TP4 : Java et C++

4.1 Plugin init pour Java

Utiliser le plugin *init* pour créer une application Java.

Observer les fichiers générés

Visualiser les taches disponibles

Exécuter l'application Java générée

4.2 Application SpringBoot

Créer un nouveau projet Java et reprendre les sources fournis. C'est une application web Java monolithique autonome qui utilise le framework SpringBoot.

Le framework fournit 2 plugins :

- org.springframework.boot : Il ajoute principalement une tache bootRun permettant de générer un exécutable qui lance l'application web.
- io.spring.dependency-management : Il permet de gérer les versions de toutes les dépendances utilisées par le framework (Equivalent à la balise dependencyManagement de Maven)

Appliquer ces 2 plugins en indiquant la version *2.1.3.RELEASE* pour *org.springframework.boot*

Spécifier une propriété groupe et version pour le projet

Déclarer ensuite le repository Maven et les dépendances suivantes :

- Dépendances du framework :
  - Pré-processing des annotations :
    - *org.springframework.boot:spring-boot-configuration-processor*
  - Nécessaires pour la compilation :
    - *org.springframework.boot:spring-boot-starter-data-jpa*
    - *org.springframework.boot:spring-boot-starter-thymeleaf*
    - *org.springframework.boot:spring-boot-starter-web*
    - *org.springframework.boot:spring-boot-starter-security*
  - Test
    - *org.springframework.boot:spring-boot-starter-test*
    - *org.springframework.security:spring-security-test*
    - *net.sourceforge.htmlunit:htmlunit*
  - Exécution uniquement :
    - *org.springframework.boot:spring-boot-starter-actuator*
    - *org.springframework.boot:spring-boot-devtools*
    - *org.hsqldb:hsqldb*

- Dépendances non gérées par le framework
  - Pour la compilation :
    - *io.jsonwebtoken:jjwt* (Trouver une version récente sur Maven Central)

        - *io.springfox:springfox-swagger2 :2.9.2*
    - Pour l'exécution
        - *io.springfox:springfox-swagger-ui :2.9.2*
        - *org.webjars:bootstrap:* (Indiquer une version 4.x)

Une fois indiquer les dépendances, exécuter successivement les tâches *compile*, *test*, puis *bootRun*

Exécuter la tâche générant la distribution

4.3 Init pour projets C++

Créer un répertoire de travail et y exécuter *gradle init*

Suivre l'assistant pour démarrer un projet de type Application C++

Visualiser les fichiers générés :

- Le wrapper
- *settings.gradle*
- *build.gradle* : plugins utilisés et machine cible
- la structure du projet

Visualiser les tâches disponibles et Exécuter un build

Refaire la même chose pour une librairie C++

4.4 Projet Librairie C++

Récupérer les sources fournis

Mettre au point le fichier Gradle pour appliquer les plugins :

- *cpp-library*
- *cpp-unit-test*
- *maven-publish*

Définit plusieurs machines cibles pour cette librairie.

Spécifier une propriété groupe et version pour le projet

Les fichiers de test dépendent de la librairie :
org.gradle.cpp-samples:googletest:1.9.0-gr4-SNAPSHOT
présent dans le dépôt Maven :
https://repo.gradle.org/gradle/libs-snapshots-local/

Indiquer cette dépendance dans *build.gradle*

Essayer d'exécuter la tâche *test*

Les tests utilisent la librairie *pthread* qui nécessite l'option *-lpthread* lors de la phase de link.

Configurer cette option pour les variantes correspondant à une machine Linux utilisant *gcc*

Définir un dépôt Maven local et publier vers ce dépôt, regarder les méta-données associées