Structure d'un projet et paquets

Alexis Nasr (d'après les slides de Arnaud Labourel)

Structure d'un projet

En Java, un projet peut être découpé en paquets (`package`).

Les paquets permettent :

- de regrouper des classes afin de mieux organiser le code
- de créer des modules indépendants réutilisables
- d'avoir plusieurs classes qui possèdent le même nom (du moment qu'elles ne sont pas dans le même paquet)

Un paquet (`package`) :

- est une collection de classes
- peut contenir des sous-paquets

Lors de l'exécution...

Java utilise l'arborescence de fichier pour retrouver les fichiers `.class`

- Une classe (ou une interface) correspond à un fichier `.class`
- Un dossier correspond à un paquet

Les `.class` du paquet `com.univ_amu` doivent :

- être dans le sous-dossier com/univ_amu
- le dossier com doit être à la racine d'un des dossiers du `ClassPath`

Le `ClassPath` inclut :

- le répertoire courant
- les dossiers de la variable d'environnement CLASSPATH
- des fichiers JAR
- des dossiers précisés sur le ligne de commande de java (`-classpath path`)

Lors de la compilation... (1/2)

Le mot-clé `package` permet de préciser le paquet des classes ou interfaces définies dans le fichier :

```
package com.univ_amu;
public class MyClass { /* ... */ }
```

Java utilise l'arborescence pour retrouver le code des classes ou interfaces :

- Une classe (ou une interface) `MyClass` est cherchée dans le fichier `MyClass.java`
- Le fichier `MyClass.java` est cherché dans le dossier associé au paquet de `MyClass`

Lors de la compilation... (2/2)

```java
package com.univ_amu;
public class MyClass { /* ... */ }
```

Dans l'exemple précédent, il est donc conseillé que le fichier :

- se nomme `MyClass.java`
- se trouve dans le dossier `com/univ_amu` (Par défaut, la compilation crée `MyClass.class` dans `com/univ_amu`)

Utilisation d'une classe à partir d'un autre paquet

Accessibilité :

- Seules les classes publiques sont utilisable à partir d'un autre paquet
- Un fichier ne peut contenir qu'une seule classe publique

On peut désigner une classe qui se trouve dans un autre paquet :

```
package com.my_project;
    public class Main {
    public static void main(String[] args) {
        com.univ_amu.MyClass myClass =
            new com.univ_amu.MyClass();
    }
}
```

Importer une classe

Vous pouvez également importer une classe :

```
package com.my_project;
import com.univ_amu.MyClass;

public class Main {
    public static void main(String[] args) {
        MyClass myClass = new MyClass();
    }
}
```

Deux classes de deux paquets différents peuvent avoir le même nom :

- Exemple : `java.util.List` et `java.awt.List`
- Attention de choisir le bon import

Importer un paquet

Vous pouvez également importer toutes les classes d'un paquet :

```
package com.my_project;
import com.univ_amu.*;

public class Main {
    public static void main(String[] args) {
        MyClass myClass = new MyClass();
    }
}
```

Remarques :

- Les classes des sous-paquets ne sont pas importées
- Le paquet `java.lang` est importé par défaut

Importer des membres statiques

Depuis Java 5, il est possible d'importer directement des méthodes ou attributs de classes (`static`).

La syntaxe est la suivante :

```
import static my_package.my_class.my_static_member;
```

Exemple :

```
import static java.lang.Math.PI;
import static java.lang.Math.pow;
```

Exemple sans import statique

```
public class HelloWorld {
  public static void main(String[] args) {
    System.out.println("Hello World!");
    System.out.println(
      "A circle with a diameter of 5 cm has");
    System.out.println("a circumference of "
                       + (Math.PI * 5) + " cm");
    System.out.println("and an area of "
            + (Math.PI * Math.pow(2.5, 2))
            + " sq. cm");
  }
}
```

Exemple d'import statique

```
import static java.lang.Math.PI;
import static java.lang.Math.pow;
import static java.lang.System.out;

public class HelloWorld {
  public static void main(String[] args) {
          out.println("Hello World!");
          out.println(
          "A circle with a diameter of 5 cm has");
          out.println("a circumference of "
                + (PI * 5) + " cm");
          out.println("and an area of "
            + (PI * pow(2.5, 2)) + " sq. cm");
    }
}
```

Un exemple

Le fichier com/univ_amu/HelloWorld.java :

```
package com.univ_amu;
    public class HelloWorld {
    public static void main(String[] arg) {
        System.out.println("Hello world ! ");
    }
}
```

```
$ javac com/univ_amu/HelloWorld.java
$ ls com/univ_amu
HelloWorld.java HelloWorld.class
$ java com.univ_amu.HelloWorld
Hello world !
```

Quelques remarques

Nommage des paquets :

- Les noms de paquets sont écrits en minuscules
- Pour éviter les collisions, on utilise le nom du domaine à l'envers ⇒ `com.google.gson`, `com.oracle.jdbc`
- Si le nom n'est pas valide, on utilise des underscores : ⇒ `com.univ_amu`

Fichier JAR (Java Archive) :

- est une archive ZIP pour distribuer un ensemble de classes Java
- contient un *manifest* (qui peut préciser la classe qui contient le main)
- peut également faire partie du `ClassPath`
- peut être généré en ligne de commande (`jar`) ou avec un IDE