TP2 - ajout

Gradle

Gradle est un outil de build un peu plus évolué, conçu pour Java. En plus de s'occuper des tâches habituelles de compilations (*comme Ant, Rake, etc*), Gradle permet de gérer les dépendances d'un projet.

En Java, l'utilisation de dépendances (également nommées *librairies*) est une chose courante. Ces dernières permettent d'étendre les fonctionnalités du langage (par exemple, la librairie *StringUtils* de la Apache Foundation ajoute de nombreuses fonctionnalités sur la gestion des chaînes de caractères), ou de ne pas avoir à réécrire du code à de multiples reprises. C'est ce qui permet à Java de disposer de *Frameworks*, c'est à dire un ensemble de classes et méthodes pré-construites permettant de grandement simplifier la construction de projets.

Sans outil comme Gradle, il vous faudrait gérer ces dépendances à la main : téléchargement, installation, maintenance (montée de version, etc). De plus, vous seriez obligé d'envoyer sur votre serveur Git l'ensemble des dépendances de votre projet, ce qui est loin d'être optimal.

Fonctionnement de base

Gradle est disponible en binaire sans-installation, ou en package/logiciel installable sur la majorité des distributions Linux, Windows et Mac. Il était conçu à la base pour construire des projets Java, mais s'est diversifié : C++, Python, [...].

La configuration d'un projet gradle s'effectue dans un fichier `build.gradle`, situé à la racine de votre projet :

```
apply plugin: 'java'
apply plugin: 'application'

mainClassName='com.uca.Start'
sourceSets.main.java.srcDirs = ['src']
sourceSets.test.java.srcDirs = ['src']

repositories {
    mavenCentral()
}

dependencies {
    implementation 'org.hamcrest:hamcrest:2.2'
    implementation platform('org.junit:junit-bom:5.7.0')
    implementation 'org.junit.jupiter:junit-jupiter'
}

test {
    useJUnitPlatform()
}

jar {
 manifest {
```

```
        attributes 'Implementation-Title': 'Gradle Quickstart',
                   'Implementation-Version': '1.0.0',
                   'Main-Class': 'com.uca.Start'
    }
}
```

Décomposons point par point :

```
apply plugin: 'java'
apply plugin: 'application'
```

Dans cette section, nous indiquons à Gradle d'utiliser plusieurs paramètres par défaut : `java` et `application`. Ces "plugins" sont conçus par l'équipe de Gradle. Les détails sont disponibles sur https://docs.gradle.org/current/userguide/java_plugin.html et https://docs.gradle.org/current/userguide/application_plugin.html.

```
mainClassName='com.uca.Start'
sourceSets.main.java.srcDirs = ['src']
sourceSets.test.java.srcDirs = ['src']
```

Ici, nous indiquons à Gradle la localisation de notre classe principale (celle contenant notre méthode statique main), et la localisation des fichiers sources et de tests.

```
repositories {
    mavenCentral()
}
```

On indique à Gradle la liste des dépôts dans lesquels aller chercher nos dépendances. Ici, nous utilisons le dépôt MavenCentral, qui fait office de référence dans le domaine.

```
dependencies {
    implementation 'org.hamcrest:hamcrest:2.2'
    implementation platform('org.junit:junit-bom:5.7.0')
    implementation 'org.junit.jupiter:junit-jupiter'
}
```

Dans cette section, on indique à Gradle l'ensemble des dépendances à récupérer. Les noms et versions sont à chercher sur MavenCentral. Exemple avec Hamcrest : https://mvnrepository.com/artifact/org.hamcrest/hamcrest

```
test {
    useJUnitPlatform()
}

jar {
 manifest {
        attributes 'Implementation-Title': 'Gradle Quickstart',
                   'Implementation-Version': '1.0.0',
                   'Main-Class': 'com.uca.Start'
    }
}
```

Enfin, nous définissons ici certaines informations en plus des plugins. Par exemple, nous disons à Gradle d'utiliser JUnit pour les tests, et nous définissons certaines informations par défaut du Manifeste d'application Java (notamment la classe principale).

Exécutation

Avec le gradle comme dans le TP (wrapper):

- Démarrer le projet

```
./gradlew run
```

- Construire le projet :

```
./gradlew build
```

- Nettoyer le projet :

```
./gradlew clean
```

- Lancer les tests unitaires :

```
./gradlew test
```

Attention, avec les PC de la FAC, le `./gradlew` ne fonctionne pas toujours ! Deux solutions alternatives :

- Essayez d'utiliser `gradle <run/build/clean/test>`
- Si cela ne marche pas, utilisez `gradle --gradle-user-home <run/build/clean/test>`

Vous allez pouvoir essayer Gradle en conditions réelles, dès le TP3 😃