Mise en œuvre de Java

Emmanuel BRUNO
Université de Toulon, LIS UMR CNRS 7020
emmanuel.bruno@univ-tln.fr

2026-10-31

Table des matières

1 Installer 1
2 Compiler et exécuter 2
3 Documenter 4
4 Empaqueter 5

Ces instructions sont destinées aux systèmes Unix ou similaires (comme Linux et MacOS). Pour les utilisateurs de Windows, il est recommandé d'installer WSL (Windows Subsystem for Linux).

1 Installer

Pour installer et utiliser Java pour le développement d'applications, il suffit de télécharger un JDK. Il est recommandé d'utiliser celui du projet Temurin en raison de ses avantages en termes de licence. Une fois téléchargé, décompressez-le dans un répertoire de votre choix. Ensuite, définissez la variable d'environnement `JAVA_HOME` pour pointer vers ce répertoire et ajoutez `$JAVA_HOME/bin` à votre variable `PATH` pour pouvoir utiliser Java à partir de n'importe où dans votre système.

```
export JAVA_HOME=/usr/local/jdk....
export PATH="$JAVA_HOME/bin:$PATH"
```

Pensez à ajouter ces commandes au `.bashrc` (ou `.zshrc`).

Pour une installation rapide et simple de Java et de Maven, il est conseillé d'utiliser l'outil SDKMan. SDKMan permet de gérer facilement différentes installations et versions de Java et d'autres outils, facilitant ainsi la transition d'une version à une autre.

```
curl -s "https://get.sdkman.io" | bash
sdk install java             # Latest LTS (21)
```

```
sdk install java 23-tem      # install a specific Java version
sdk install maven            # Install latest Maven
```

on peut lister les JDK installés :

```
sdk list java|grep installed
```

On peut vérifier que les chemins sont corrects :

```
echo "JAVA_HOME = $JAVA_HOME"
echo "PATH = $PATH"
```

Et que les versions de la machine virtuelle

```
java --version
```

et du compilateur sont correctes :

```
javac --version
```

> A Faire
>
> Installer un JDK au minimum version 21.

2 Compiler et exécuter

Pour commencer, créez un répertoire qui contiendra tous vos projets, par exemple `/home/jovyan/work` (vous pouvez adapter ce chemin selon vos préférences). Ce répertoire sera appelé répertoire de travail ou workspace.

À l'intérieur de ce répertoire de travail, ajoutez le répertoire du projet, par exemple `java/TP1/premierprogramme/`, que nous appellerons répertoire de projet.

Ensuite, dans votre répertoire de projet, ajoutez les répertoires standards suivants : `src`, `bin`, `doc`, `dist` et `lib`.

Dans le répertoire `src`, créez l'arborescence correspondant au paquetage : `fr.univtln.login.tp.tp1` (remplacez `login` par VOTRE login). À l'intérieur de ce paquetage, ajoutez la classe Java `HelloWorld.java` ci-dessous.

```
package fr.univtln.jovyan.tp.tp1;

/**
```

```
 * A very simple class to say Hello
 * @author Emmanuel Bruno
 *
 */
public class HelloWorld {

    /**
     * A dummy constructor
     */
    public HelloWorld() { }

    /**
     * The entry point of the class
     * @param args The first parameter is a firstname
     */
    public static void main(String[] args) {
        if (args.length==1)
            System.out.println("Bonjour "+args[0]);
        else
            System.out.println("Usage : PremierProgramme [prenom]");
    }

}
```

créer les répertoires standards (adapter le chemin `/home/jovyan/work/`):

```
mkdir -p /home/jovyan/work/java/TP1/premierprogramme/{src,bin,doc,dist,lib}
```

créer l'arborescence des packages:

```
cd /home/jovyan/work/java/TP1/premierprogramme/src/
mkdir -p fr/univtln/jovyan/tp/tp1
```

Créer le fichier de la classe dans le répertoire de son package.

Le préfixe unique du package doit être adapté pour vous.

cela donne l'arborescence suivante :

```
tree "/home/jovyan/work/java/TP1/premierprogramme/"
```

A partir de votre répertoire de projet, compiler le programme avec la commande suivante qui indique la fichier à compiler, l'emplacement des sources au cas ou d'autres classes sont nécessaires et la destination du bytecode :

A Faire

Mettre en place manuellement le projet et modifier ce programme pour que l'instruction package soit correcte en adaptant votre login.

Pour compiler, il faut utiliser commande `javac` en indiquant le répertoire des sources (si d'autres classes doivent être compilées), éventuellement le répertoire de destination et obligatoirement le fichier source.

```
cd "/home/jovyan/work/java/TP1/premierprogramme/"
javac -sourcepath src -d bin src/fr/univtln/jovyan/tp/tp1/HelloWorld.java

tree "/home/jovyan/work/java/TP1/premierprogramme/"
```

Pour exécuter le programme, depuis votre répertoire de projet utiliser la commande `java` et ajouter le répertoire `bin` au classpath :

```
cd "/home/jovyan/work/java/TP1/premierprogramme/"
# Displays current directory
pwd
# Load the Helloworld class and run the main method.
java -classpath bin fr.univtln.jovyan.tp.tp1.HelloWorld Pierre
```

A Faire

Compiler et exécuter la classe `HelloWorld`.

3 Documenter

La commande `javadoc` génère de la documentation à partir de commentaires spécifiques insérés dans le code source des classes (encadrés par `/** ... */`). Cette fonctionnalité permet de documenter des packages, des classes ou interfaces, des variables d'instance, des méthodes, et plus encore.

Les commentaires peuvent contenir du texte simple ainsi que des balises HTML pour la mise en forme (`<I>` pour l'italique, `<B>` pour le gras, etc.). La balise `<code>` est utilisée pour inclure des extraits de code dans les commentaires. Des annotations spéciales, débutant par le caractère `@` (comme `@author`, `@version`, `@param`, etc.), permettent de définir des informations standard.

Il est impératif que les commentaires soient placés juste avant l'élément qu'ils documentent. Pour plus de détails, consultez la documentation officielle sur les commentaires Javadoc.

A partir de maintenant, tous vos programmes seront commentés.

```
cd "/home/jovyan/work/java/TP1/premierprogramme/"

javadoc -d doc -sourcepath src -subpackages fr.univtln.jovyan.tp.tp1
```

La documentation peut alors être consultée à partir du répertoire `doc` ici /home/jovyan/work/java/TP1/premierprogramme/doc/index.html.

un package Java est commenté en créant un fichier `package-info.java` dans le répertoire correspondant :

```
/**
 * This package is a first dummy exemple.
 * <p>
 * It presents a minimal Java Exemple.
 * </p>
 *
 * @since 1.0
 * @author Emmanuel BRUNO
 * @version 1.1
 */
package fr.univtln.jovyan.tp.tp1;
```

> A Faire
>
> Générer et consulter la documentation depuis votre répertoire de projet. Ajouter un commentaire pour le package `tp1`.

4 Empaqueter

Pour regrouper un ensemble de classes Java, on utilise la commande `jar`, qui permet de créer un fichier JAR (Java ARchive). Ce fichier contient les packages et les classes nécessaires à l'application.

```
cd "/home/jovyan/work/java/TP1/premierprogramme/"
cd bin
jar cvf ../dist/TP01EX01.jar fr
cd ..
```

Exécuter le programme en ajoutant le jar au classpath et en indiquant la classe exécutable avec la commande suivante :

```
cd "/home/jovyan/work/java/TP1/premierprogramme/"
java -cp dist/TP01EX01.jar fr.univtln.jovyan.tp.tp1.HelloWorld Max
```

> **A Faire**
>
> Générer l'archive de votre projet et l'exécuter.

Le fichier `.jar` contient des metadonnées dans le fichier `META-INF/MANIFEST.MF`:

```
cd "/home/jovyan/work/java/TP1/premierprogramme/"
unzip -q -c dist/TP01EX01.jar META-INF/MANIFEST.MF
```

Pour rendre un JAR exécutable en Java et spécifier la classe principale, vous pouvez suivre ces étapes :

1. Créez un fichier `monManifest.txt` à la racine de votre projet.
2. Ajoutez la ligne suivante dans ce fichier, en remplaçant `<nom de la classe point d'entrée>` par le nom complet de votre classe contenant le point d'entrée principal (c'est-à-dire la méthode `main` de votre application) :

   ```
   Main-Class: <nom de la classe point d'entrée>
   ```

   Par exemple, si votre classe principale s'appelle `MainApp`, le fichier `monManifest.txt` ressemblera à ceci :

   ```
   Main-Class: MainApp
   ```

3. Ensuite, pour créer votre JAR exécutable avec ce manifest personnalisé, utilisez la commande `jar` avec l'option `c` pour créer l'archive et l'option `m` pour spécifier le manifest :

```
jar cfm monApplication.jar monManifest.txt -C chemin/vers/les/classes .
```

- `monApplication.jar` est le nom que vous souhaitez donner à votre JAR exécutable.
- `monManifest.txt` est le nom du fichier manifest que vous avez créé.
- Assurez-vous de remplacer `chemin/vers/les/classes` par le chemin relatif ou absolu vers le répertoire contenant vos fichiers `.class`.

Pour plus de détails et d'exemples sur la création et l'utilisation de fichiers manifest dans les JAR Java, vous pouvez consulter la documentation Oracle sur les fichiers manifest.

Le manifest a insérer :

```
cd "/home/jovyan/work/java/TP1/premierprogramme/"
cat monManifest.txt
```

Créer le jar:

```
cd "/home/jovyan/work/java/TP1/premierprogramme/"
cd bin
jar cvfm ../dist/TP01EX01.jar ../monManifest.txt fr
cd ..
```

Puis exécuter directement l'archive :

```
cd "/home/jovyan/work/java/TP1/premierprogramme/"
java -jar dist/TP01EX01.jar Pierre
```

A Faire

Rendre l'archive de votre projet « exécutable » et l'exécuter.