

Eclipse: Introduction



Laboratorio di Reti
2014/2015
Prof. Laura Ricci



Speaker: Alessandro Lulli - lulli@di.unipi.it

What is Eclipse?

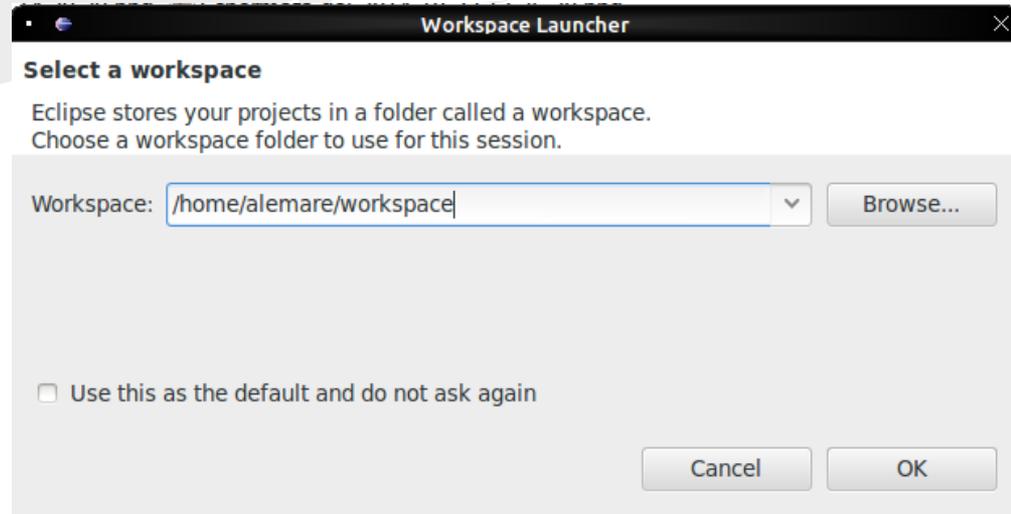
- an Integrated Development Environment (IDE)
 - provides tools for coding, building, running and debugging applications
- designed for Java, now supports many other languages
 - good support for C, C++, Python, PHP, Ruby, etc...
- Eclipse can be downloaded from:
 - <http://www.eclipse.org/downloads/packages/>
- grab “Eclipse IDE for Java Developers”: Eclipse comes bundled as a zip file (Windows) or a tarball (all other operating systems)

Eclipse How-To

- where are my files
- creating a new project
- set project's properties
- run configurations
- debugging

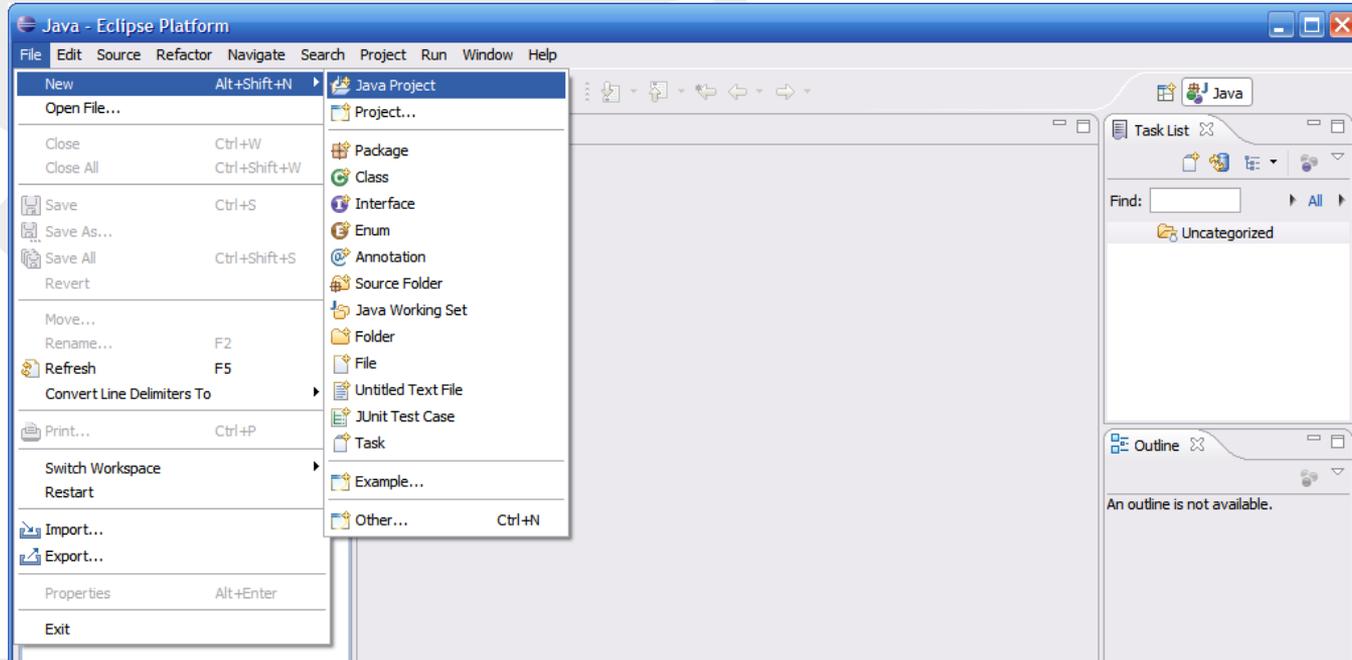
Where are my files?

- when launching Eclipse you need to choose the workspace
- the workspace identifies the root folder where all your projects will be saved



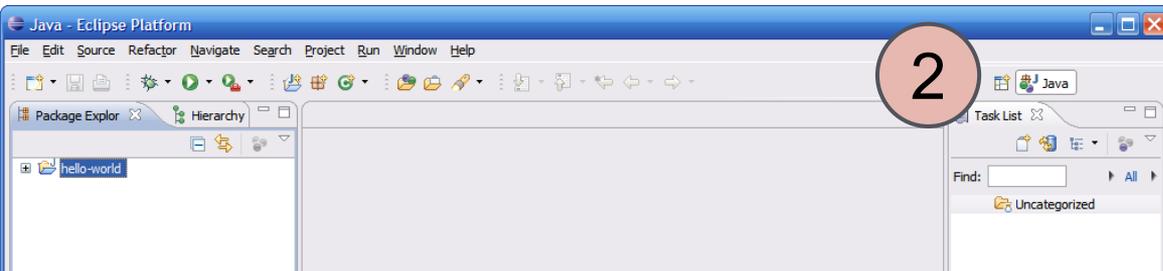
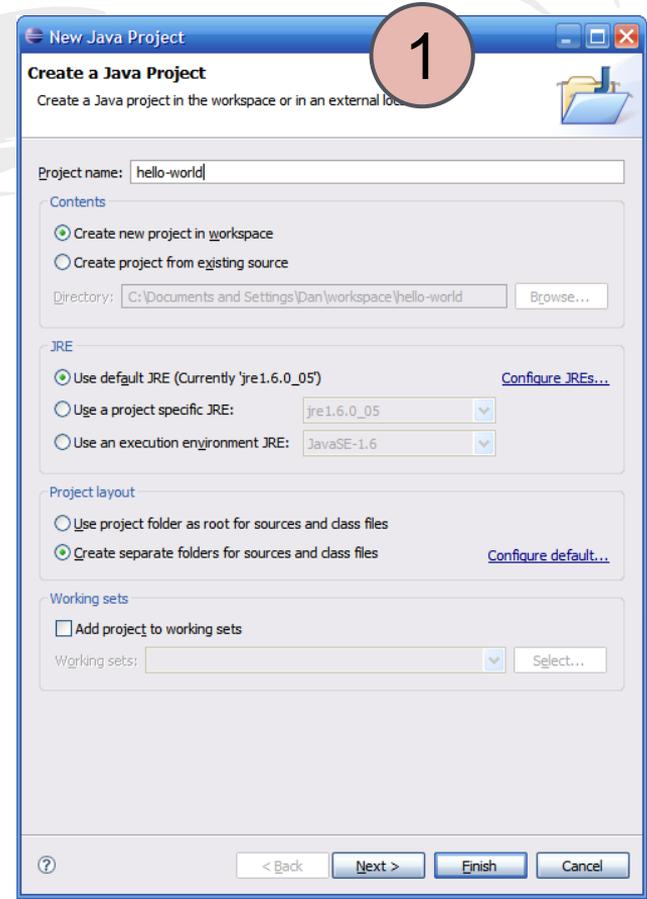
Creating a New Project (I)

- all code in Eclipse needs to live under a project
- to create a project: File->New->Java->Project



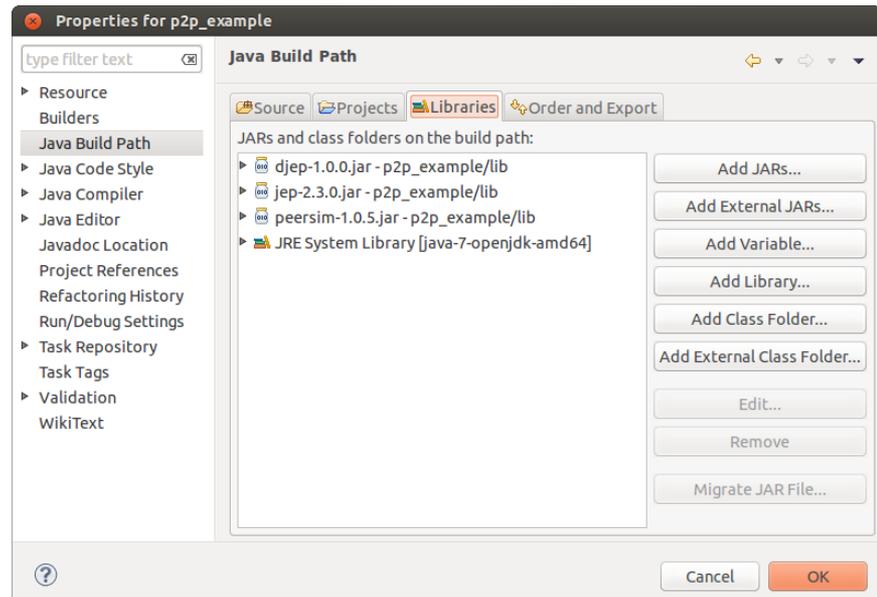
Creating a New Project (II)

1. enter a name for the project, then click Finish
2. the newly created project should then appear under the Package Explorer
3. verify that a new folder has been created in your workspace folder



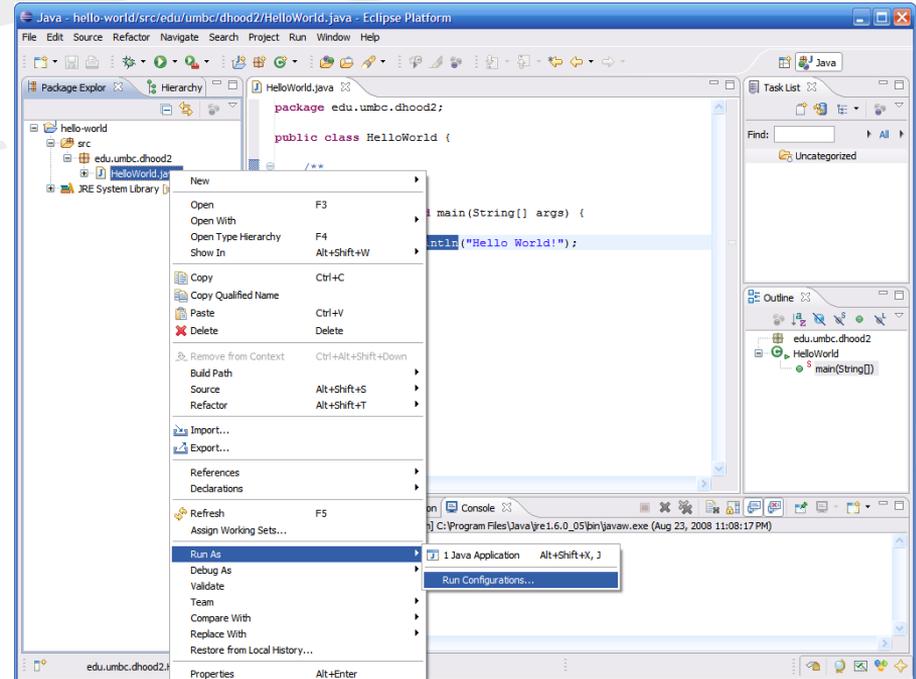
Project's properties

- find under Project -> Properties
- to add third-party libraries
- to set the correct Java library

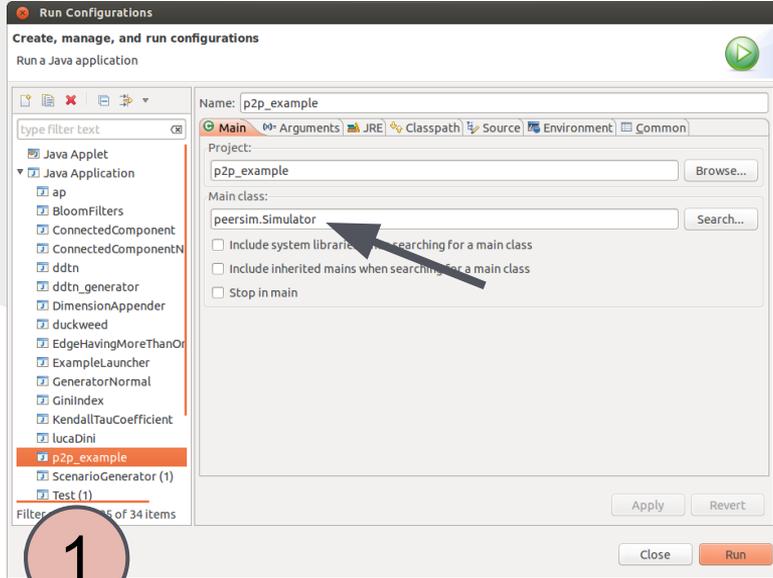


Run configuration (I)

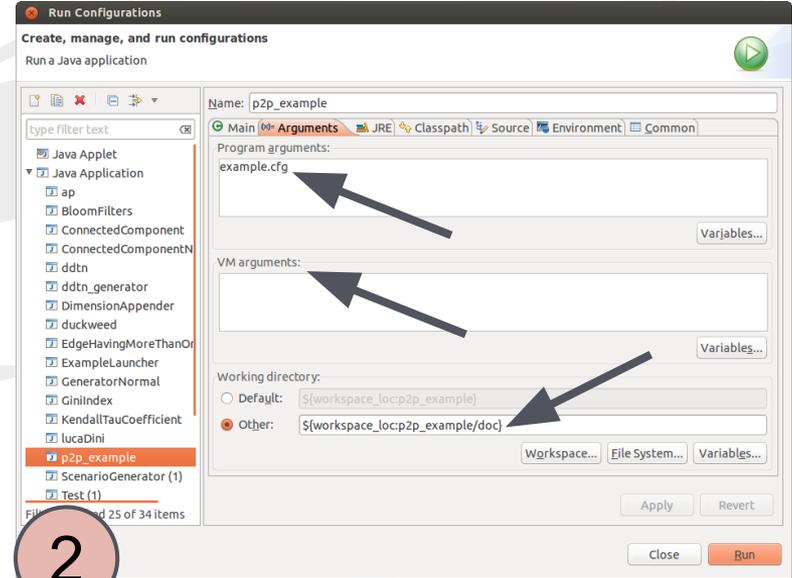
- advanced options for executing a program can be found by right clicking the class then clicking Run As -> Run Configurations



Run configuration (II)



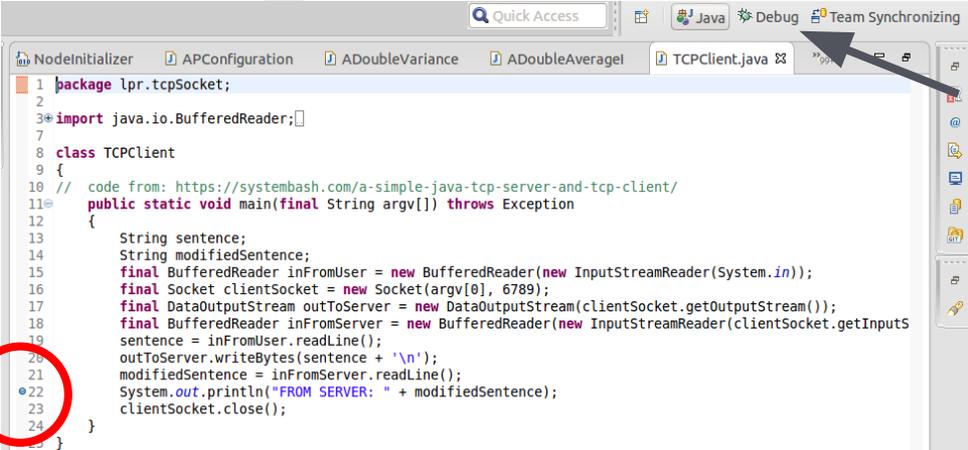
- set the main class



- set program arguments
- set JVM arguments
- set the working directory

How to debug (I)

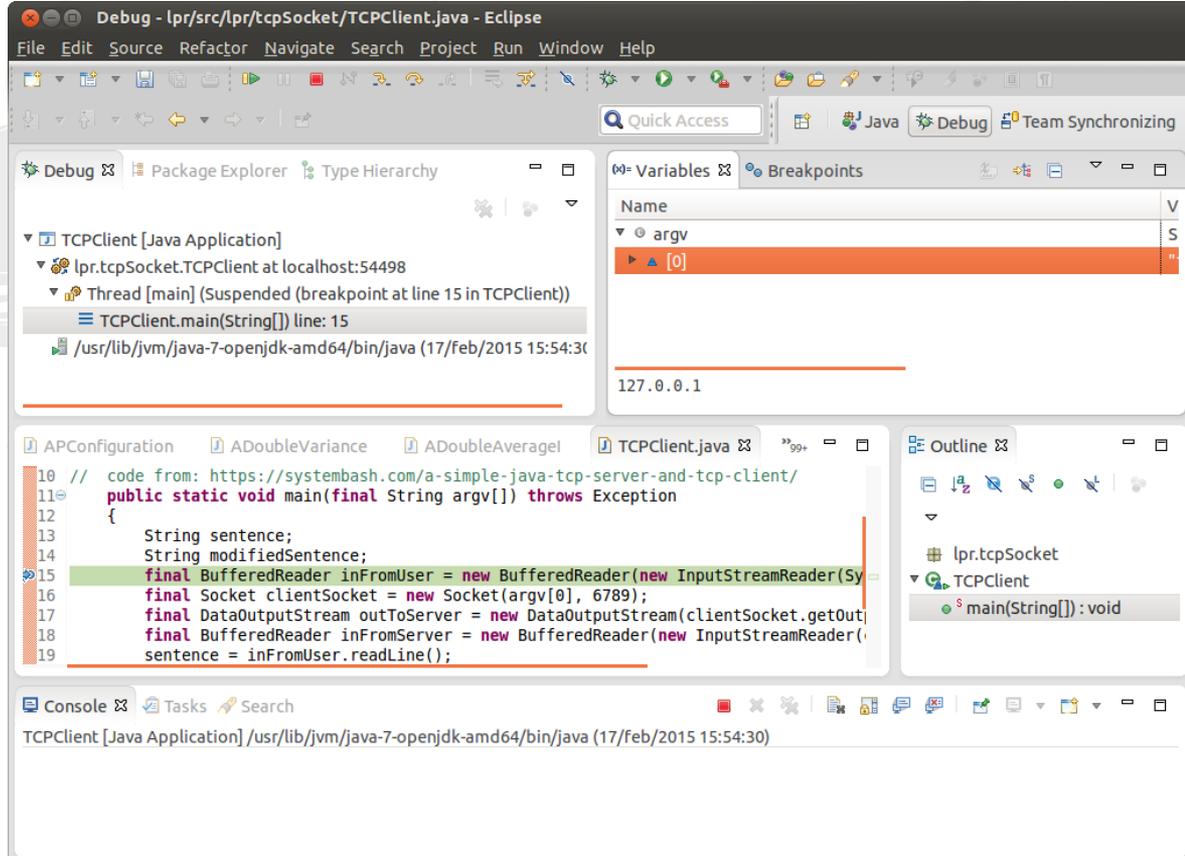
- breakpoints:
 - cause the thread of execution to suspend and return to the debugger
 - to set a breakpoint double-click the left margin
- open the debug perspective



```
1 package lpr.tcpSocket;
2
3 import java.io.BufferedReader;
4
5 class TCPClient
6 {
7
8 // code from: https://systembash.com/a-simple-java-tcp-server-and-tcp-client/
9 public static void main(final String argv[] throws Exception
10 {
11     String sentence;
12     String modifiedSentence;
13     final BufferedReader inFromUser = new BufferedReader(new InputStreamReader(System.in));
14     final Socket clientSocket = new Socket(argv[0], 6789);
15     final DataOutputStream outToServer = new DataOutputStream(clientSocket.getOutputStream());
16     final BufferedReader inFromServer = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));
17     sentence = inFromUser.readLine();
18     outToServer.writeBytes(sentence + '\n');
19     modifiedSentence = inFromServer.readLine();
20     System.out.println("FROM SERVER: " + modifiedSentence);
21     clientSocket.close();
22 }
23 }
24 }
```

How to debug (II)

- step into 
- step over 
- watch variables



The screenshot shows the Eclipse IDE interface during a debug session. The title bar indicates the current file is `lpr/src/lpr/tcpSocket/TCPClient.java`. The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar contains various icons for file operations and debugging. The Package Explorer on the left shows the project structure, with the current thread `Thread [main] (Suspended (breakpoint at line 15 in TCPClient))` selected. The Variables view on the right shows the `argv` array with the value `["127.0.0.1"]`. The Code Editor in the center displays the `main` method of `TCPClient`, with line 15 highlighted. The Outline view on the right shows the `main(String[])` method. The Console at the bottom shows the output of the application.

```
10 // code from: https://systembash.com/a-simple-java-tcp-server-and-tcp-client/  
11 public static void main(final String argv[]) throws Exception  
12 {  
13     String sentence;  
14     String modifiedSentence;  
15     final BufferedReader inFromUser = new BufferedReader(new InputStreamReader(Sy  
16     final Socket clientSocket = new Socket(argv[0], 6789);  
17     final DataOutputStream outToServer = new DataOutputStream(clientSocket.getOut  
18     final BufferedReader inFromServer = new BufferedReader(new InputStreamReader(  
19     sentence = inFromUser.readLine();
```