n'est pas pré-déterminée. Ils offrent ainsi une solution aux applications devant faire face à une charge de traitement variable avec une quantité de ressources limitées. La programmation réactive repose sur la gestion de d'évènements asynchrones et un modèle déclaratif qui peuvent être difficile à appréhender. Il est par ailleurs nécessaire de maîtriser l'ensemble des couches de l'architecture applicative (BDD, Service, Client) afin de savoir un flux continu de données entre un producteur et un consommateur séparés par de nombreux intermédiaires. A travers l'utilisation de _RxJava_ (implémentation en Java de _ReactiveX_, un API de programmation asynchrone très populaire), cette formation abordera la programmation réactive dans son ensemble en Java.

Objectifs

- Comprendre le paradigme de programmation réactive
- Maîtriser la librairie RxJava (Développement et Test)
- Savoir intégrer de bout en bout une architecture réactive (BDD, Service, Client)

Public

- Développement Java

Prérequis

- Connaissance du language Java (Java 8 recommandé).

Répartition

40% Théorie, 60% Pratique

Programme

Introduction

- Pourquoi la programmation réactive ?
- Histoire de la programmation réactive
- Paysage des frameworks de programmation réactive
- Histoire de RxJava

La programmation réactive

- Flux d'évènements asynchrones
- Push / Pull
- Design pattern Observer
- "Flux d'Évènements" et "Valeur au cours du temps"
- Les "marble diagrams"
- La standardisation de la programmation réactive en Java

Les bases de RxJava

- Observables / Observers / Flowable
- Créer des Observables et des Flowables : from / of / create
- S'abonner à un Observable : subscribe
- Gestion des erreurs
- Gestions des "subscriptions"
- Multi-threading

Les opérateurs : transformer des observables

- Rappels de programmation fonctionnelle
- Description d'un opérateur
- Transformer les évènements : map / scan
- Agir sur la chronologie : delay / buffer
- Filter les évènements : filter / debounce / take
- Combiner les observables : concat / merge / switch

Concepts avancés

- Hot / Cold observables
- Observables multicast
- Subjects
- Schedulers

Tester avec RxJava

- Le TestScheduler
- Créer des observables pour les tests
- Vérifier les observables

Client / Serveur

- Les protocols de communication
- Faire des appels côté client
- Bonnes pratiques

RxJS côté back-end

- Les "drivers" de base de données
- Utilisation dans les "Message Oriented Middleware" (MoM
- Intégration dans Spring Framework

Conclusion