## Spring Framework 5.0

Speaker : Stéphane Nicoll (Pivotal)
Format : Conférence
Date : 6 avril 2017

**Résumé de Spring Framework 4.3**

- Génération 4.x releasée en 2013
- Versions successives avec la 4.3 en 2016
- Support étendu jusqu'en 2019
- Supporte
    - JDK 6 à 8
    - Tomcat 6 à 8.5
    - Websphere 7 à 9
- Possibilités d'utiliser les fonctionnalités de Java 8 avec une détection au runtime par le framework. Exemple sur l'API Date Time et de la classe @Optional

**Spring Framework 5.0**

- 5.0 M5 sorti il y'a quelques semaines
- RC1 prévue pour fin avril 2017
- GA prévu pour juin 2017
- Spring Boot 2 est basé sur Spring Framework 5
- Pré-requis revus à la hausse :
    - JDK 8 et +
    - Java EE 7 : Servlet 3.1, JMS 2, JPA 2.1
    - JUnit 5 (possibilité d'utiliser Jupiter pour continuer à utiliser JUnit 4)

## Amélioration de performances

Cible : temps de démarrage et occupation mémoire
Avec l'arrivée des conteneurs, le temps de démarrage d'une application est central.
L'un des mythes : le classpath scanning est lent.
Fonctionnalité de création d'un index. Peu d'intérêt pour gagner du temps au démarrage.
Mais finalement utilisé pour améliorer le cache.

Optimisation du path matcher dans @RequestMapping. Non activée par défaut.

Support de zéro copie : servir les ressources statiques sans solliciter le CPU.

## Support du JDK 9

- Optimisations : compact Spring + G1
- API de collections immutables

Build qui compile avec les earlier version du JDK 9

Le décalage de la sortie de Java 9 a impacté la sortie de Spring Framework 5.
Utilisation des modules automatiques de Jigsaw.
Autre déception : absence du support de http2 (module externe expérimental)

## HTTP/2

Spécs HTTP 1 sortie en 1996. Les navigateurs supportent HTTP/2. Spring Framework 5 également. Nouveau client web.
Intégration de Servlet 4 en avance.

## Reactive Spring

Nouvelle offre pour le développement d'applications web. Depuis 2 ans, beaucoup de travail a été réalisé sur ce sujet.
Aujourd'hui, un thread traite l'exécution d'une servlet. Plus il y'a de clients, plus il faut de threads. Il y'a une limite en termes de mémoire et CPU.
Reactive Progamming : se focalise sur la stabilité du serveur en cas de forte concurrence.

Sources de lenteur :

1. Lorsque les clients sont mobiles (réseau lent)
2. Latence côté service à cause d'un service qui appel d'autres services

Slides sur la recherche d'une personne attaquant une base de données de manière bloquante.
En programmation réactive, on définit un pipeline. L'information sera poussée lorsqu'elle sera disponible.

L'idée : être réactif sur tout le traitement de la requête. Tous les composants doivent être réactifs. Pas possible avec du JDBC (drivers pas disponibles et problématique des transactions).

Reactive Steams API : Publisher, Subscriber ; Subscription, Processor, Mono, Flux.
Un Publisher va pousser de l'information à un Subscriber.
Mono retourne 0 ou 1 élément. Flux retourne une collection d'éléments.

Ce modèle de programmation ne remplace pas la programmation impérative mais offre un modèle complémentaire.

**Démo**

Avec Spring Boot 2, Reactive Stream et starter Mongo DB.
Spring Data supporte Reactive Stream. Le Repository Mongo doit étendre ReactiveCrudRepository.
Le contrôleur web annotée avec @GetMapping renvoie un Flux<Speaker>

Particularité sur le save : la méthode retourne un Mono<Void>
Le @RequestBody prend un Mono<User>

Nouvelle API de client web.
Sur la classe Mono, l'opérateur **and** permet de combiner 2 Mono lorsque les flux sont disponibles.

Spring WebFlux est le pendant de Spring MVC pour les applications réactives.
L'application WebFlux doit tourner sur Netty.
Spring MVC ne va pas disparaître.

## Functional APIs

API fonctionnelle pour enregistrer des beans Spring : le contexte applicatif peut être créé de manière programmatique, sans aucune annotation.

API fonctionnelle sur WebFlux : série de building blocks à combiner. Exemple de création de toutes de manière programmatique

## Support de Kotlin

Pour aller plus loin dans l'approche fonctionnelle, Kotlin est particulièrement intéressant.
Spring Framework 5 apporte un support explicite : inférence de types, ajout d'extensions (pour RestTemplate, Spring Boot, Spring Data), …