University of Puerto Rico
Department of Electrical and Computer Engineering

ICOM 4015 Laboratory: Advanced Programing

Laboratory 1: Introduction to Eclipse, Editing and Executing programs in Java

Completed by:
ID:
Date:

1 Introduction

In this laboratory we will learn to work with Eclipse IDE. During the process, we will create workspaces to save all of our projects and we will create and edit programs in Java. Also, we will work with the process of compiling, error correction and execution of Java language programs.

2 Accessing the lab computers.

Through the laboratory experience, we will be using a Unix/Linux based operating system called Ubuntu. Maybe some of you are not familiar with it. But soon you'll learn it is a very friendly OS just like Windows. You can download and install the open source Ubuntu OS in your computer as an alternative OS to practice in similar environment to the laboratory.

To log in your computer you will be requested to input your already known user name and password. The home interface of Ubuntu should load shortly as any other desktop computer.

3 Preparing to create our programs.

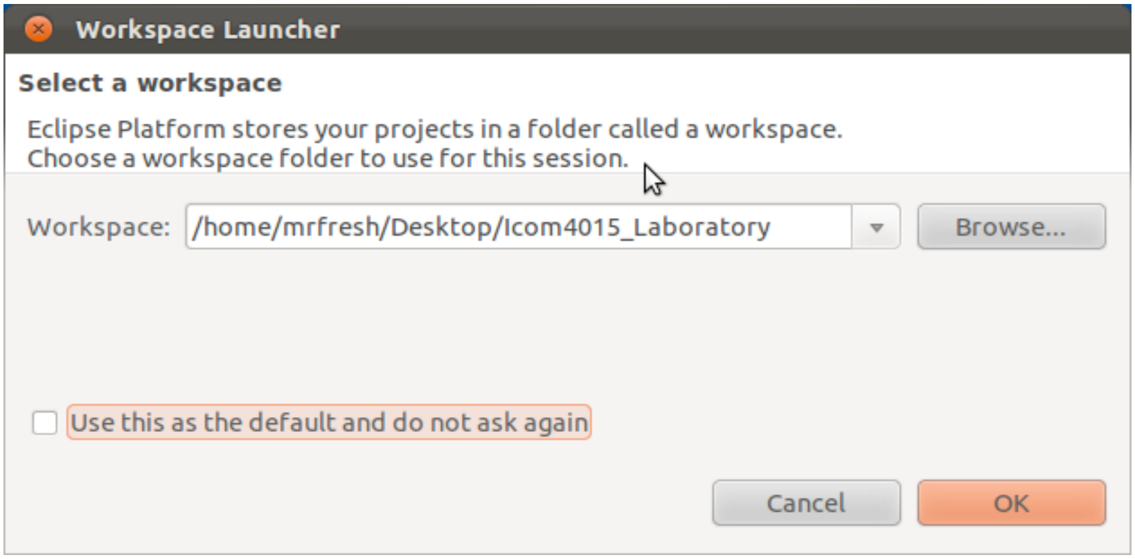
3.1 Creating workspaces to save our projects

When we work with the Eclipse IDE we have to indicate where to store our work. This is known as the *workspace.* Although, we can decide where to store our projects once the IDE has started, in this case we are going to decide where to store our work first and then indicate it to Eclipse. The laboratory instructor will give further instructions on where to create a folder to be our *workspace* to store the laboratories.

In the home screen of Ubuntu make a right click to create a new folder named Icom4015_Laboratory.

3.2 Entering Eclipse and creating a *workspace*

Locate Eclipse IDE program, launch it and click on Browse button to select the folder created in the previous step to be used as workspace. See figure 1 below.



Workspace Launcher

Select a workspace

Eclipse Platform stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: /home/mrfresh/Desktop/Icom4015_Laboratory

□ Use this as the default and do not ask again

Figure 1

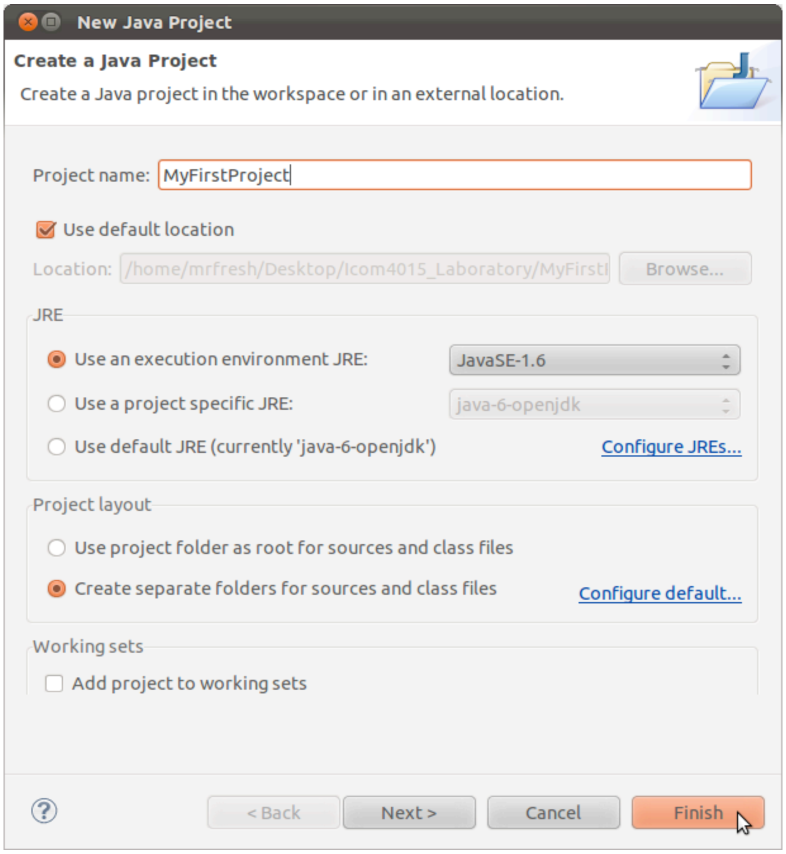
Afterwards, click the OK button to load Eclipse.

3.3 Preparing a project

When we create an application in Java, it can have multiple files associated with it. Those files, interrelated, are grouped in what is known as a project.

3.3.1 Creating a project

Once the IDE has loaded, in the upper left of the screen click on File followed by New, select Java Project. Give a name to your project and click on Finish. See figure 2.



Create a Java Project

Create a Java project in the workspace or in an external location.

Project name: MyFirstProject

☑ Use default location

Location: /home/mrfresh/Desktop/Icom4015_Laboratory/MyFirstI

JRE

● Use an execution environment JRE: JavaSE-1.6

○ Use a project specific JRE: java-6-openjdk

○ Use default JRE (currently 'java-6-openjdk')

Project layout

○ Use project folder as root for sources and class files

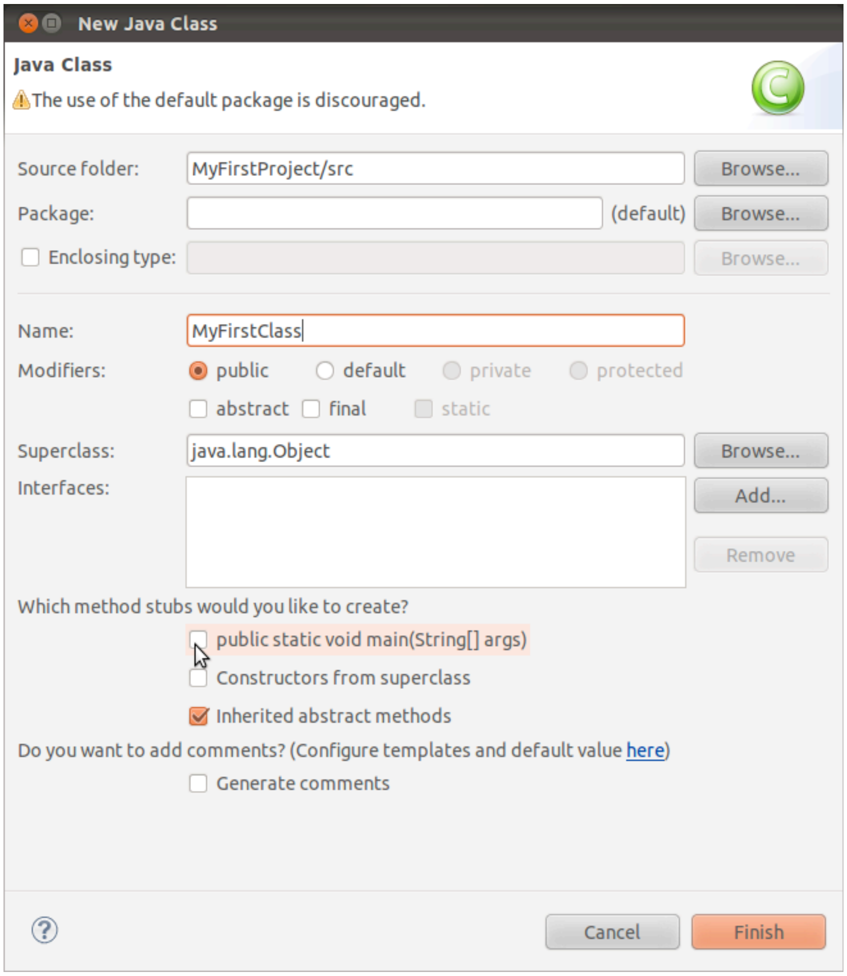
● Create separate folders for sources and class files

Working sets

□ Add project to working sets

Figure 2

The next step is to add a class to the project so we can start to write Java code. This is done by clicking on the Package Explorer window of the IDE and expanding the folder created in the previous step. There we should see a folder named *src.* With the cursor on top of that folder, make a right click and select New and select Class. This will open the New Class window. See Figure 3.



Java Class

The use of the default package is discouraged.

Source folder: MyFirstProject/src

Package: (default)

☐ Enclosing type:

Name: MyFirstClass

Modifiers: ● public ○ default ○ private ○ protected

☐ abstract ☐ final ☐ static

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

☐ public static void main(String[] args)

☐ Constructors from superclass

☑ Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

☐ Generate comments

Figure 3

In this window we give name to the Class we which to create and click on Finish. Another feature is that we can select whether or not the IDE creates a Class with a predefined *main* method. In this case we will not check the option to include a *main* as seen in Figure 3 where the cursor is pointing.

4 Working with programs in Java

4.1 Using the code editing section on the IDE, write the following program.

```
//My first program
public class HelloWorld
{
   public static void main(String args[])
   {
      System.out.println("The program works!");
   }
}
```

Compile the program, verifying that there is no syntax error. If there is any, fix the program. Execute the program and observe where the result appears.

4.2 Practice exercise

Write a program that generates the following output:

```
**************************
*     Student name       *
*          ID            *
**************************
```

Where the “Student name” is your name and the ID is your student number. For example,

```
*******************
*  Tito Trinidad  *
*   123-45-6789   *
*******************
```

The box will contain the name and the ID with a blank space in between. All centralized along with the borders.

Once the program is running correctly, copy and paste the code inside the frame of diagonals below:

```
/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

4.3 Program with arithmetic operations and keyboard input

Write the following program in Eclipse, compile and execute it.

```
//Example of keyboard input read
import java.util.Scanner;
public class KeyboardInputAndArithmetic
{
   public static void main(String args[])
   {
      Scanner keyboard = new Scanner(System.in);

      int value1,
          value2,
          sum;

      String wordForTheResult;

      System.out.print("Enter the first value    : ");
      value1 = keyboard.nextInt();
      System.out.print("Enter the second value    : ");
      value2 = keyboard.nextInt();
      System.out.print("Word that will precede the result  : ");
      wordForTheResult = keyboard.next();

      sum = value1 + value2;

      System.out.println(wordForTheResult + " = " + sum);

   }
}
```

Note where and how the operations of data input are performed.

What happens if we enter a phrase of various words instead of a word?

//

///

What happens if we enter decimal numbers instead of integer numbers?

//

///

What happens if we enter a string instead of numbers?

//

///

4.4 Practice exercise

Converting from months to years and months.

Write a program that prompts the user for his age in months. The program will calculate how many years and months the user has from the input and will display the result in the console. An example of the execution of the program is the following:

```
Enter your age in months: 297
Your age in years and months is: 24 years and 9 months
```

Once the program is running correctly, copy and paste the code inside the frame of diagonals below:

///

//