Software Engineering Course

Tutorial

Getting started with Java

2022-2023

Dr. Kouah S.

Creating a Java Program with Eclipse

- The **Eclipse environment** allows you to edit **classes Java** using text files with **.java extension.**
- Before editing a class, you must **define a project** .
- Once the **project name is specified** , the corresponding directory is created in the Workspace **directory .**
- Then, classes are created gradually and the programmer inserts **the necessary attributes and methods** .

Java

To make the programs easier to read, we define **some conventions** :

- the **class name** is **meaningful** and **always begins with a capital letter** , avoiding it being too long.
- the **name of attributes and methods begins with a lowercase letter** .
- A method is differentiated from an attribute using parentheses.
- **Special case** : constructors start with capital letters (class name).

Class, Constructor, Attribute, Method

- Example :
- **Hello** : is the name of the **class**
- **Hello()** : is the name of the **constructor**
- **word** : is the name of an **attribute**
- **print (word)** : is the name of a **method**

Example of JAVA program

```
public class Hello {
public static void main(String[ ] args )
/* the program starts from this line – the argument args is the array where the program variables are saved */
{
System.out.println (“Hello every one”);
// print on a console window
}
}
```

Reminder : **File names** are of the form class **name.java** (**Hello.java**)

Compilation

- Compiling **a project** allows you to generate **.class files** that correspond to the different classes of the project.
- These generated files do not correspond to the application **executable but to precompiled code used by the Java Virtual Machine (JVM)** to run the project.
- ***javac*** command lines to generate the precompiled code and *java* to launch the program with a ***run* command** (from the menu).
- **Environment variables** can be initialized initially, so when the program runs, they are used in the defined order.
- NB: if the modifications are not taken into account during execution, consider refreshing your project.

Syntax of a Java program with Eclipse

- Defining a class consists of **describing its variables (attributes) and its methods** while specifying their nature (modifier type, variable type, return type, etc.).
- The syntax is very similar to the syntax used in C/C++ with ease for programmers (we do not manipulate pointers).
- Class instances are manipulated only by their name.

Syntax of a Java Program

class Classe {

Type varibale1;

Type varibale2, variable3;

Type[] variable4;// variable4 est un tableau de type Type.

Modifier Type variable2;

Modifier Type method1() {

// code of method1

... }

Modifier void methode2() {

// code of method2 ... }

}

Modifiers

- The type of a variable or the return type of a method can be an elementary type (int , float , string, etc.) or an already defined class.
- Modifiers of classes, variables or methods in Java allow you to define authorized access and visibility towards other classes. There are several modifiers:
- **public** : A class, variable or method declared as public is visible to all other methods inside or outside the class. Generally, we avoid declaring variables as public and we create special methods for updating. This allows better control and safe code (principle of object encapsulation).
- **private** : This is the strongest level of visibility restriction. Only the methods of the class in which the private entity will be able to see it (variable or method).

Modifiers

- **protected** : this modifier defines access as follows: if in a class, we declare a method or a variable as protected , only methods present in the same package will be able to access it or derived classes (even in a different package) . You cannot use protected to qualify a class.
- **static** : until now, we have considered that each variable is specific to an object. This is why we talk about a class instance. We may need to define variables common to several instances of the same class that we qualify as static (pi=3.14 variable common to all Circle instances).
- **final** : a variable is qualified as final means that the variable is a constant. When this modifier is added to a class, prohibits creating a class that inherits it.

Exercise 01

- Consider the code of the following class:

```
public class Livre {
// Variables
private String title, Author;
private int nbOfPages // nombre de pages
// Constructor
public Livre(String unAuthor, String aTitle) {
Author = unAuthor;
title = aTitle;   }
// Getters
public String getAuthor() {      return A;      }
// Setters
void setNbOfPages(int n) {      nbPages = nb;      }
```

Correct the errors, complete the code, and add a main() method for:

- Create 2 Books,

- Display the authors of these 2 Books.

Exercise 02

1) Define a Pen class whose attributes are length and diameter with a print () display method that displays the values of the initialized attributes.

2) In the main program (define a Test class):

- Create an instance of a pen with length 50 and diameter 0.2.

- Modify your program so that you can enter the properties of a pen from program variables (generalize your program to define n pens)

3) We want to define three categories of pens: Pencil, Bic and Feather with the common properties length and diameter and a method that displays these properties and the category. Implement the corresponding class(es).

4) Write a display method which takes as input a pen type instance (Pencil, Bic or Feather) and displays the stated properties specifying the category.