

CONSTRUCTION D'APPLICATION ET GESTION DES DÉPENDANCES LOGICIELLES (BUILD AND DEPENDENCIES MANAGEMENT)

GERALD OSTER <GERALD.OSTER@TELECOMNANCY.EU>

Dans un premier temps, vous consulterez la présentation suivante (les 45 premières slides) qui constitue une bonne introduction à Gradle en indiquant l'intérêt d'un tel outillage et en effectuant une comparaison rapide avec l'existant (Ant, Maven, etc.)

(<https://fr.slideshare.net/jadsonjs/gradle-53757208>)

Ensuite, vous lirez le tutoriel sur Gradle (*Lars Vogel et Simon Scholz, 2018*) disponible à l'adresse (<http://www.vogella.com/tutorials/Gradle/article.html>), plus particulièrement les sections suivantes (vous pouvez ne pas lire les autres sections).

- (1.) Introduction to the Gradle build system - <http://www.vogella.com/tutorials/Gradle/article.html#introduction-to-the-gradle-build-system>
- (2.) Gradle plug-ins - <http://www.vogella.com/tutorials/Gradle/article.html#gradle-plug-ins>
- (5.) Dependency management for Java projects - <http://www.vogella.com/tutorials/Gradle/article.html#dependency-management-for-java-projects>
- (6.) Running a build - <http://www.vogella.com/tutorials/Gradle/article.html#running-a-build>
- (7.1.) Gradle Tasks - Default Gradle tasks - http://www.vogella.com/tutorials/Gradle/article.html#gradle_tasks
- (15.) Building Java projects - <http://www.vogella.com/tutorials/Gradle/article.html#building-java-projects>
- (17.) Testing with Gradle - http://www.vogella.com/tutorials/Gradle/article.html#gradle_testing

Si vous souhaitez intégrer vos projets Gradle à votre environnement de développement, vous pouvez consulter les ressources suivantes :

- Working with Gradle in IntelliJ IDEA - <https://www.youtube.com/watch?v=JwPYjnhah3g>
- Using the Gradle build system in the Eclipse IDE - <http://www.vogella.com/tutorials/EclipseGradle/article.html>
- Working with Maven in IntelliJ IDEA - <https://www.youtube.com/watch?v=pt3uB0sd5kY>

Quelques ressources supplémentaires :

- Gradle - <https://gradle.org/>
- Gradle Docs - The Java Plugin - https://docs.gradle.org/current/userguide/java_plugin.html
- Gradle Docs - The Application Plugin - https://docs.gradle.org/current/userguide/application_plugin.html
- Gradle Docs - Building Java Projects - https://docs.gradle.org/current/userguide/building_java_projects.html

TRAVAIL À RÉALISER

Connectez vous sur la plateforme gitlab de l'école (<https://gitlab.telecomnancy.univ-lorraine.fr>).

Rendez-vous sur le projet **Gerald.Oster/pcd2k19-gradle-bootstrap**

Créer un duplicata de ce projet (*fork*). Vous placerez ce duplicata dans l'espace de nommage relatif à votre compte.

Ajouter l'utilisateur dénommé **Gerald.Oster** comme membre de votre projet avec le rôle **Maintainer**.

Cloner en local votre nouveau dépôt. Celui-ci contient de quoi amorcer un projet gradle sans avoir à installer gradle :

```
$ git clone https://gitlab.telecomnancy.univ-lorraine.fr/<VOTRE_IDENTIFIANT/pcd2k19-gradle-bootstrap>.git
```

Dans la suite, pensez à faire des commits réguliers aux différentes étapes de votre projet.

Initialiser un nouveau projet gradle pour une application java :

```
$ ./gradlew init --type java-application
```

ou `./gradlew init --type java-library`

Vous pouvez supprimer les fichiers `src/main/java/App.java` et `src/test/java/AppTest.java` car vous ne les utiliserez pas.

Dans votre projet, ajouter les classes suivantes :

- `eu.telecomnancy.Student` (dans un fichier `Student.java` à placer au bon endroit dans votre arborescence). Une instance de la classe `Student` possède :
 - 2 attributs privés :
 - un nom `name` (une chaîne de caractères)
 - un numéro `id` (une valeur entière de type `int`)
 - 6 méthodes :
 - un *getter/setter* pour chacun des deux attributs (`getName`, `setName(String name)`, etc.)
 - une méthode `toString` permettant de générer une chaîne de caractères indiquant la valeur des attributs
 - une méthode `equals` permettant de comparer deux instances
- `eu.telecomnancy.StudentTest` (dans un fichier `StudentTest.java` à placer également au bon endroit) qui teste le comportement de la classe `Student` en utilisant la librairie `JUnit`.
- `eu.telecomnancy.App` qui possède une méthode `main` qui crée deux objets étudiants et affiche leurs noms et numéro d'étudiants.

Configurer gradle afin de pouvoir :

- compiler votre projet java
- exécuter votre projet java par l'intermédiaire de la commande `./gradlew run`
- exécuter les tests de votre application par la commande `./gradlew test`
- générer une archive exécutable (*jar*) par la commande gradle appropriée

Editer le fichier `README.md` afin de préciser le contenu de ce projet et d'indiquer vos nom, prénom et adresse email (`@telecomnancy.eu`).

Déposer l'ensemble de votre travail sur gitlab de l'école en *poussant vos commits* avant la date d'échéance communiquée (**Dimanche 3/11/2019 à 23:59**).

POUR ALLER PLUS LOIN

Vous avez maintenant toutes les clés pour automatiser la gestion des dépendances de vos projets logicielles et leur construction.

Vous pouvez ainsi automatiser la génération de vos analyseurs de votre projet PCL à partir du fichier de description de votre grammaire en utilisant le plugin gradle ANTLR (https://docs.gradle.org/current/userguide/antlr_plugin.html).

Tout ceci vous ouvre la voie vers l'intégration continue (<https://martinfowler.com/articles/continuousIntegration.html>) qui peut se mettre facilement en place en utilisant les pipelines GitLab (<https://docs.gitlab.com/ee/ci/pipelines.html>).

Informations concernant l'évaluation

Pour corriger, l'équipe pédagogique exécutera les commandes :

```
git clone ... // commande qui va bien
./gradlew run
./gradlew test
./gradlew jar
```

Toutes les commandes doivent s'exécuter sans erreur. Il doit être possible d'exécuter le `jar` produit en utilisant la commande `java -jar LE_NOM_DE_VOTRE_ARCHIVE.jar`

Nous tiendrons compte également du contenu de votre dépôt (uniquement les fichiers nécessaires doivent avoir été commités et ils devront être correctement organisés).