La Programmation Réactive Fonctionnelle avec RxJS

Durée 2 jour(s) (RXJS-02)

Découverte et implémentation de la programmation réactive

Description

La programmation fonctionnelle a le vent en poupe dans l'écosystème JavaScript. Elle est mise en avant par des bibliothèques comme React, Redux ou Lodash et de nouveaux langages de programmation comme Elm ou Clojure. Le concept de _Programmation Réactive_ est très lié au paradigme fonctionnel et fait de plus en plus parler de lui grâce à son utilisation dans le framework Angular et la standardisation en cours du type _Observable_ en JavaScript. Parfois définie comme la "manipulation de flux d'évènements asynchrones", la Programmation Réactive Fonctionnelle vous permettra d'écrire du code déclaratif, maintenable et facilement testable. Cependant, il s'agit d'un nouveau paradigme pour beaucoup de développeurs et celui-ci peut être difficile à appréhender. À travers l'utilisation de RxJS (la bibliothèque la plus populaire, utilisée notamment par Angular), cette formation aborde le paradigme de programmation réactive dans son ensemble et les patterns classiques, ainsi que leur application en JavaScript.

Objectifs

- Comprendre le paradigme de programmation réactive
- Maitriser la librairie RxJS (Développement et Test)
- Savoir utiliser RxJS au sein du navigateur ou dans Node

Public

- Développeurs JavaScript

Prérequis

- Connaissance du langage JavaScript (ES6 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 RxJS

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 JavaScript

Les bases de RxJS

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

Les opérateurs : transformer des observables

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

Concepts avancés

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

Tester avec RxJS

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

RxJS dans le navigateur

- Interagir avec le DOM
- Faire des appels HTTP
- Bonnes pratiques

RxJS côté serveur

- Streams / EventEmitters vs Observables
- Interagir avec le FS
- Interagir avec le réseau

Conclusion