Page de Jean-Michel Richer

Maître de Conférences en Informatique

<<<M1 - UE2

0 Rappels Java

Note : ce chapitre a pour objet de rappeler les notions essentielles pour la compilation et l'exécution des programmes Java en dehors d'Eclipse.

Eclipse est un outil formidable, mais il masque au programmeur la complexité relative de la compilation et l'exécution avec Java. Nous tentons donc de montrer dans ce chapitre comment procéder.

1. Organisation des sources

On crée un répertoire pour notre projet, par exemple `projet` :

```
mkdir projet
cd projet
```

On peut reprendre l'organisation d'Eclipse, qui est celle liée aux projets Java de manière générale. On crée deux répertoires principaux :

- `src` : le répertoire dans lequel on place les sources
- `bin` (sous Eclipse) ou `build` : le répertoire dans lequel on place les fichiers .class

Eventuellement, si on utilise des librairies externes, on crée un répertoire `lib` dans lequel on placera les **.jar**

En Java, afin de séparer et identifier les classes, on crée des **packages** :

```
package com.public.server;

public class ServerHttp {
...
}
```

Au niveau de l'arborescence cela implique que le fichier **`ServerHttp.java`** sera placé dans le répertoire : **`src/com/public/server`**.

Voici un exemple d'arborescence :

```
rmi_example/
├── build
│   ├── client
│   │   ├── ComputeApplet$1.class
│   │   ├── ComputeApplet.class
│   │   └── ComputeClient.class
│   ├── common
│   │   ├── ComputeInterface.class
│   │   ├── InputData.class
│   │   └── Result.class
│   └── server
│       ├── ComputeImplementation.class
│       ├── ComputeServer.class
│       └── ComputeServerRegistryLaunch.class
├── build.xml
├── README
├── server.policy
└── src
    ├── client
    │   ├── compute_applet.html
    │   ├── ComputeApplet.java
    │   └── ComputeClient.java
    ├── common
    │   ├── ComputeInterface.java
```

```
    │   ├── InputData.java
    │   └── Result.java
    └── server
        ├── ComputeImplementation.java
        ├── ComputeServer.java
        └── ComputeServerRegistryLaunch.java
```

2. Compilation des sources

On se place dans le répertoire du projet :

```
cd projet
```

2.1. avec un fichier

```
javac -d build -cp src src/com/public/server/ServerHttp.java
```

Les options sont les suivantes :

- `-d [directory]` : indique le répertoire destination où on placera les fichiers **.class**
- `-cp src` : indique le **classpath**, c'est à dire l'ensemble des répertoires (et fichiers jar) où on trouve le code source

2.2. avec plusieurs fichiers

Si il existe une hiérarchie de répertoires alors, on peut utiliser :

```
files=`find src -name "*.java" -print | tr '\n' ' '` && javac -d build -cp src $files
```

2.3. librairies externes

Si on fait appel à une librairie externe (.jar) il faudra la spécifier dans le classpath, par exemple :

```
javac -d build -cp src:lib/logging.jar src/com/public/server/ServerHttp.java
```

3. Execution des fichiers objects : .class

On utilise le même principe que pour la compilation concernant le classpath, par contre on fait référence aux classes par leurs noms et le package dans lequel elles se trouvent :

```
java -cp build com.public.server.ServerHttp
```

Ici, on tente d'exécuter :

- le fichier `ServerHttp.class`
  - attention : on ne précise pas l'extension **.class**
  - ce fichier doit contenir une méthode `public static void main(String args[])`
- qui appartient au package `com.public.server`
- à partir du répertoire `build`

4. Automatisation de la compilation et l'exécution

On peut utiliser **make** pour automatiser la compilation et l'exécution, ou alors **ant** qui est un make au format XML dédié initialement à Java

Voici un exemple (build.xml) de fichier de compilation lié à **ant** :

```
1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <project name="ant_example" default="compile" basedir="." >
3.
4.    <!-- ================================= -->
5.    <!-- ========== properties ========== -->
6.    <!-- ================================= -->
7.
8.    <property name="project.version" value="1.0" />
9.    <property name="jar.file" value="${ant.project.name}-
    ${project.version}.jar" />
10.
```

```
11.   <property name="src.dir" value="src" />
12.   <property name="build.dir" value="build" />
13.   <property name="build.src.dir" value="${build.dir}/${src.dir}" />
14.
15.   <property name="lib.dir" value="lib" />
16.   <property name="doc.dir" value="doc" />
17.
18.   <property name="tests.dir" value="tests" />
19.   <property name="build.tests.dir" value="${build.dir}/${tests.dir}" />
20.   <property name="tests.report.dir" value="report" />
21.   <property name="tests.report.html.dir" value="report" />
22.
23.
24.
25.   <!-- ================================ -->
26.   <!-- ========== paths ========== -->
27.   <!-- ================================ -->
28.   <path id="src.compile.classpath">
29.    <fileset dir="${lib.dir}" >
30.     <filename name="log4j-1.2.17.jar" />
31.    </fileset>
32.   </path>
33.
34.   <path id="tests.compile.classpath">
35.    <fileset dir="${lib.dir}" />
36.    <pathelement path="${build.src.dir}" />
37.   </path>
38.
39.
40.   <path id="run.classpath">
41.    <fileset dir="${lib.dir}" >
```

```
42.     <filename name="log4j-1.2.17.jar" />
43.    </fileset>
44.    <pathelement path="${build.src.dir}" />
45.   </path>
46.
47.   <!-- ================================= -->
48.   <!-- ========== targets ========== -->
49.   <!-- ================================= -->
50.
51.   <!-- clean -->
52.   <target name="clean" description="clean project">
53.    <delete dir="${build.dir}" />
54.   </target>
55.
56.   <!-- initialize -->
57.   <target name="init" description="create directories">
58.    <mkdir dir="${build.src.dir}" />
59.    <mkdir dir="${build.tests.dir}" />
60.   </target>
61.
62.   <!-- compile source -->
63.   <target name="compile" depends="init" description="source
    compilation">
64.    <javac srcdir="${src.dir}" destdir="${build.src.dir}" fork="true"
65.     classpathref="src.compile.classpath" includeantruntime="false" >
66.
67.    </javac>
68.   </target>
69.
70.   <!-- compile tests -->
71.   <target name="compile_tests" depends="compile" description="tests
    compilation">
```

```
72.     <javac srcdir="${tests.dir}" destdir="${build.tests.dir}" fork="true"
73.      classpathref="tests.compile.classpath" includeantruntime="false" >
74.     </javac>
75.   </target>
76.
77.   <!-- run simple application -->
78.   <target name="run_app">
79.     <delete>
80.       <fileset dir="." includes="*.log"/>
81.      </delete>
82.     <copy file="${src.dir}/log4j.properties" todir="${build.src.dir}" />
83.     <java classname="com.ant_example.code.Application" fork="true"
84.      classpathref="run.classpath" >
85.     </java>
86.   </target>
87.
88.   <!-- run tests -->
89.   <target name="tests" depends="compile">
90.     <mkdir dir="${tests.report.dir}" />
91.     <copy file="${src.dir}/log4j.properties" todir="${build.tests.dir}" />
92.     <junit printsummary="true" haltonfailure="true">
93.           <classpath refid="run.classpath" />
94.           <classpath>
95.             <pathelement location="${build.dir}" />
96.           </classpath>
97.           <formatter type="plain" />
98.      <formatter type="xml" />
99.           <batchtest todir="${tests.report.dir}">
100.            <fileset dir="${build.tests.dir}"
    includes="**/*TestCase.class" />
101.          </batchtest>
```

```
102.       </junit>
103.   </target>
104.
105.   <target name="tests_report" depends="tests">
106.     <mkdir dir="${tests.html.dir}" />
107.     <junitreport todir="${tests.dir}">
108.       <fileset dir="${tests.dir}">
109.         <include name="TEST-*.xml" />
110.       </fileset>
111.       <report format="noframes" todir="${tests.html.dir}" />
112.     </junitreport>
113.   </target>
114.
115.   <target name="javadoc" depends="compile" description="Generate
documentation">
116.     <mkdir dir="${doc.dir}" />
117.     <javadoc sourcepath="${src.dir}" destdir="${doc.dir}"
118.       author="true" version="true"
119.       windowtitle="${ant.project.name} API">
120.       <classpath refid="compile.classpath" />
121.
<doctitle><![CDATA[<h1>${ant.project.name}</h1>]]></doctitle>
122.       <bottom><![CDATA[Copyright &#169; 2012 ALl Rights
Reserved]]></bottom>
123.     </javadoc>
124.
125.   </target>
126.
127.   <target name="jar" depends="compile" description="generate jar
file">
128.     <jar destfile="${jar.file}" basedir="${build.classes.dir}" />
129.   </target>
```

```
130. </project>
```