



UNIVERSITÀ DI PISA

Laboratorio di reti B

Introduction to Eclipse IDE

Prof. Alina Sîrbu (alina.sirbu@unipi.it)

Assistant: Andrea De Salve (desalve@unipi.it)



2016/2017



Overview

- Background
- Obtaining and installing Eclipse
- Creating a workspace / project
- Creating class
- Compiling and Running code
- Debugging

Background: Eclipse IDE

- What is Eclipse?

An Integrated Development Environment: A program that facilitates and supports the development of software. Created by OTI and IBM and written in java

- Providing tools for: Coding, building, debugging
- Language-neutral: Java, C, HTML, XML, PHP,..
- Open Source
- Rich community of Eclipse plug-in development



Obtaining and Installing Eclipse

- Download and Install Eclipse IDE for you platform:

<https://eclipse.org/ide/>

- Requirements: either

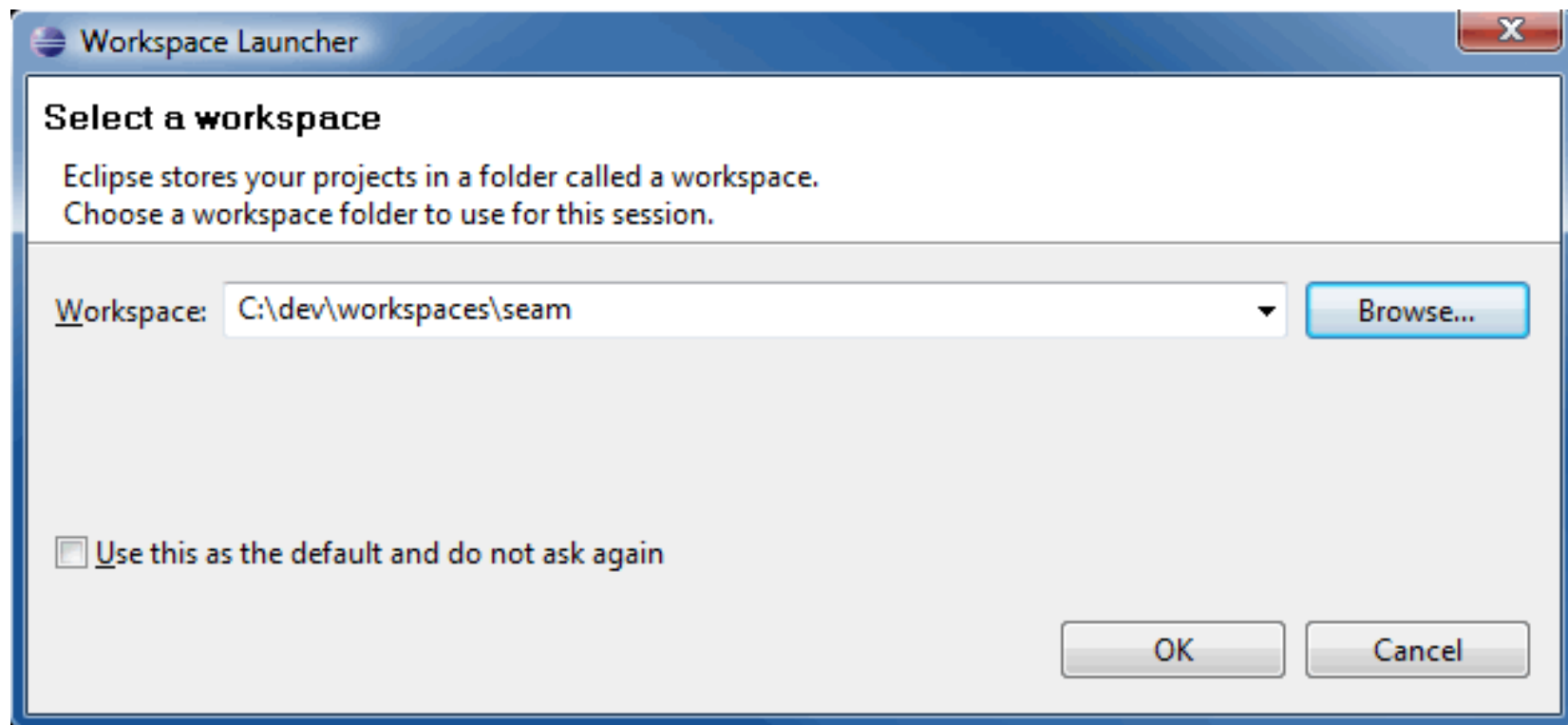
- the Java JRE (Java runtime engine)
- Java JDK (Java development kit): recommended for java development because it allows you to see documentation and source code for the standard Java classes)



- Once you have the installed it, launch Eclipse
- Lab M,H
 - Linux (Java 1.6 and Eclipse 4.3) Windows (Java 1.8 and Eclipse Mars)

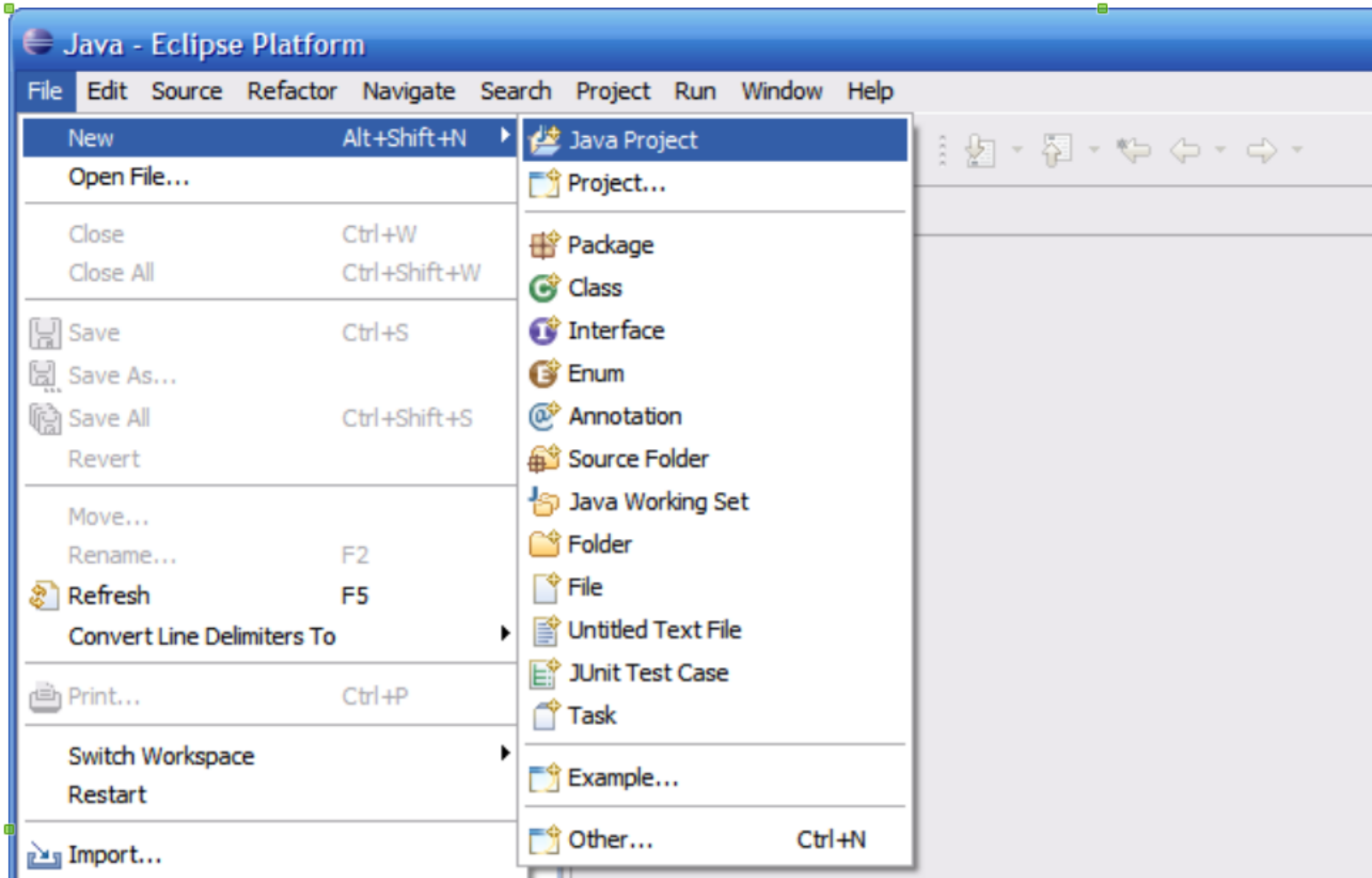
Creating/Selecting a Workspace

- Select the top-level directory (workspace) where you want to store and organize the files of your projects
- It can hold multiple projects



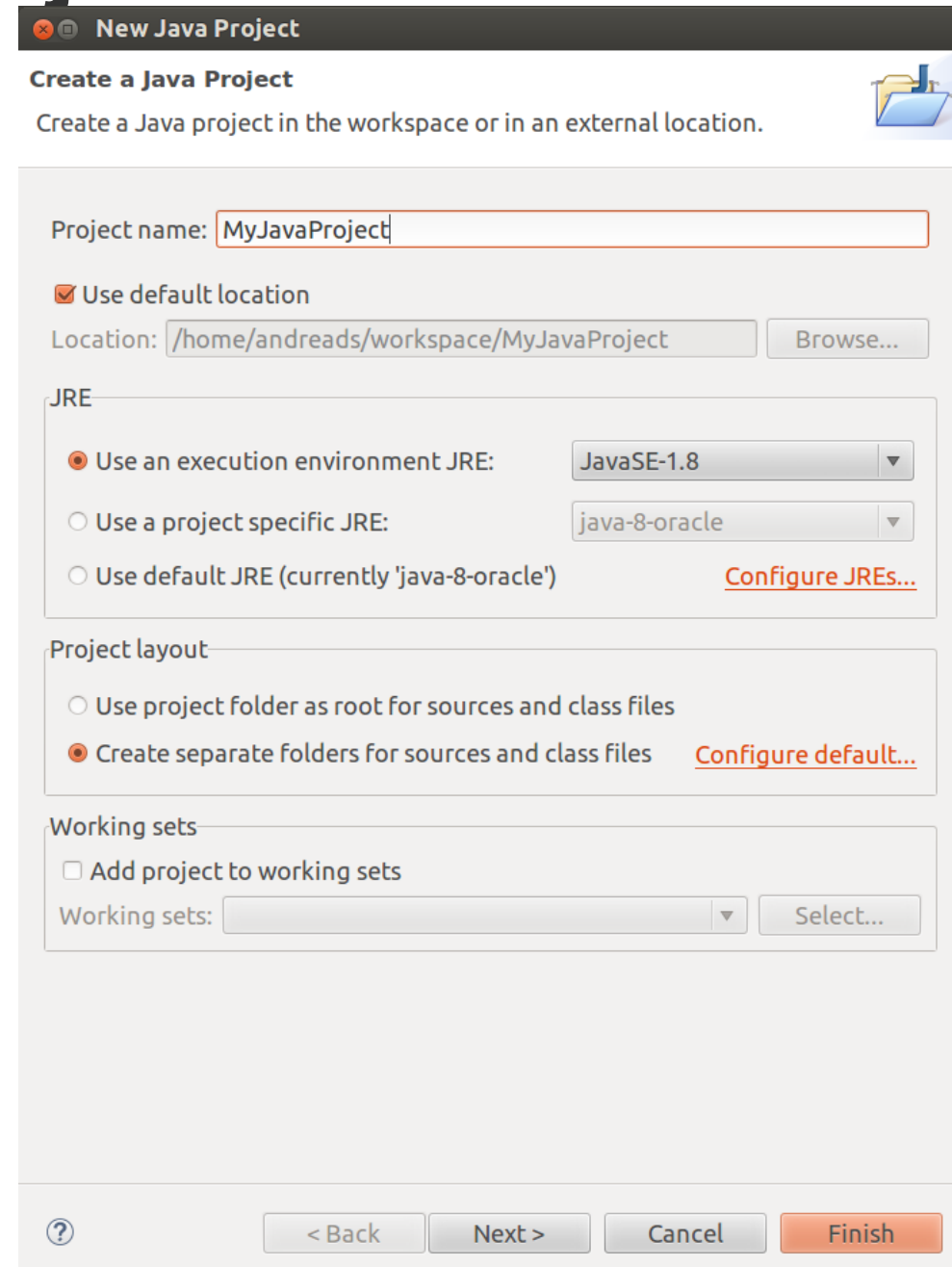
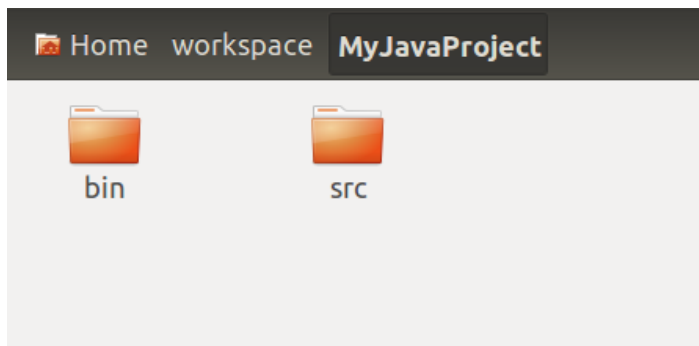
Creating a Project

- Navigates: File → New → Java Project



Properties of a Project

- Project Settings:
 - Project name
 - Location
 - JRE
- The new empty Project should appear under the Package Explorer



Properties of a Project

- Advanced Project Settings (Project → properties):

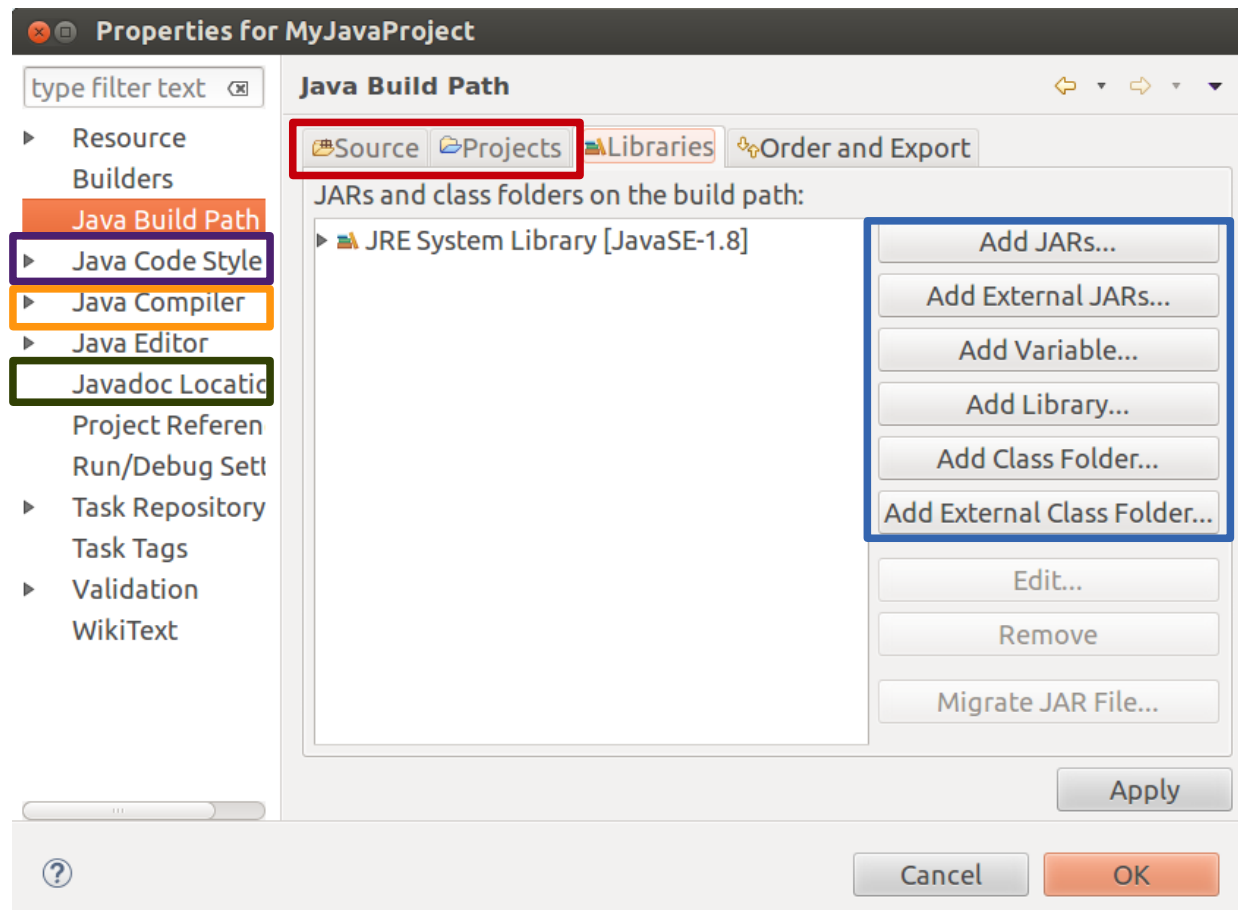
– Adds external libraries

– Link external source code

– Compiler settings

– Code Style

– JavaDoc



Creating a Class

- Class Settings:

- Class name

- Package

- Auto-generated methods

New Java Class

Java Class
Create a new Java class.

Source folder: MyJavaProject/src

Package: mycode

Enclosing type:

Name: Main

Modifiers: public package 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

The Java Perspective

The screenshot displays the Eclipse IDE interface in the Java perspective. The main editor window shows the code for `Main.java` in the `mycode` package. The code is as follows:

```
package mycode;

public class Main {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("My first Eclipse Java Project...");
    }
}
```

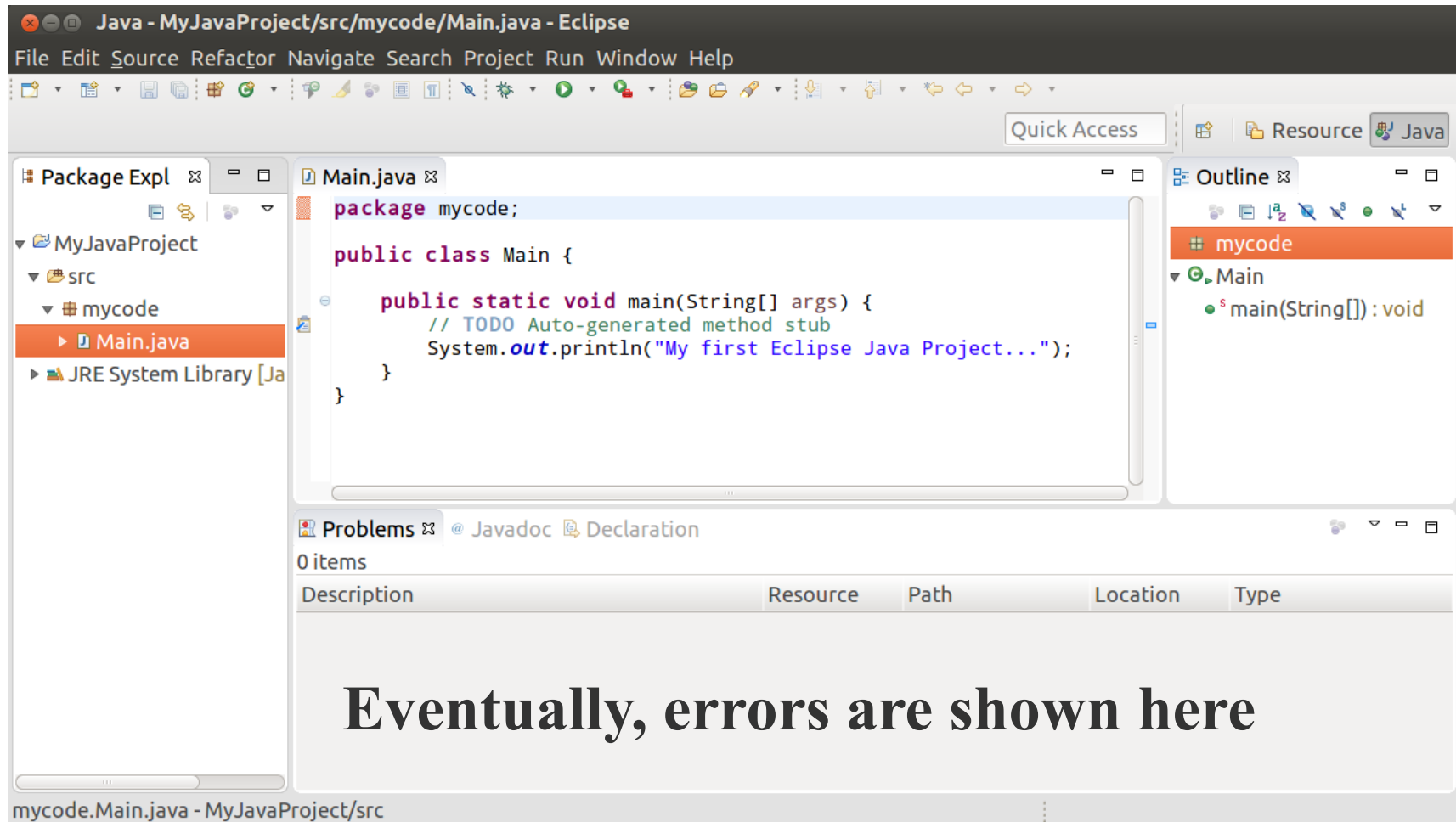
Annotations on the screenshot include:

- Editor**: Points to the main code editor window.
- Methods**: Points to the `main` method in the Outline view on the right.
- Packages and class**: Points to the Package Explorer on the left, which shows the project structure including `MyJavaProject`, `src`, `mycode`, and `Main.java`.
- Console message**: Points to the Problems view at the bottom, which currently shows 0 items.

The status bar at the bottom of the IDE indicates the current file: `mycode.Main.java - MyJavaProject/src`.

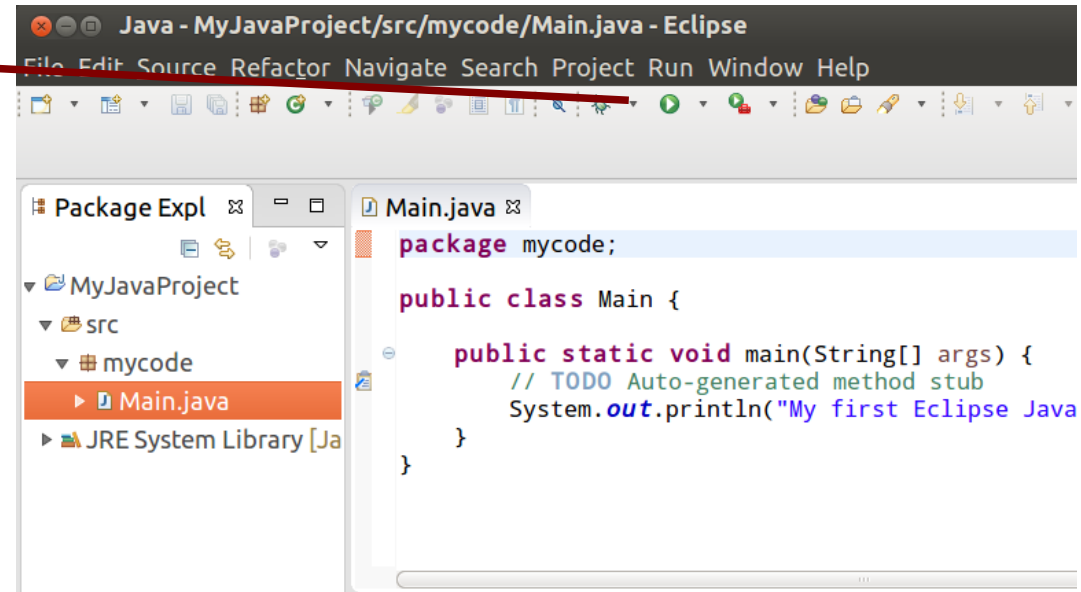
Compiling the code

- Eclipse automatically compile the source



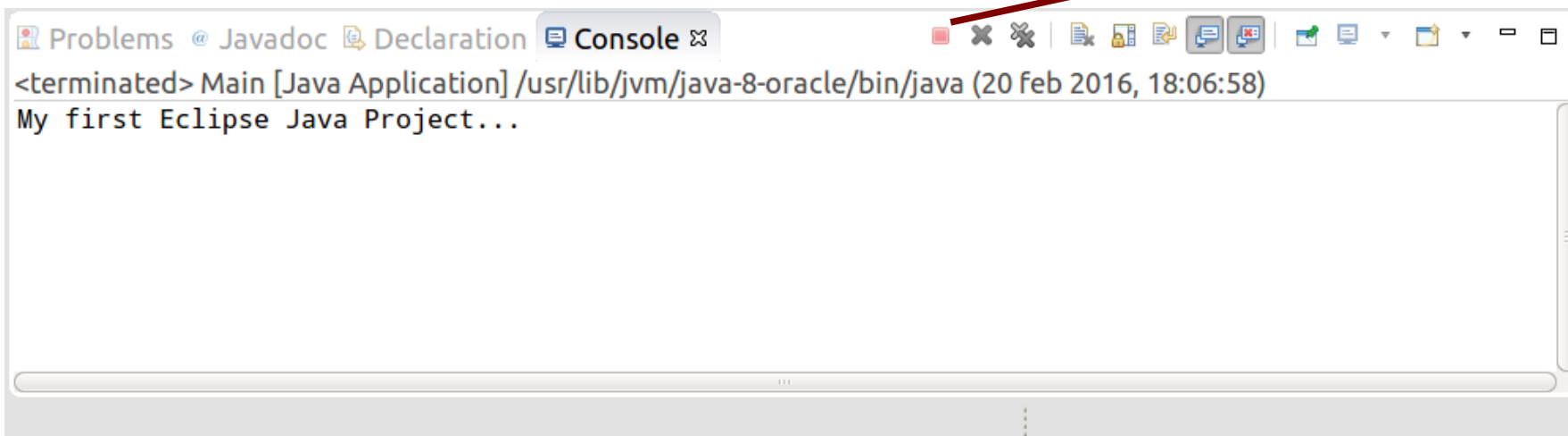
Running the code

- Run button 
- File (or Right-Click on the Java Project and) → Run As → Java Application



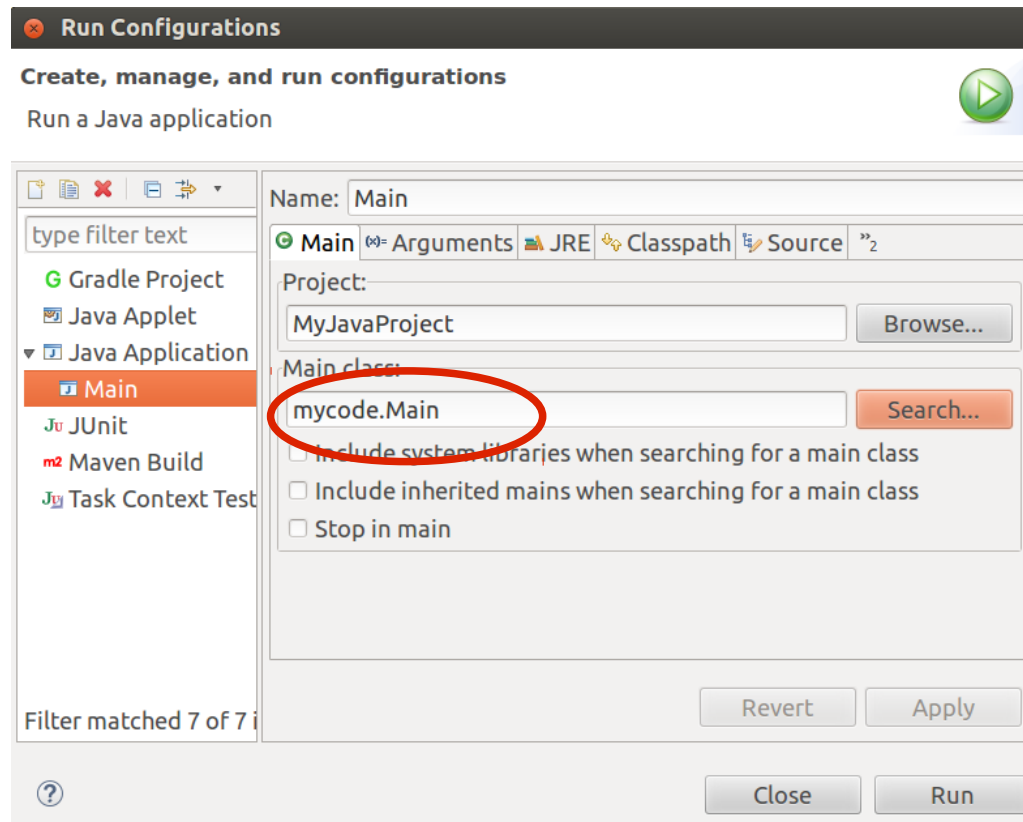
- The output of you program

 Stop button



Run configuration

- Run → Run Configurations:
 - Manages the execution of your Java Project
 - Select the class that contains the main()



Run configuration: Arguments

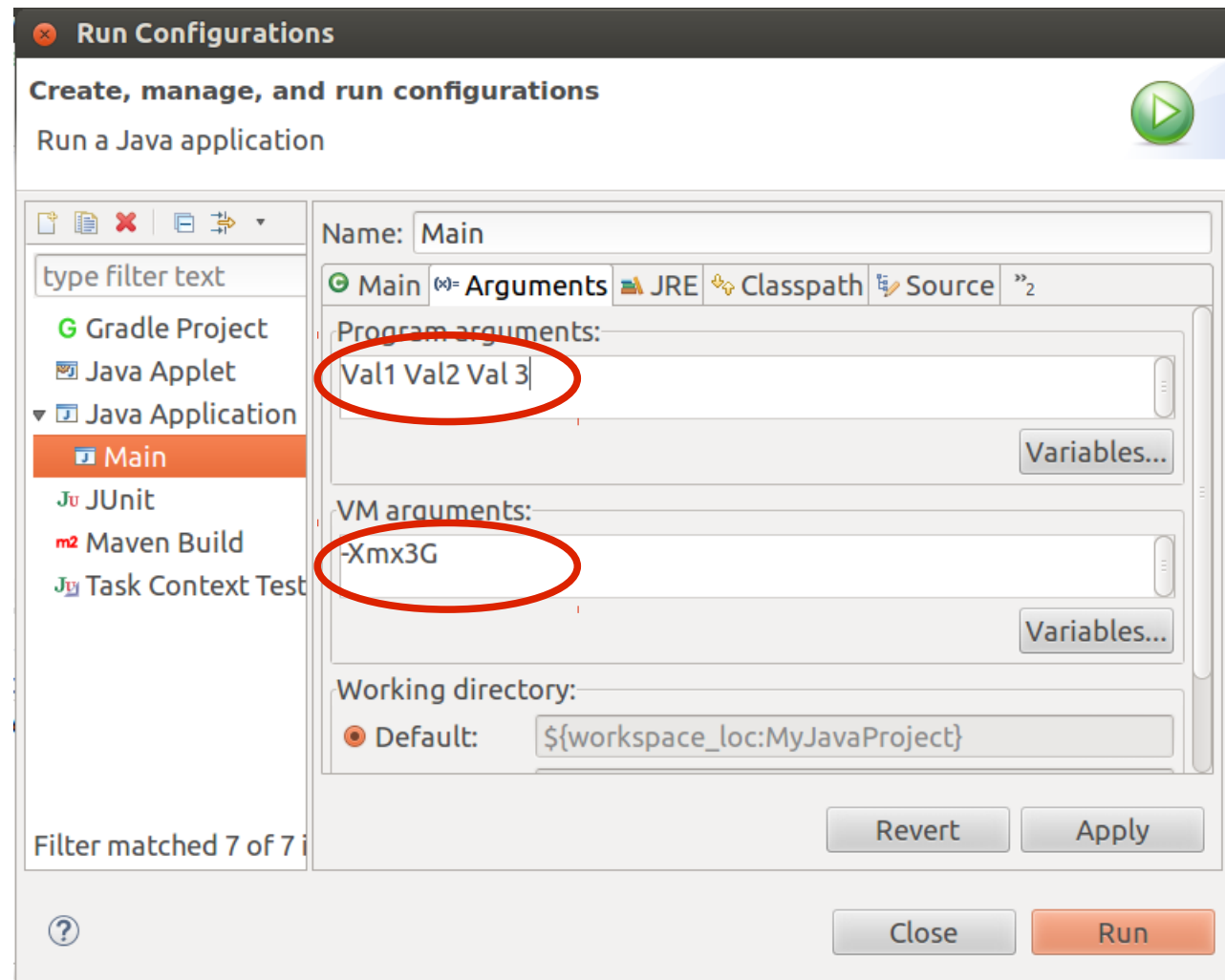
- Run → Run Configurations:

Setting the input
params:

Input parameters of
the main()

Input parameters of
the Virtual Machine

Es: -Xmx3G Set the Heap
size to 3GB



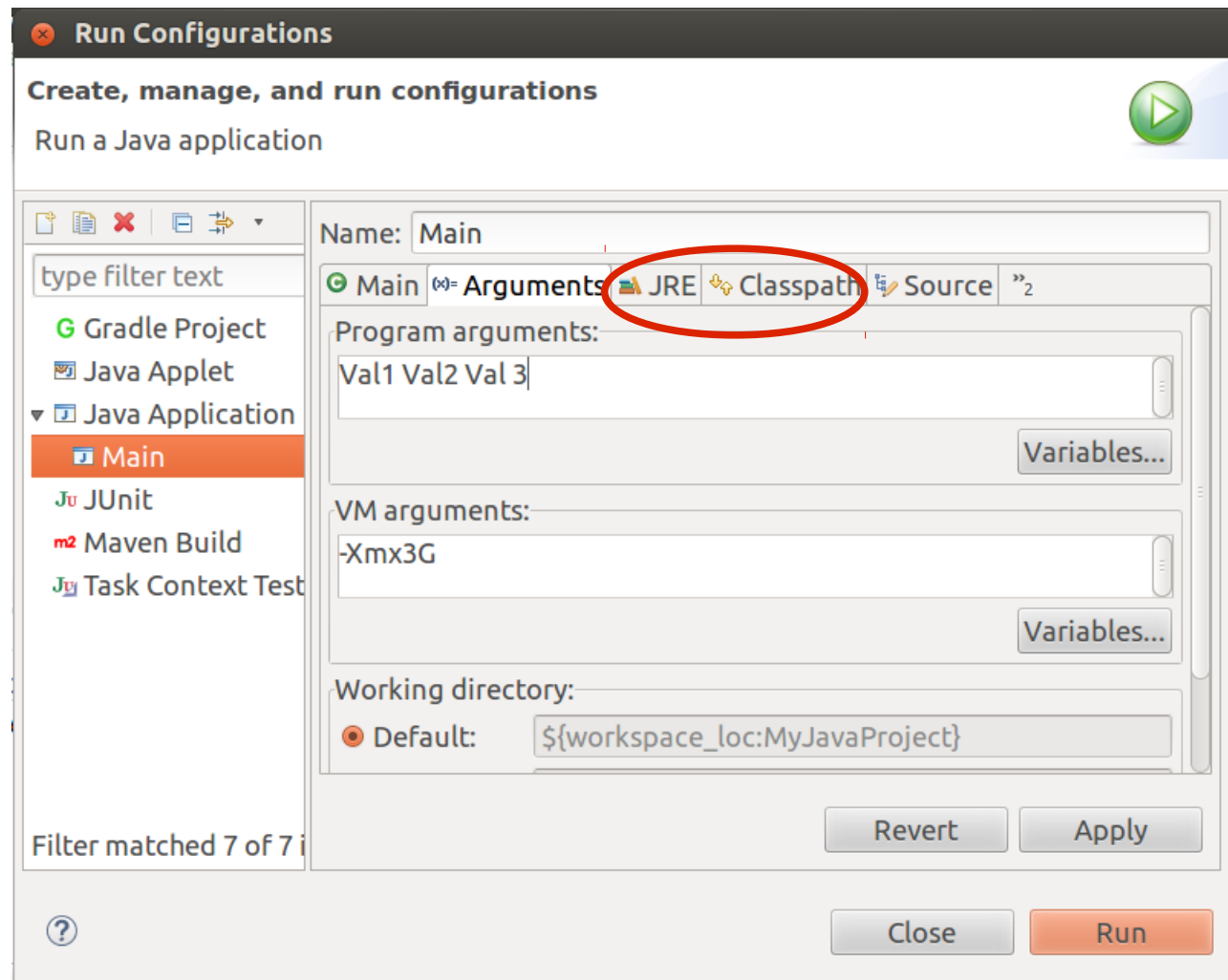
Run configuration: Arguments

- Run → Run Configurations:

Others settings:

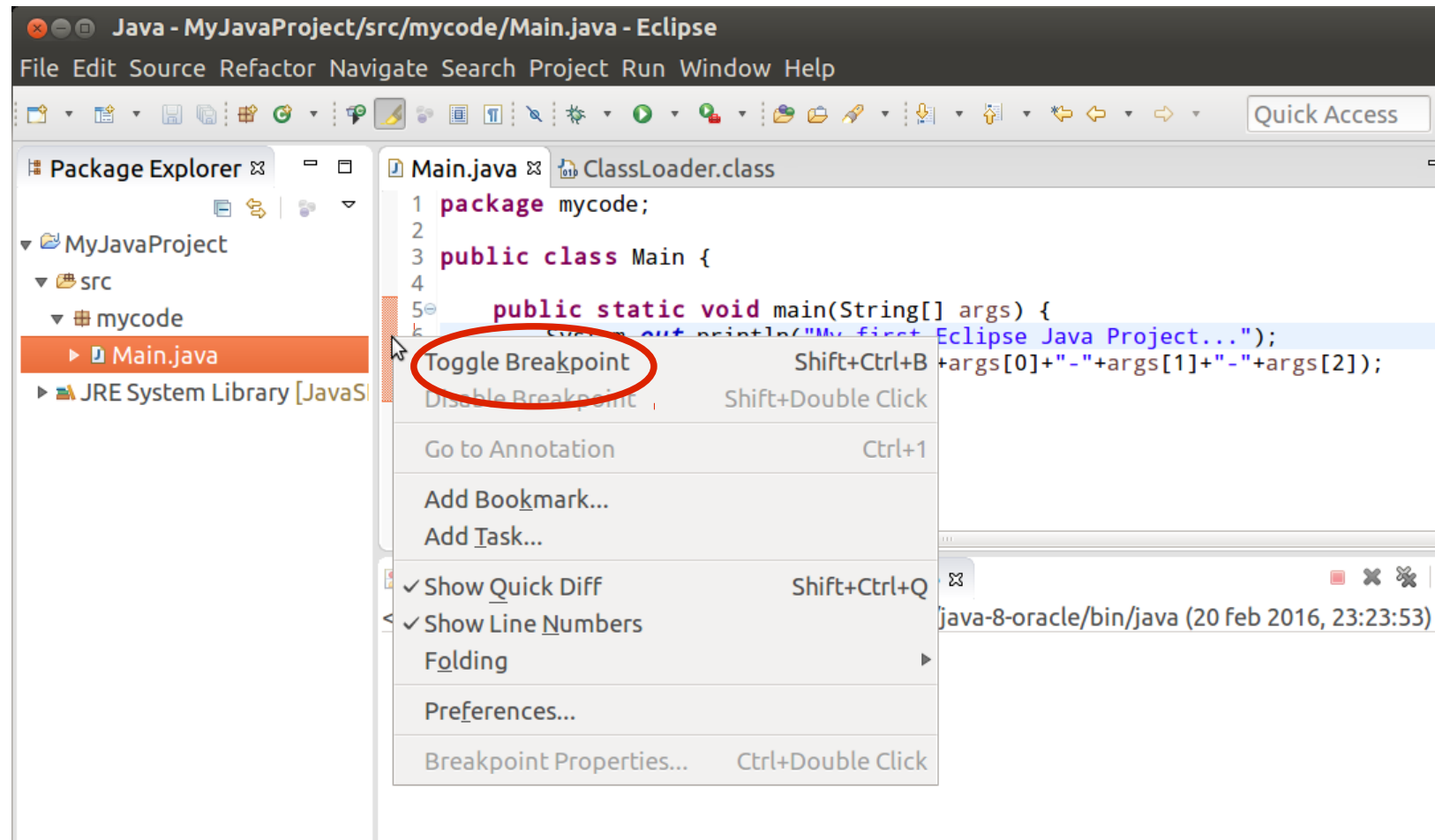
The JRE version

The classpath which contains all the code needed for the execution



Debugging the code

- Toggle Breakpoint: a snapshot that shows the status of your program in this point



Debugging the code

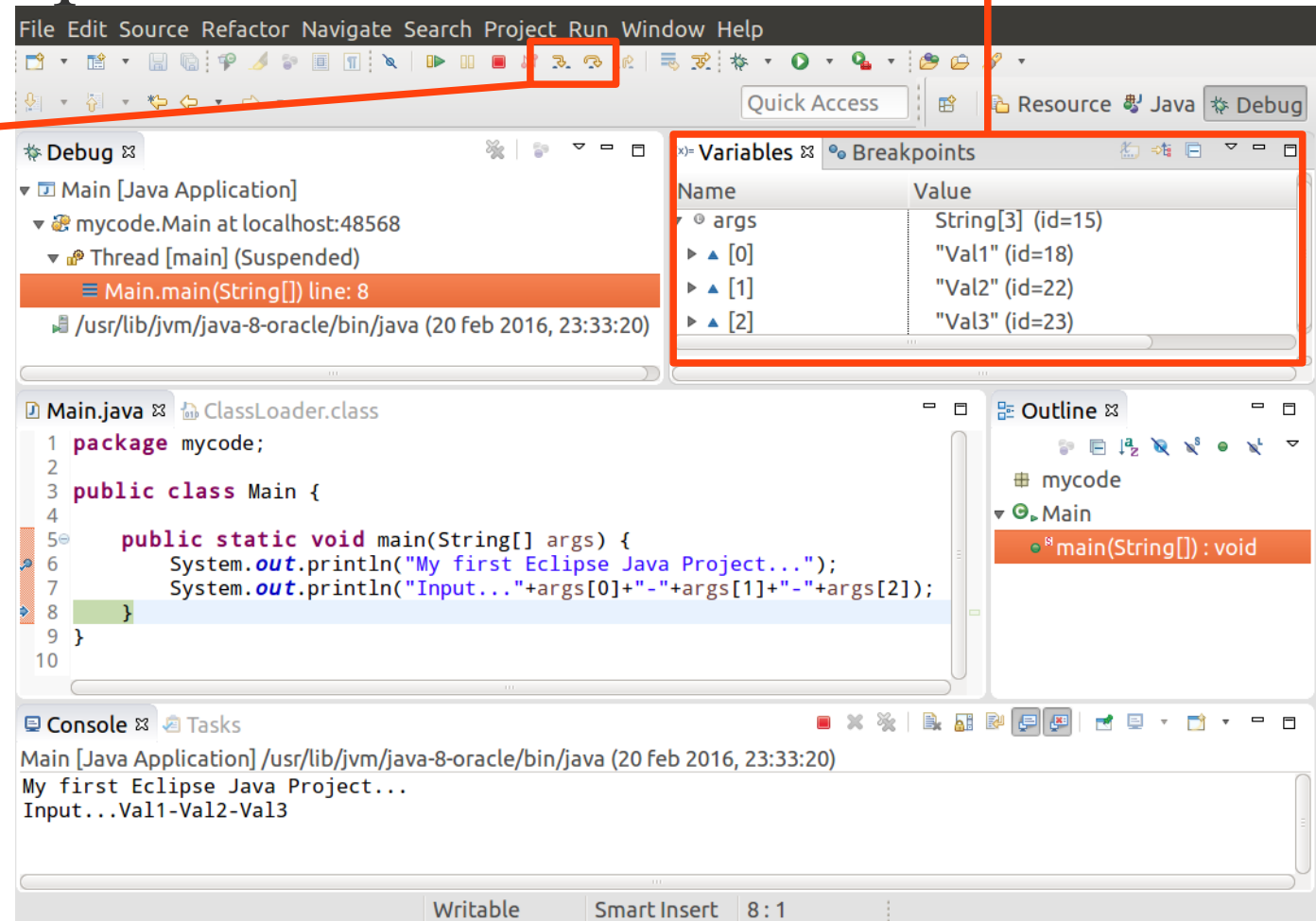
 Run in Debug perspective

 Skip all breakpoints

 Step Into

 Step Over

Variables



The screenshot shows the Eclipse IDE interface with a Java application running in debug mode. The 'Variables' window is open, displaying the following data:

Name	Value
args	String[3] (id=15)
▶ [0]	"Val1" (id=18)
▶ [1]	"Val2" (id=22)
▶ [2]	"Val3" (id=23)

The code editor shows the following code:

```
1 package mycode;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         System.out.println("My first Eclipse Java Project...");
7         System.out.println("Input..." + args[0] + "-" + args[1] + "-" + args[2]);
8     }
9 }
10
```

The console output is:

```
Main [Java Application] /usr/lib/jvm/java-8-oracle/bin/java (20 feb 2016, 23:33:20)
My first Eclipse Java Project...
Input...Val1-Val2-Val3
```

More Info

- Eclipse Web Site: <http://www.eclipse.org>
- Dexter, Mark. "Eclipse And Java For Total Beginners Companion Tutorial Document." Mark Dexter. Licensed under the Educational Community License (2007).
- Eclipse [Wiki](http://wiki.eclipse.org/Eclipse_Articles,_Tutorials,_Demos,_Books,_and_More):
http://wiki.eclipse.org/Eclipse_Articles,_Tutorials,_Demos,_Books,_and_More