Object Oriented Programming

Java: installation, configuration and tools

Tools – revision

- **J2SE JDK (last version)**
  http://www.oracle.com/technetwork/java/javase/downloads/index.html
- **Eclipse, for java developers**
  http://www.eclipse.org/downloads/
  - **NetBeans, Java SE** (Eclipse alternative)
    http://www.netbeans.org/downloads/index.html
- **Visual paradigm**
  https://delta.ist.utl.pt/software/visual_paradigm.php

Java

- Download **JDK** from:
  http://www.oracle.com/technetwork/java/javase/downloads/index.html
- Last version
  - To obtain version, execute `java -version`

Java Platform – revision (1)

- Java technology is distributed for 3 plataforms:
  - J2EE (*Entreprise Edition*), to develop enterprise applications.
  - J2ME (*Micro Edition*), to embedded devices (mobiles and PDA's).
  - **J2SE (*Standard Edition*), to desktops and servers**.

Java Platform – revision (2)

- Inside Java platform there is:
  - **J2xx Runtime Environment (JRE)**:
    - JVM interpreter, environment classes, …
    - Used only to run applications.
  - **J2xx Development Kit (JDK)**:
    - JRE, compiler, utility classes (Swing,…), …
    - Used for application development.

Java Platform – revision (3)

- Java 2 API consists in different classes organized within packages and sub-packages.
- Basic packages:
  - **java.lang**: environment classes (automatically imported)
  - **java.util**: utility classes (data types, etc)
  - **java.io**: I/O classes
  - java.net: classes for networking applications (TCP/IP)
  - java.sql: classes for database connection via JDBC
  - java.awt: native graphical user interface
  - **javax.swing**: graphical user interface (lighter than java.awt)
- Documentation:
  - HTML: https://docs.oracle.com/javase/8/docs/api/
  - Sources: http://www.docjar.com/
    http://grepcode.com/project/repository.grepcode.com/java/root/jdk/openjdk/

Java – Linux (1)

- Usually, **in Linux Java is located at**:
  - **/usr/java/jdk1.5.0_06**
    - **Compiler (javac) and JVM Interpreter (java)**
      - **/usr/java/jdk1.5.0_6/bin/**
    - **JVM Interpreter (java)**
      - **/usr/java/jre1.5.0_06/bin/**
- Pre-defined Java classes are organized in the system file following the name of the packages where they are found:
  - **/usr/java/jdk1.5.0_06/src.zip**
    For instance, class `String` is defined in package `java.lang` therefore it is found in the zip file `src.zip` inside `java/lang/String.java`

Java – Linux (2)

- Configuration:
  - Update **PATH variable** if you want to run JDK executables are needed (`javac`, `java`, `javadoc`, etc) from the working directory without having to use the complete path to the executables.
  - Similarly, update **CLASSPATH variable** if the directory to other classes are needed (for instance, to a library) and you do not want to use the complete path to them.
  - Are you using the updated PATH?

```
% which java
% java: Command not found
```

Java – Linux (3)

- For C shell (csh):
  - Add to the startup file (`~/.cshrc`) the directory of the compiler and the JVM interpreter:

```
setenv Ljava /usr/java/jdk1.5.0_06
set path=($path $Ljava/bin)
```

    - Load the startup file and check java path:

```
source ~/.cshrc
which java
```

  - Directory to other classes (for instance, environment classes) are indicated in the environment variable CLASSPATH

```
setenv CLASSPATH .:$Ljava
```

Java – Linux (4)

– For ksh, bash or sh:
  - Add to the startup file (`~/.profile`) the directory of the compiler and the JVM interpreter:
    ```
    PATH=/usr/java/jdk1.5.0_06/bin:$PATH
    ```
    – Load the startup file and check java path:
    ```
    . $HOME/.profile
    which java
    ```
  - Directory to other classes (for instance, environment classes) are indicated in the environment variable CLASSPATH
    ```
    CLASSPATH=/usr/java/jdk1.5.0_06:$CLASSPATH
    ```

Java – Windows (1)

- Usually, **in Windows Java is located at**:
  - **C:\Program Files\Java**
    - **Compilader (javac) and JVM interpreter (java)**
      - **C:\Program Files\Java\jdk1.6.0\bin**
    - **Interpreter JVM (java)**
      - **C:\Program Files\Java\jre1.6.0\bin**
- Pre-defined Java classes are organized in the system file following the name of the packages where they are found:
  - **C:\Program Files\Java\jdk1.6.0\src.zip**
    For instance, class `String` is defined in package `java.lang` therefore it is found in the zip file `src.zip` inside `java/lang/String.java`

Java – Windows (2)

- Configuration:
  - Update **PATH variable** if you want to run JDK executables are needed (`javac`, `java`, `javadoc`, etc) from the working directory without having to use the complete path to the executables.
  - Similarly, update **CLASSPATH variable** if the directory to other classes are needed (for instance, to a library) and you do not want to use the complete path to them.

Java – Windows (3)

- Include in PATH variable the directory of Java executables (typically in `Control Panel` + `System` + `Advanced` + `Environment Variables` + `User variables` or `System variables`).
- Directory to other classes (for instance, environment classes) are indicated in the environment variable CLASSPATH (typically in `Control Panel` + `System` + `Advanced` + `Environment Variables` + `User variables`).

The `main` method – revision

- The JVM interpreter executes the **`main` method** of the class indicated in the command line:
  - Qualifiers: **public static**
  - Return: **void**
  - Parameters: **String[] args**
- All classes in the application may have a `main` method. The `main` method to execute is specified each time the program is run.

Executing Java programs (1)

- Steps to execute a Java program:

  1) Create a directory `dir` ← This correspond to the Java package!!

  2) In `dir` edit `File.java` ↙

  - The first line of this file must be `package dir;`
  - `File.java` contain class `File`.
  - All extra classes should be in the directories indicated in the CLASSPATH.

  3) In `dir` compile `javac File.java`

  Or in the parent directory of dir `javac dir/File.java`

  - The option `-cp` can be used to indicate needed directories that were not indicated in the CLASSPATH.
  - After compiling, a `File.class` is created.

  4) Go back to the parent directory of `dir`

Executing Java programs (2)

5) Execute `java dir.File` ← You need to provide the full path from the root package (and sub-packages) until the class that contains the main!!

- The class `File` should contain the `main` method.
- The option **`-cp`** can be used to indicate needed directories that were not indicated in the CLASSPATH.
  - In Windows, for instance:
    `java -cp %CLASSPATH%;C:\libs\lib.jar Fich`
  - In Linux, for instance:
    `java -cp \usr\libs\lib.jar:$CLASSPATH Fich`
- The option **`-verbose`** can be used, that list all steps and loaded classes.

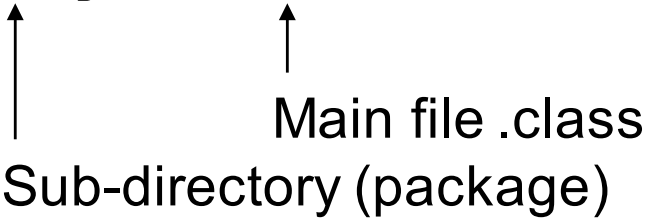
The jar tool (1)

- The **jar tool** (*Java archive tool*) manages archive files `.class`, preserving directory hierarchy.
  - The directory hierarchy should preserve the package hierarchy. For instance, the class `String` of package `java.lang`, is defined inside `/java/lang/String.java`

The jar tool (2)

- The JAR archive may contain the directory
  `META-INF/`
  where the following file can be found
  `MANIFEST.MF`
  with the information about the class to run:
  - Directives (example: version, tool)
  - Main class (class to run)
  - White line

```
Manifest-Version: 1.0
Created-By: 1.5.0_01 (Sun Microsystems Inc.)
Main-Class: project.Simulator
```



The jar tool (3)

Command	Objective
`jar cf archive.jar file-list`	Creates a jar archive with a default manifest
`jar cfm archive.jar manifest-file file-list`	Creates a jar archive with a given manifest
`jar tf archive.jar`	List archive contents
`jar xf archive.jar [file-list]`	Extract files

The jar tool (4)

- The JVM interpreter can also run a jar archive:
  `java -jar project.jar`
  The class having the main method to run is indicates in the file `META-INF/MANIFEST.MF`
- The jar program, is developed in C, and it is available in JDK.
- In Windows, the JAR archive can be opened by WinRAR.

The jar tool (5)

- **Executable**:
  - To make a jar archive file executable the **`MANIFEST.MF`** file **should contain a line corresponding to the `Main-Class`**.
  - To execute a jar archive use **option –jar**.
- **Libraries**:
  - To distribute a library one just need to make available a **jar archive with the compiled classes** (in that case the `MANIFEST.MF` file should not contain a line corresponding to the `Main-Class`).
  - To use a library one just need to compile/execute the program having the **library jar archive in the CLASSPATH**.