De Java 8 à Java 11 sur un gros projet : les pièges à éviter

Speakers : Alexis Dmytryk (efluid), Thomas Collignon (efluid)
Format : Conférence
Date : 18 avril 2019

Lors de l'annonce de JigSaw à la JavaOne, les speakers se demandaient si le projet sortirait un jour.
L'application web de l'éditeur de logiciel efluid possède une base de code de 3 millions de ligne de code en Java 8 (migré en 2015).
Outils de build classique : Maven 3, Jenkins, Artifactory, SonarQube
L'application se déploie sur du WebLogic et propose également une version avec un Tomcat embarqué.

Pour la prochaine version majeure de leur logiciel prévue pour début 2020, ils ont étudié en 2018 la faisabilité de migration vers Java 11.

A première vue, aucun problème bloquant en perspective :

- Les Java modules de Java 9 ne sont pas obligatoires
- Ils n'utilisent pas d'API dépréciées
- WebLogic est édité par Oracle donc sera compatible

Quels en seraient les bénéfices ?

- Nouveauté du langage pour les développeurs
- Sécurité renforcée
- Amélioration des perfs (compact string)

Ils ont donc fait le choix de basculer sur Java 11. Ils ont commencé par utiliser Java 10 qui était déjà sorti. En septembre 2018, ils ont migré sur Java 11.

Guide de migration officiel : https://docs.oracle.com/en/java/javase/11/migrate/
Outil : https://github.com/marchof/java-almanac pour suivre les API qui ont été ajoutées ou supprimées

Choix de la distribution

OpenJDK : référentiel de code source
Plusieurs distributions d'OpenJDK :

- Oracle OpenJDK
- Oracle JDK
- AdoptOpenJDK (communautaire)
- Amazon Correto
- Alibaba

Plusieurs questions se posent lors du choix de sa distribution :

- Est-ce que les outils tels que JavaFX ou JavaMissionControl seront présents dans toutes les distributions ?
- Quel est le rythme des patchs de chaque distribution ?

Pour l'instant, efluid a opté pour Oracle OpenJDK (choix inchangé entre juin 2018 et avril 2019). Ce choix pourra évoluer.

Upgrade des outils de build

Afin de prévenir les problèmes, ils ont commencé à mette à jour tous leurs outils de build : maven 3.5.3, maven-compiler-plugin 3.8.0, suppression des références à tools.jar

Les problèmes rencontrés :

- Le plugin asciidocteor-maven-plugin ne fonctionnaient pas en JDK 11. Depuis si
- Le plugin maven-plugin-plugin n'était pas compatible : MPLUGIN-336
- Messages de warning dans le groovy-maven-plugin. Ils attendent simplement que Groovy les corrige.

Outil interne open-sourcé : test-control-byte-code permet de contrôler les classes en doublon (NB : semble ressembler à maven-duplicate-plugin) et de vérifier que toutes les méthodes appelées par le code existent dans les dépendances (JAR).

Montées de version des frameworks

- JAXB ne fait plus partie JDK : il faut ajouter manuellement les dépendances.
- Le plugin maven-cxf-codegen ne fonctionnait pas. Il a fallu attendre la version 3.3.0 sortie en janvier 2019.
- JUnit 4.12 et AssertJ 3.11.1 OK
- Migration vers Mockito 2.23 qui a un meilleur support de Java 11 que ne l'a Mockito 1
- Powermock : pas de support pour Java 9, 10 et 11. Projet en fin de vie. Par chance, peu de tests étaient basés dessus. Ils ont supprimé cette dépendanc, ce qui n'est pas plus mal car en général cela cache des mauvaises pratiques de dév
- Reflection : le projet ronmamo/reflections a été remplacés par ClassGraph

IDE

- Efluid utilise le plugin m2eclipse-webby à la place de M2
- La version Eclipse IDE 2018-09 ne fonctionnait pas avec Java 11
- Lenteur d'indexation et surcharge mémoire : blocage pendant 10 minutes, plusieurs fois par jour
    - Espace disque en RAM insuffisant
    - Patch manuel du plugin org.eclipse.jdt.core
- Problème d'import de certaines classes : import org.w3c.dom.*
- Le plugin build-helper-maven-plugin ne fonctionnait plus

- SonarLint fonctionnait mais les règles SonarQube 4.5.7 ne fonctionnait plus en Java 10 / 11. Depuis, ils ont migré à la version 7.

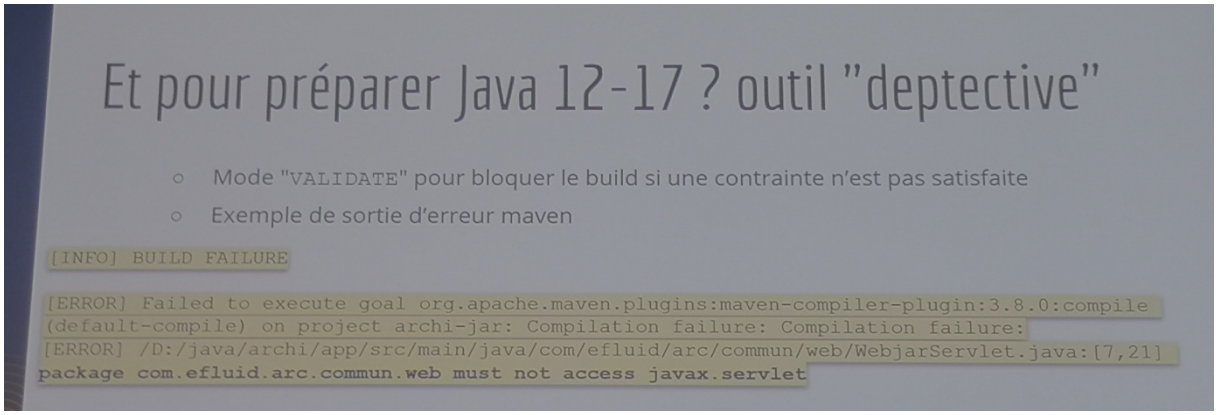
Runtime

- Efluid embedded : la composition maison créée en 2015 en embarquant Tomcat
    - Passage à Tomcat 9
- WebLogic
    - Sans application, à vide, WebLogic ne démarrait pas avec Java 10 en juiller/août 2018
    - Aucune roadmap publique sur le Net
    - Appel au support commercial de WebLogic : ils ne souhaitent supporter que les LTS. Java 11 ne sera supporté qu'à partir de juin 2019. Problématique pour efluid car support hypothétique.
    - Dans le descripteur de déploiement WebLogic, efluid doit spécifier à WebLogic tous les JAR à surcharger

Efluid décide de mettre à niveau Efluid embedded pour qu'elle soit iso-fonctionnelle avec l'ex-version WebLogic (ex : support JMS). Possibilité de repasser sur WebLogic s'ils supportent enfin Java 11

Et pour préparer Java 12-17 ?

- Ajout de règles de développement
    - Interdiction du split-package en prévision de l'activation de JigSwa
- Compilation plus régulière avec les prochaines versions du JDK : test de compatibilité incrémentale
- Mise en place de modules sur des projets un peu plus petits
    - Outil deptective : plugin Maven/Gradle permet de bloquer le build si une contrainte n'est pas satisfaite. Permet une sortie GraViz.



A retenir

- Des difficultés classiques et anticipées (quelques lignes CORBA à supprimer)
- Des difficultés plus inattendues :

    - Concernant l'IDE : ils se sont demandés s'ils n'allaient pas passés d'Eclipse à IntelliJ. Ils ont souhaité utiliser Java 11 dès le jour de sa sortie : les outils n'étaient pas encore à jour. Il faut l'accepter. Mode mixte : Eclipse et IntelliJ en fonction du développeur
    - Conteneur WebLogic : les serveurs d'application mettent toujours un peu de temps. JBoss supporte Java 11 depuis janvier 2019. OpenLiberty depuis février 2019.
- Points d'attention :
    - Eviter les projets exotiques : PowerMock, Reflections
    - Ne pas migrer top vite après la sortie : attendre 3 à 6 mois laisse le temps aux outils et frameworks de se mettre à jour.