ITEC 4020

Internet Client-Server Systems

Project 1

January 2021

Introduction and Goal of the Project

The purpose of this project is to learn three foundational technologies of Internet client-server computing and set the foundation for exploring more technologies on your consequent projects.
More specifically, you will learn Inter-process communication using Sockets, XML-RPC, and RMI. The client side of your project will be a Java client program which will be sending requests to another program which acts as a server. The server will act as a façade to access two "banking-type" of services namely *Withdrawal* and *Deposit*. The schematic of the high-level organization of the client, the server, and the back-end services of Withdrawal and deposit for your project is depicted in Figure 1.

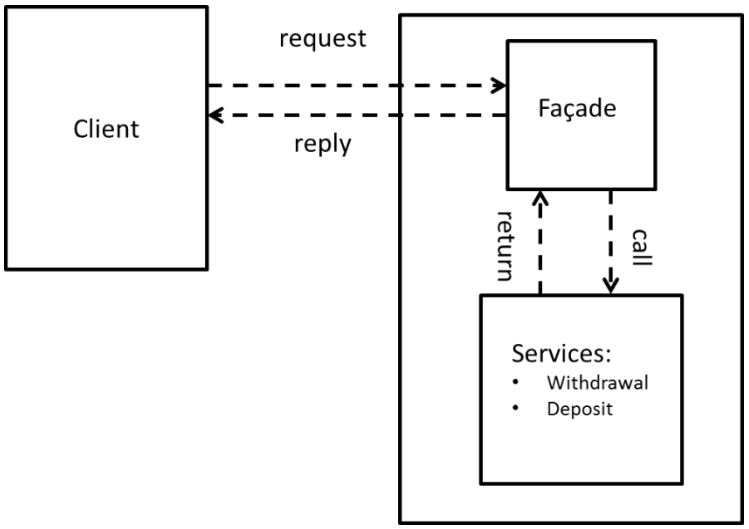


Figure 1. Schematic of client-server organization

You will find on Moodle:

- Instructions on how to install Java EE (Java Enterprise Edition) to your computer
- Instructions on how to install the Tomcat Web Server and integrate it with Eclipse (to be used in the follow-up projects). These two points are also provided on *Appendix I* below.
- Sample code for client-server communication using these three different client-server inter-process communication technologies for you to try and experiment with. The archive to download is called *SampleCodeForClass.zip*. Once unzipped its three sub-directories (RMIExample, SocketExample, and XML-RPC) can be imported as a Java Maven Projects in Eclipse (File → Import → Maven → Existing Maven Projects).
- Code for the Banking application on the file *BankingCodeForClass.zip.* Once unzipped, its *banking* directory can be imported as a Java Maven Project in Eclipse(File → Import → Maven → Existing Maven Projects)

The instructions on how to install and run the sample code are given on *Appendix II*. Running this code will constitute the first part of your project.

The instructions on how to import the banking project on your Eclipse development environment are given in *Appendix III*.

Project Description

Part I. Run the sample code

The first part of your project is to install and run the code. The sample code is organized in three Maven projects (under the workspace name you will choose – see Eclipse installation and run instructions). These projects can be imported to your workspace from where you have extracted them from the .zip files using File → Import → Maven → Existing Maven Projects. (see instructions).

These projects are:

1. *SocketExample.* In this Eclipse Java project and under the *src* subfolder, there is the package *socket.examples*. In this package there are two files *MyClient.java* and *MyServer.java*. Each of these files has a main method, therefore can independently run as standalone applications. In order to be able to see the console output (in case this is not already automatically configured when you installed Eclipse) choose *Window →ShowView → Console.* First run the *MyServer.java* program. In order to do that you need to right click on the file and select *Run As →Java Application* from the drop down menu. Next, run the *MyClient.java* program using the same procedure (i.e. right click on the file and select *Run As →Java Application* from the drop down menu). Once you switch to the client Console (so you can see the input/output console for the client program) you can type a something and hit return. The server receives the string you typed on the client's console and prints it in its own console. To switch between consoles see screenshot in Figure 2. Before you run the next program (using XMLRPC) make sure you ***stop*** the client and the server program (see Figure 3). To stop the client switch to the client's console (see Figure 2), and press the red box shown in Figure 3, and to stop the server do the same after you switch to the server's console.

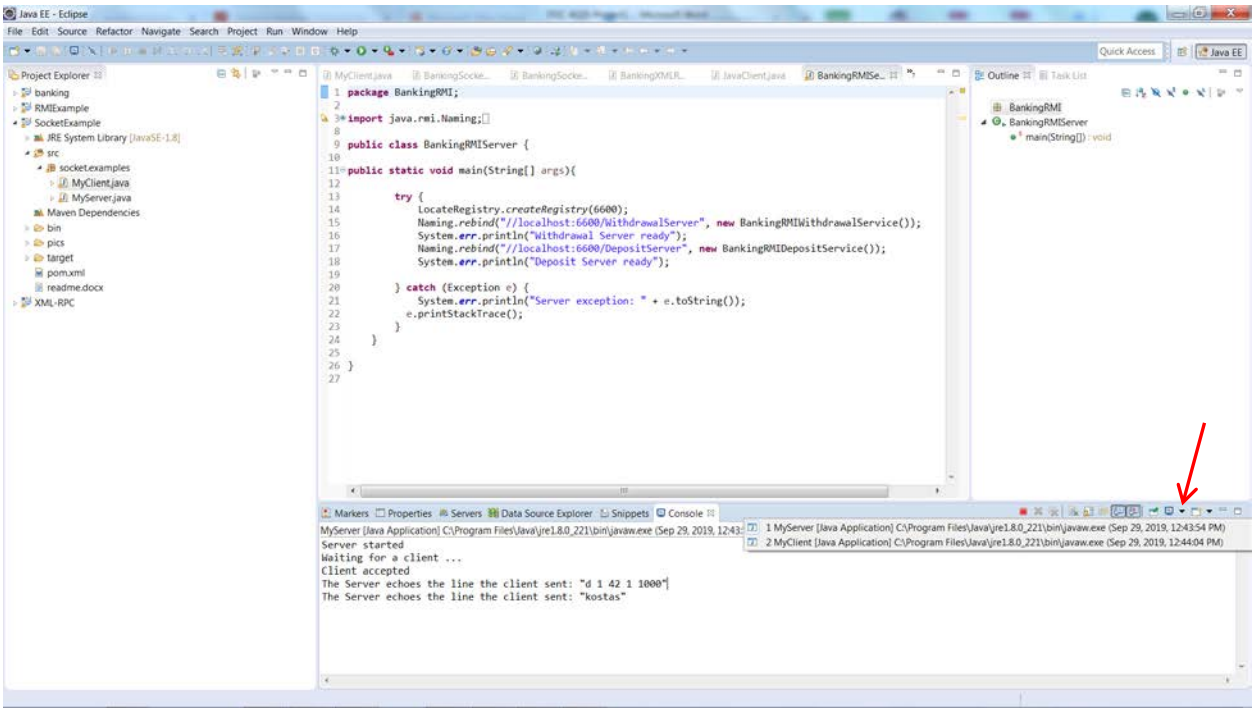


Figure 2. Toggling between Client and Sever Consoles

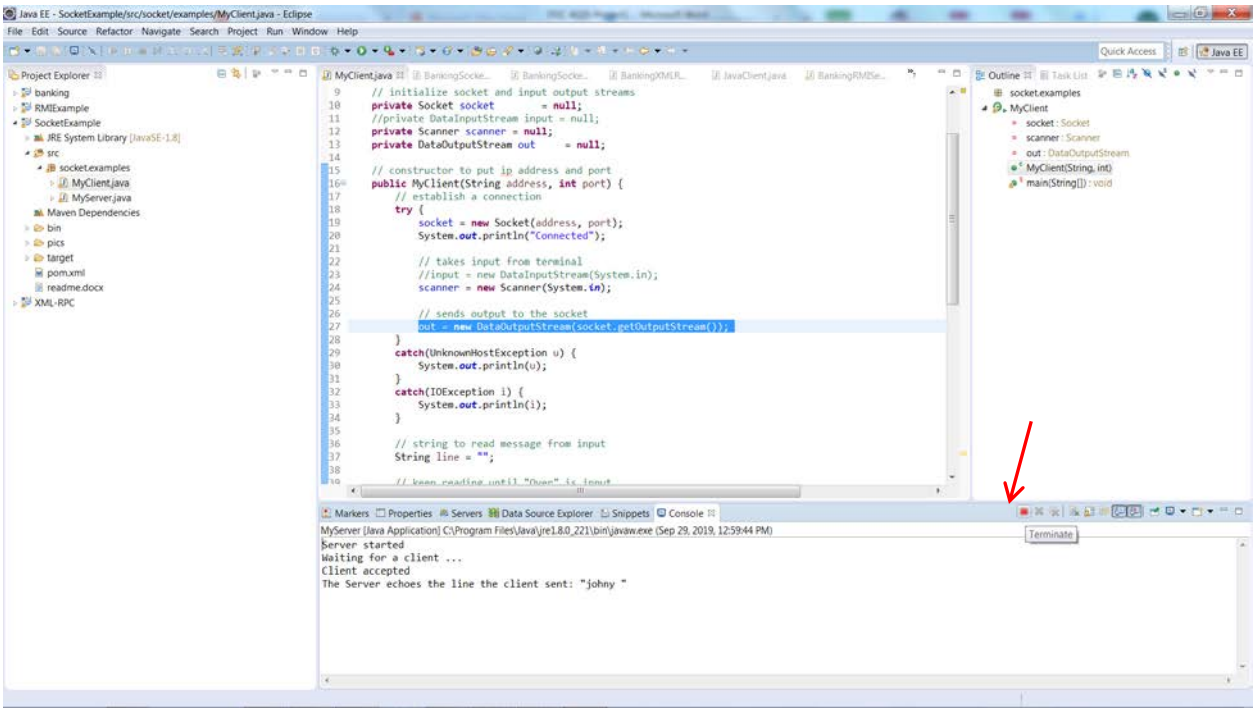


Figure 3. Stopping a Client or a Server program

2. *XML-RPC.* In this Eclipse Java project and under the *src/main/java* subfolder, there is the package *ITEC_4020.XML_RPC.* In this package there are two files *JavaClient.java* and *JavaServer.java*. Each of these files has a main method, therefore can independently run as standalone applications. In order to be able to see the console output (in case this is not already automatically configured when you installed Eclipse) choose *Window → ShowView → Console.* First run the *JavaServer.java* program. In order to do that you need to right click on the file and select *Run As → Java Application* from the drop down menu. Next, run the *JavaClient.java* program using the same procedure (i.e. right click on the file and select *Run As → Java Application* from the drop down menu). Once you switch to the client Console (so you can see the input/output console for the client program) you will prompted to provide two numbers. The server receives these two numbers and calls its service which is to add these two numbers and return the result to the client, which prints the result in its Console. Before you try the RMI part stop the client and the server process of the XML RPC example.
3. *RMIExample*. In this Eclipse Java project and under the *src/main/java* subfolder, there is the package *rmi.examples.* In this package there are several files. The two files of direct interest to run the RMI example code is the *MyClient.java* and *TheServer.java* files. Each of these files has a main method, therefore can independently run as standalone applications. In order to be able to see the console output (in case this is not already automatically configured when you installed Eclipse) choose *Window → ShowView → Console.* First run the *TheServer.java* program. In order to do that you need to right click on the file and select *Run As → Java Application* from the drop down menu. Next, run the *MyClient.java* program using the same procedure (i.e. right click on the file and select *Run As → Java Application* from the drop down menu).In this example, the server exposes two services (MyService1.java, and MyService2.java) which implement the interfaces RMIInterface.java and RMIInterface2.java respectively. One you run the client follow the prompts on the windows opened.

Deliverable

Your first deliverable will have two parts.

Part I. Comment on the Sample Code

Run the three examples (Sockets, XMLRPC, RMI) and briefly comment on what is happening in each example. More specifically comment on *a)* which library methods are used to pass data or performing an invocation of a service on the server program (for example in the Sockets example you send data using the *out.writeUTF(<string>)* call); *b)* the mechanics of each technology (e.g. in the Sockets example you pass a string from the client application to the server application, and the server interprets it, and does something – in this case just prints the string it receives from the client application) in terms of the messages exchanged, and how the access of a client's application to the services offered by the server is achieved. Part I has to be organized as follows:

1. Use one section heading per technology (Sockets, XML-RPC, RMI)
2. In each section heading briefly discuss:
    a. The functionality of the library methods utilised in the different technologies (e.g. discuss what out.writeUTF(<string>) method is used for in the sockets example)
    b. The mechanics of each technology

Part II. Complete and run the Banking application

The second part of your first deliverable is to complete and run the banking application. The banking application is organized under a Maven Eclipse project called *banking* which you can import as a Maven Project (File → Import → Maven → Existing Maven Projects) from the directory you have extracted the *BankingCode.zip* file.
Once you import the *banking* project into your Eclipse environment, and under the subdirectory src/main/java you will see five packages as follows:

1. *BankingSockets.* In this package you will find the following files:
    a. *BankingSocketsClient.java.* This file is the banking client application. It has a main method and can run as a standalone client with its own Console.
    b. *BankingSocketsServer.java.* This file is the banking server application. It has a main method and can run as a standalone client with its own Console.
    c. *BankingSocketsInputReader.java.* This file has the utility to print the prompt messages to the client, so the user can add the details of a transaction (i.e. whether it is a deposit, the card number, the pin, the account, and the amount). Legal values for the example data base are: card number value: *1*, pin value *42*, account number value *1*, amount *what ever you choose*. Keep in mind for withdrawals the max daily limit is $300. Above this limit an exception will be raised.
    d. *UserInputTokenizer.java.* This file has the utility to tokenize the input sent from the client application and pass the individual tokens as parameters to the *perform* method of the *Withdrawal* and *Deposit* services (see Withdrawal.java and Deposit.java in the transactions package described below).
2. *BankingXMLRPC.* In this package you will find the following files:
    a. *BankingXMLRPCClient.java.* This file is the banking client application for the XML RPC technique. It has a main method and can run as a standalone client with its own Console.
    b. *BankingXMLRPCServer.java.* This file is the banking server application for the XML RPC technique. It has a main method and can run as a standalone client with its own Console.
    c. *BankingXMLRPCInputReader.* This file has the utility to print the prompt messages to the client, so the user can add the details of a transaction (i.e. whether it is a deposit, the card number, the pin, the

account, and the amount). Legal values for the example data base are: card number value: *1*, pin value *42*, account number value *1*, amount *what ever you choose*. Keep in mind for withdrawals the max daily limit is $300. Above this limit an exception will be raised.

3. *BankingRMI.* In this package you will find the following files:
    a. *BankingRMIClient.java.* This file is the banking client application for the RMI technique. It has a main method and can run as a standalone client with its own Console.
    b. *BankingRMIServer.java.* This file is the banking server application for the RMI technique. It has a main method and can run as a standalone client with its own Console.
    c. *BankingRMIWihdrawalService.java.* This file contains the code that calls the Withdrawal service (see the transactions package description below).
    d. *BankingRMIWithdrawalInterface.java.* This is the interface the *BankingRMIWihdrawalService* implements.
    e. *BankingRMIDepositService.java.* This file contains the code that calls the Deposit service (see the transactions package description below).
    f. *BankingRMIDepositInterface.java.* This is the interface the *BankingRMIDepositService* implements.
    g. *BankingRMIInputReader.java.* This file contains the code for the system to prompt the user to provide his or her input. Note as above that legal values for the example data base are: card number value: *1*, pin value *42*, account number value *1*, amount *what ever you choose*. Keep in mind for withdrawals the max daily limit is $300. Above this limit an exception will be raised.
4. *Transactions.* This package has the following files
    a. *Withdreaw.java.* It contains the code that offers the actual withdrawal service.
    b. *Deposit.java.* It contains the code that offers the actual deposit service.
    c. *Accounts.java.* It is a file that defines some 2-d matrix variables that "simulate" the database. Note that each line pertains to a different user.
    d. *CredentialsChecker.java.* It contains the code to provide the valid pins. The array variable PIN has the valid PINs for account 0, account 1, etc. For example for account 1 the valid PIN is 42.
5. *transaction.exceptions*. It contains the code of all exceptions that can be raised in the banking system.

Fill in the parts of the code missing in the following files:

1. BankingSocketsClient.java
2. BankingSocketsServer.java
3. BankingXMLRPCClient.java
4. BankingXMLRPCServer.java
5. BankingRMIClient.java
6. BankingRMIServer.java

The points the code needs be added is flagged in the code as ***// ITEC_4020: ADD CODE HERE***

What to submit

Submit by February 8 midnight the following in one archive:

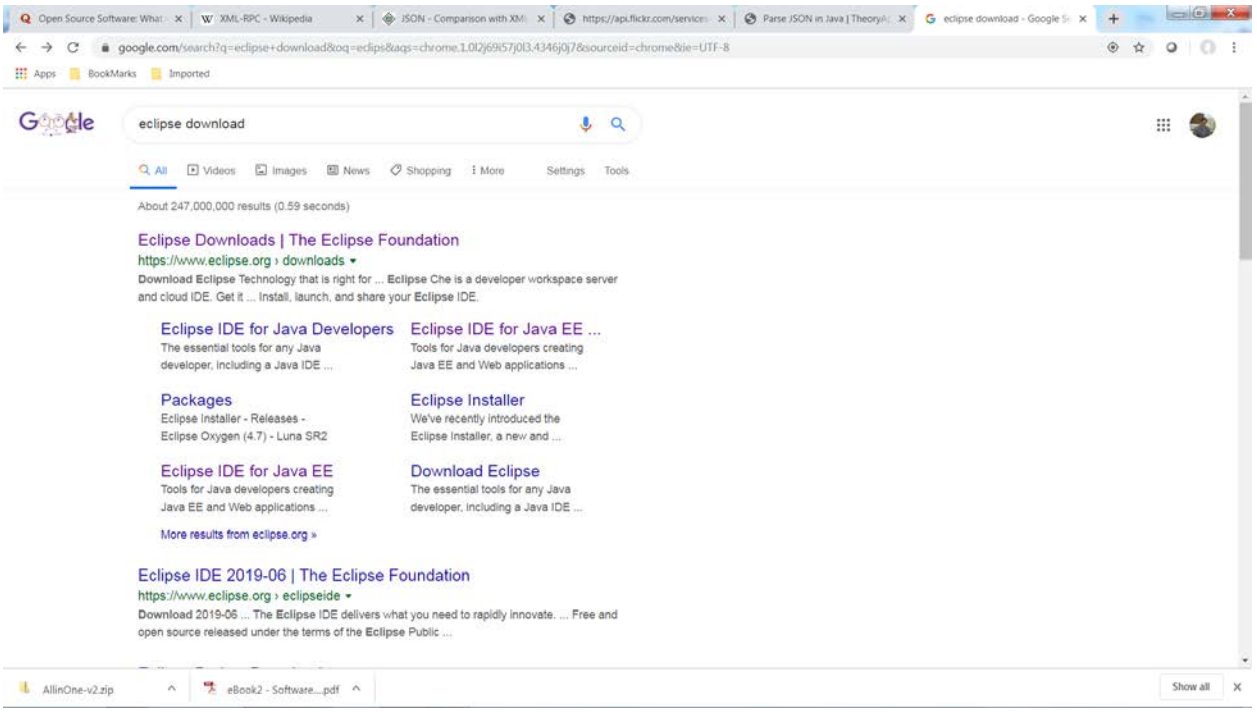
1. Your report for Deliverable 1 (Part I)
2. Your code in the same structure and packages as given to you in a .zip file (as archive inside your submission archive) (Part II)

Submit your archive file to kkontog@yorku.ca with the subject "Project1-Group<your-group-ID>"

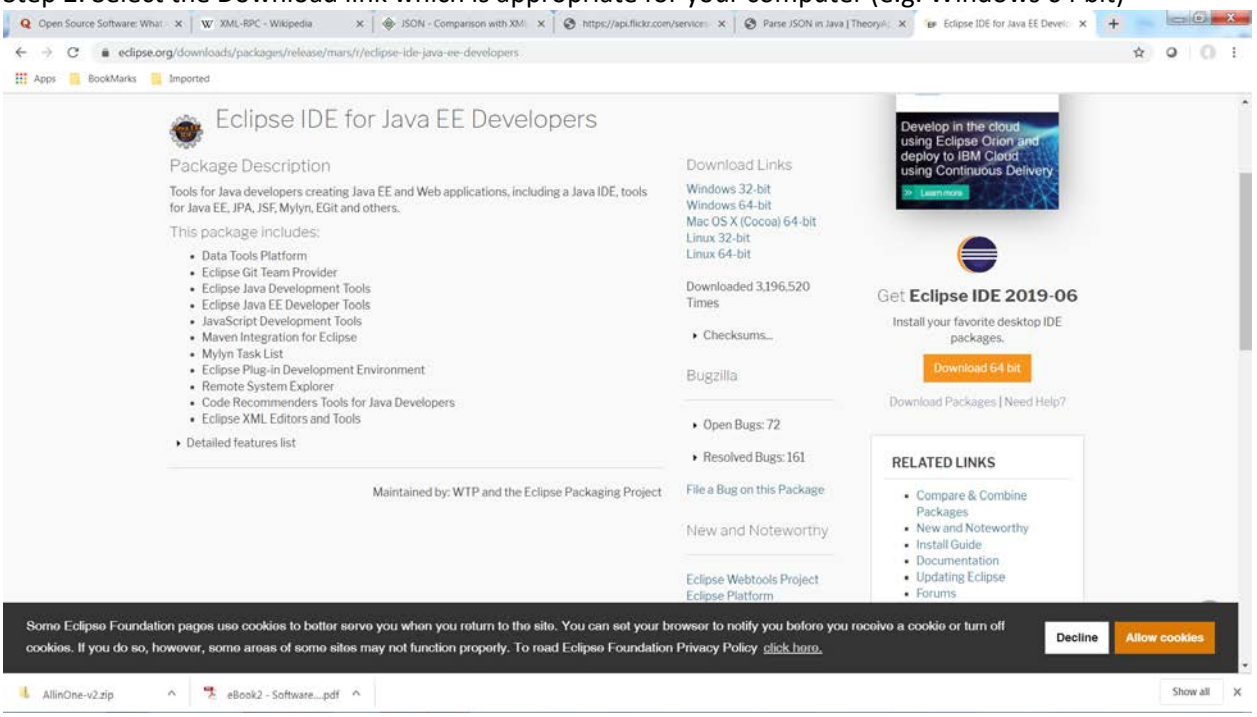
APPENDIX I

Install Eclipse and Integrate Tomcat with Eclipse

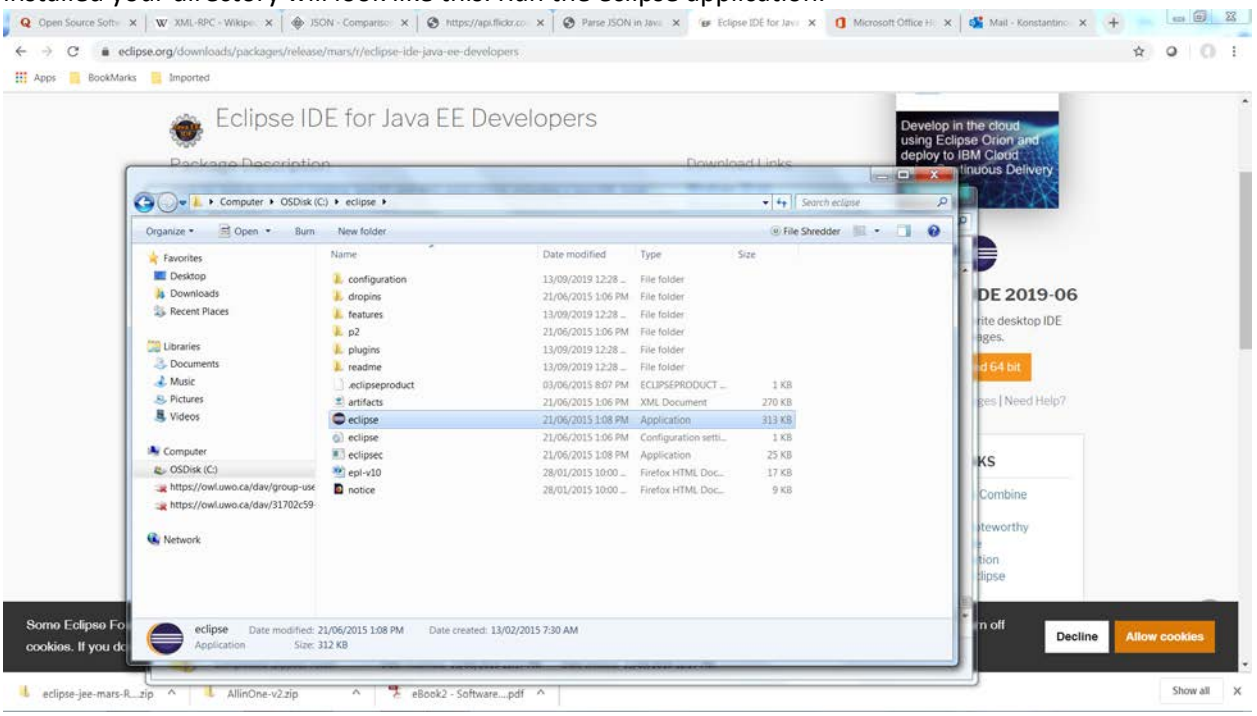
Step 1. Search for Eclipse IDE for Java EE. Install Eclipse Java EE (**NOT THE SIMPLE ECLIPSE**). Search for Eclipse IDE Java EE downloads.



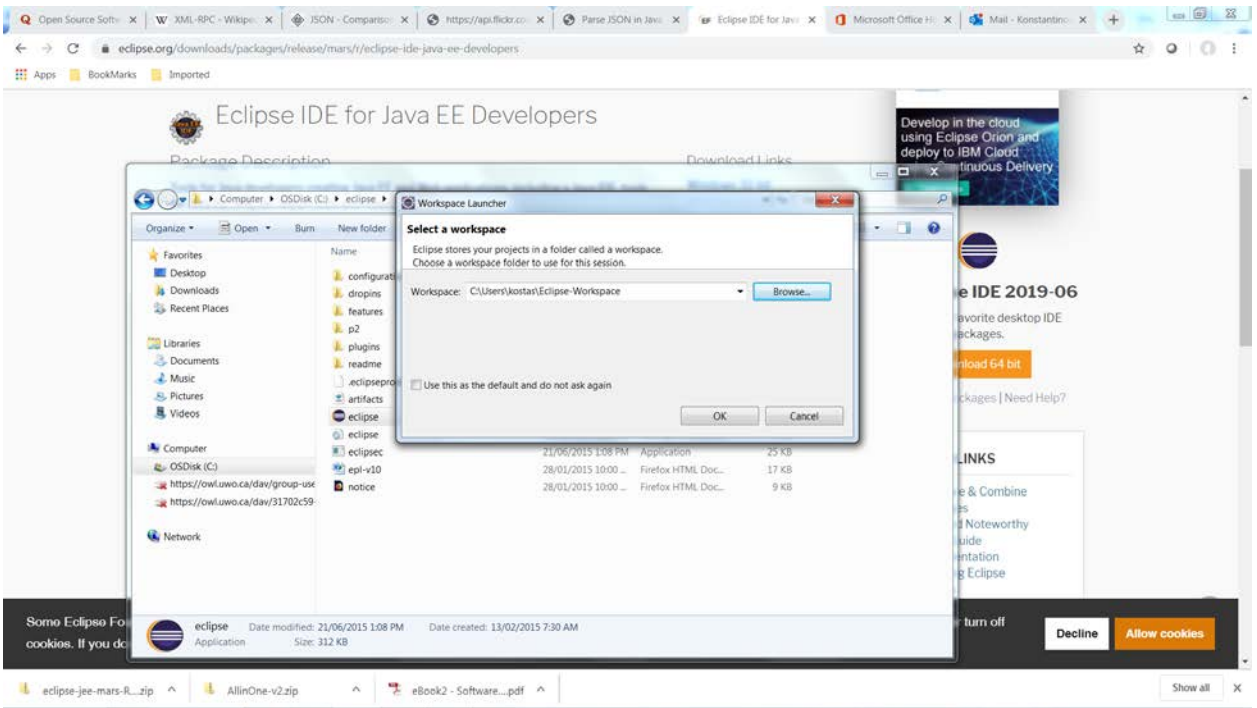
Step 2. Select the Download link which is appropriate for your computer (e.g. Windows 64 bit)

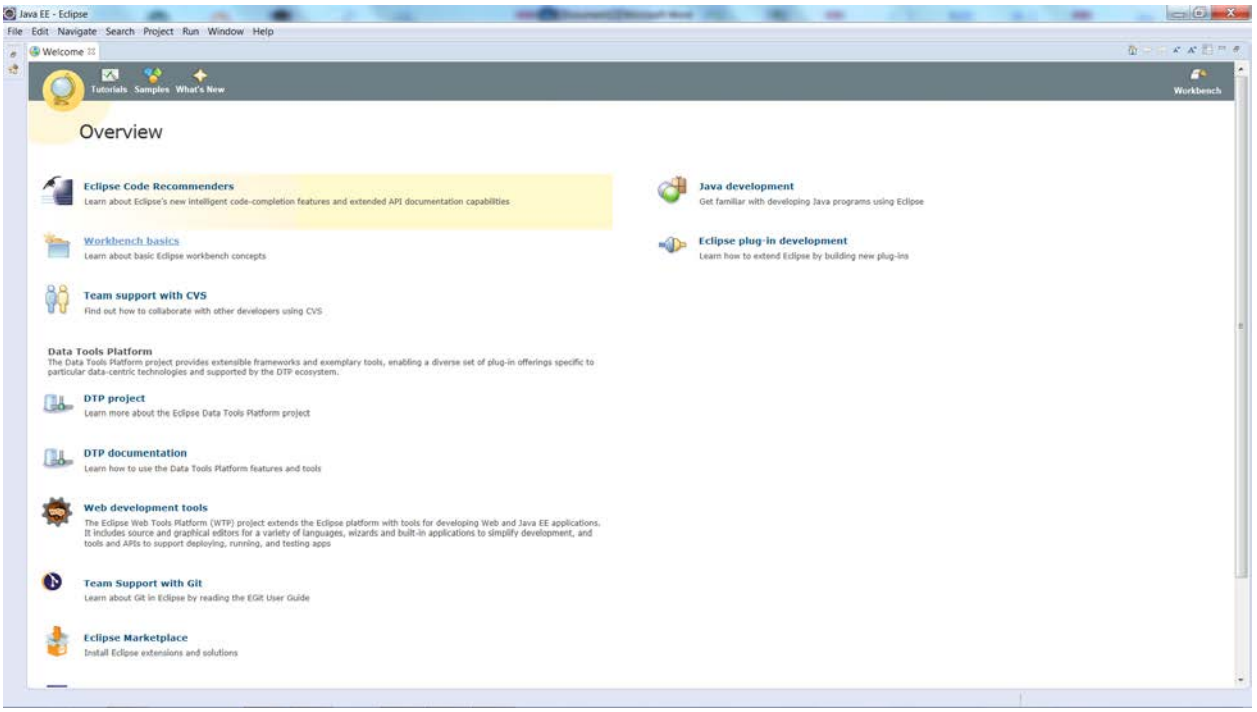


Step 3. Download Eclipse and install it on a directory of your choice. In this example is on C:\eclipse directory. Once installed your directory will look like this. Run the eclipse application.

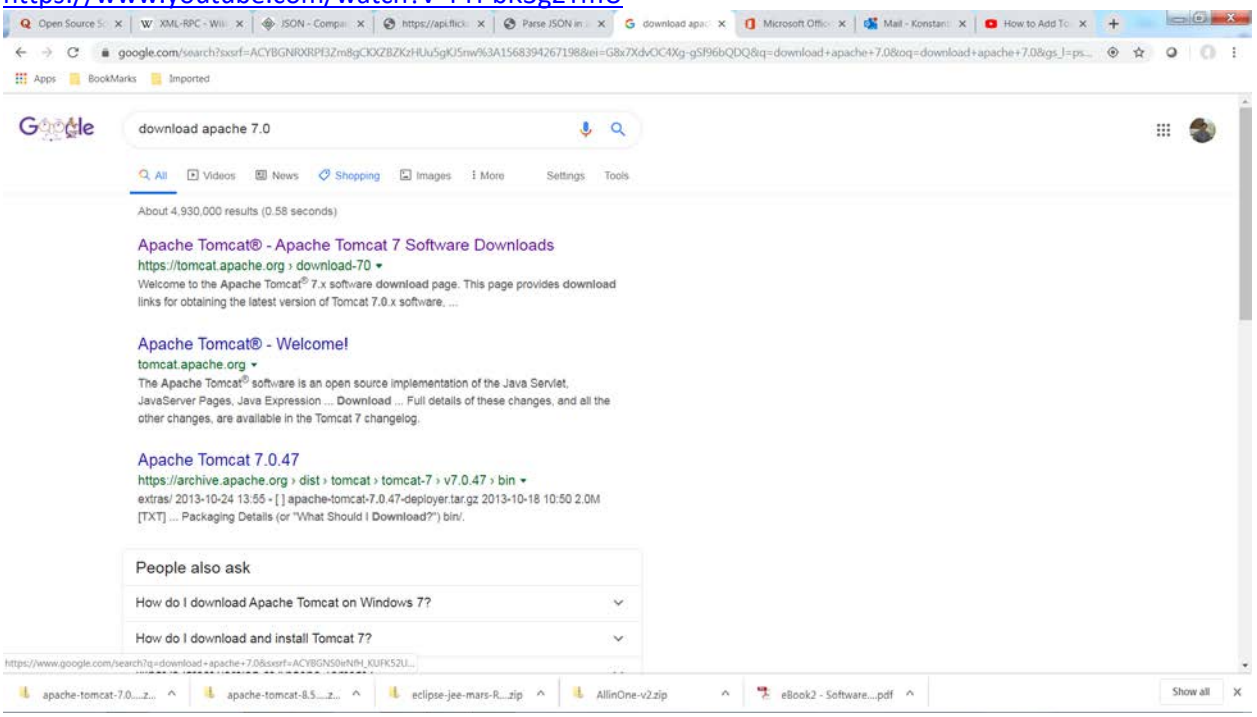


Step 4. Once you run eclipse select a workspace directory and close the Welcome page and. You can use the default one or cerate your own. In this example I created my own on C:\users\kostas\Eclipse-Workspace

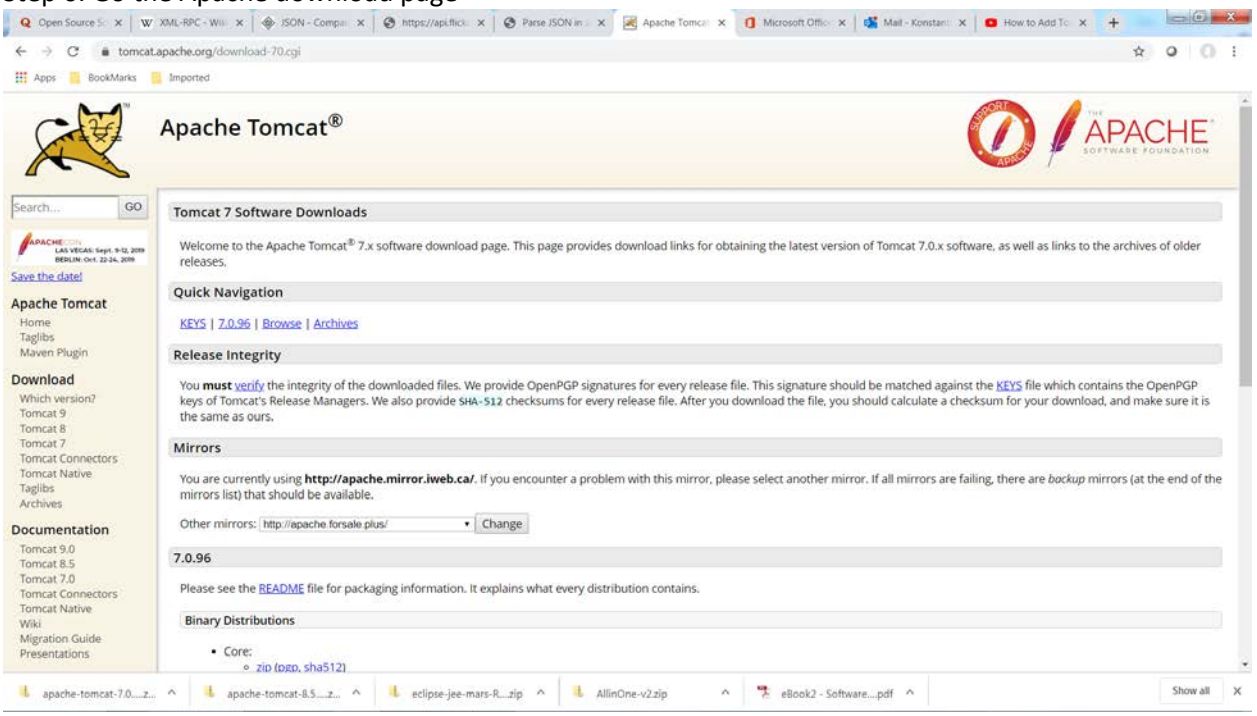




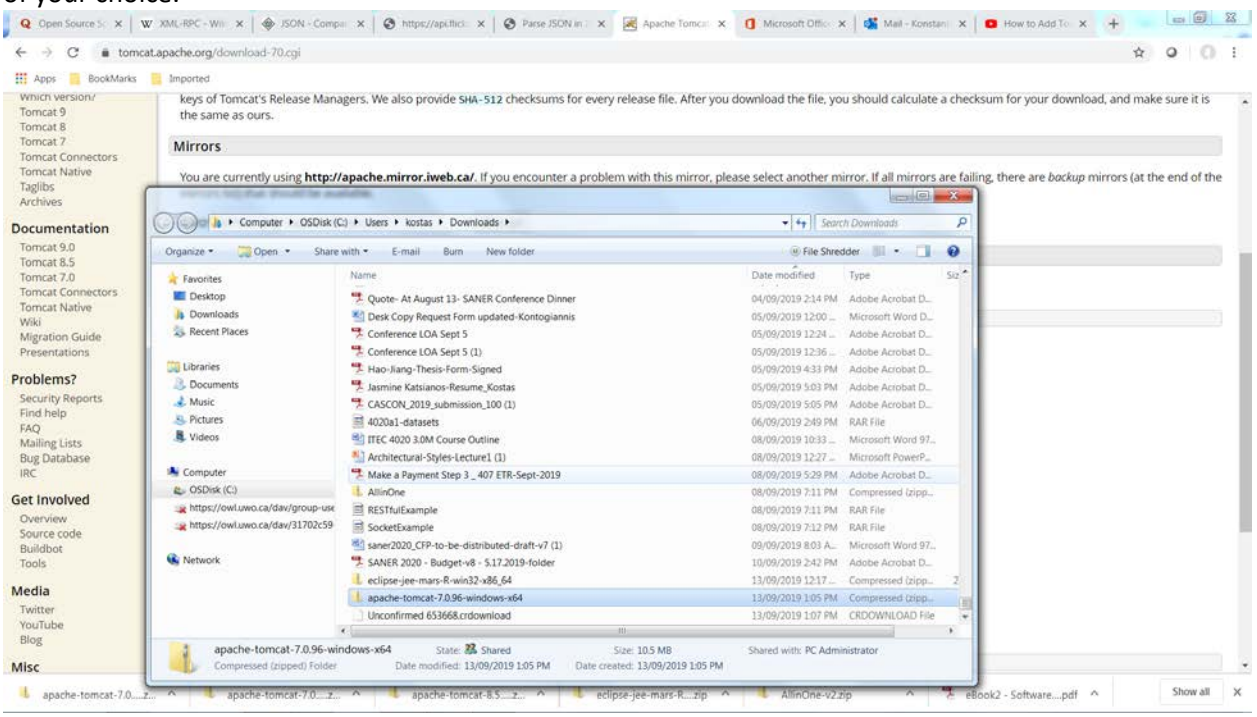
Step 5. Install Tomcat. Google for Apache Tomcat 7.0. An instructive video is at https://www.youtube.com/watch?v=PH-bK3g2YmU



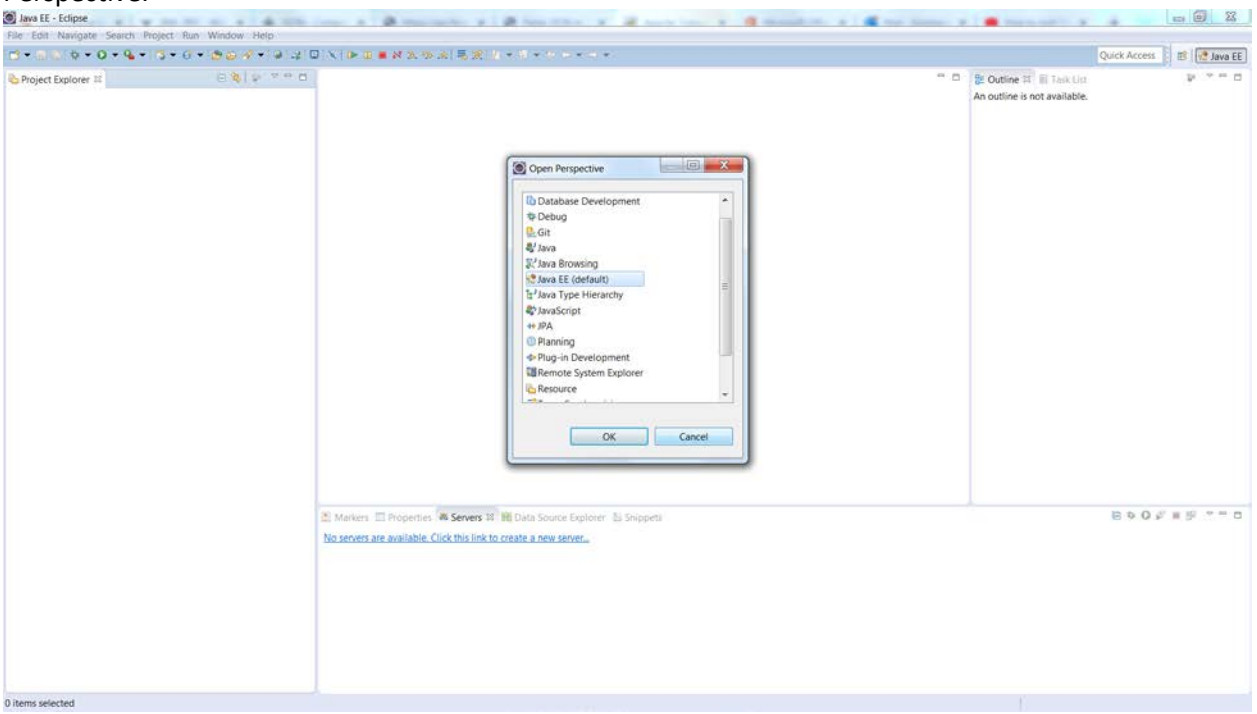
Step 6. Go the Apache download page



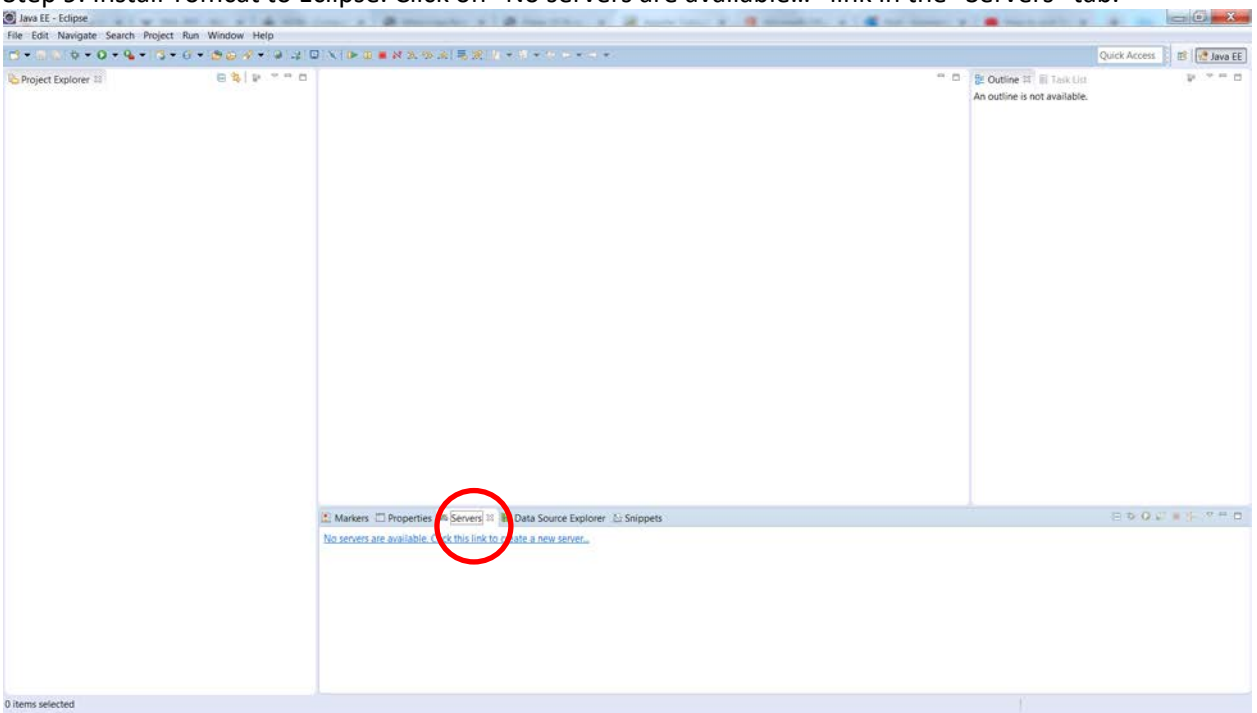
Step 7. Download the binary version appropriate for your computer. Extract the downloaded archive to a directory of your choice.



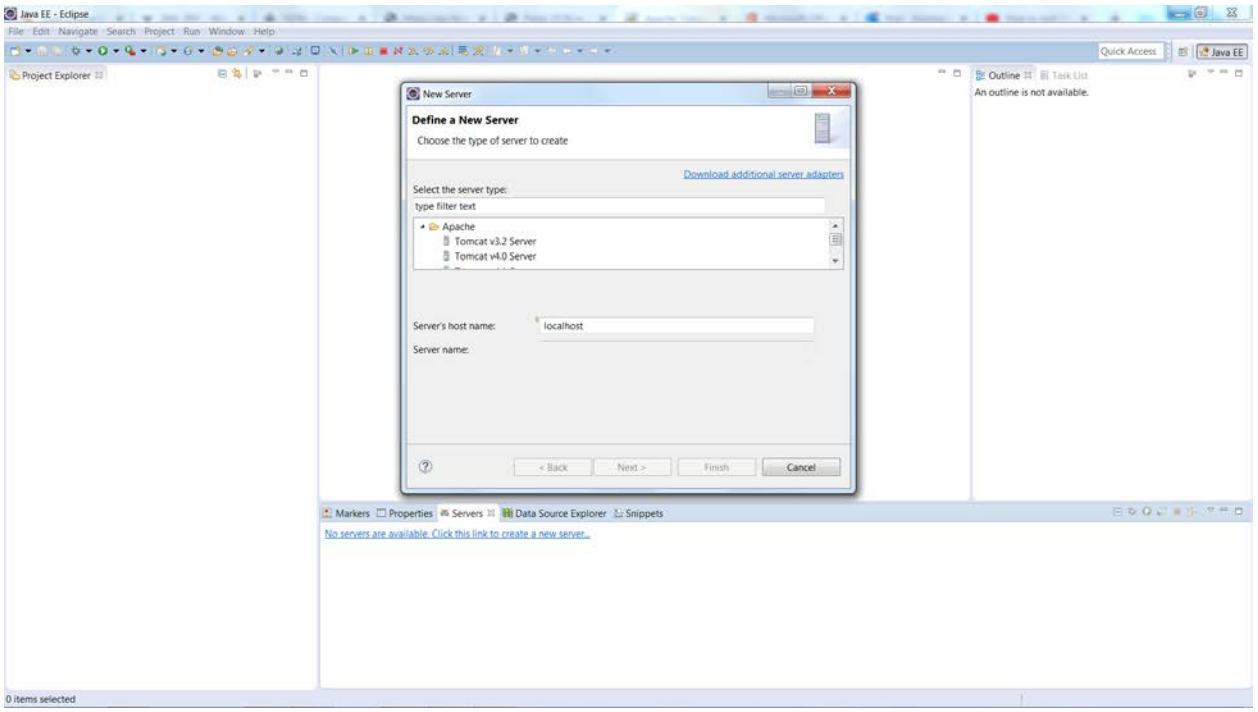
Step 8. Go to Eclipse. Select the Java EE perspective (see top right), or from Windows → Perspective → Open Perspective.

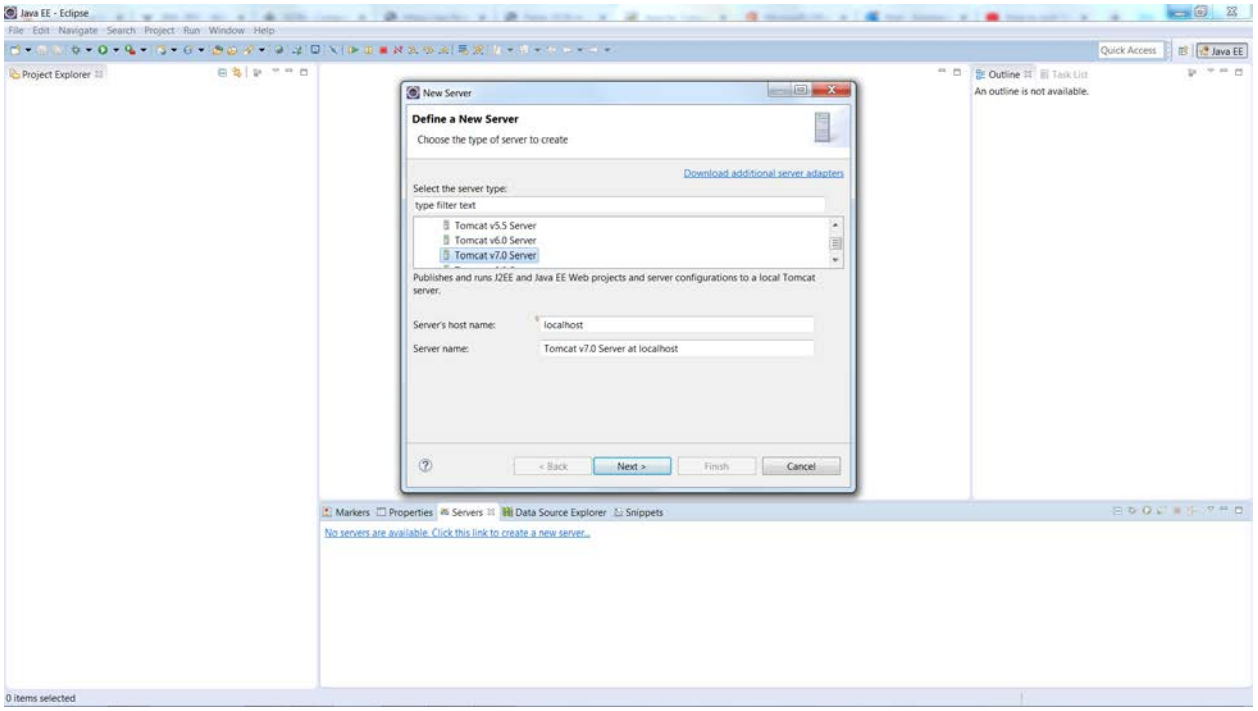


Step 9. Install Tomcat to Eclipse. Click on “No servers are available…” link in the “Servers” tab.

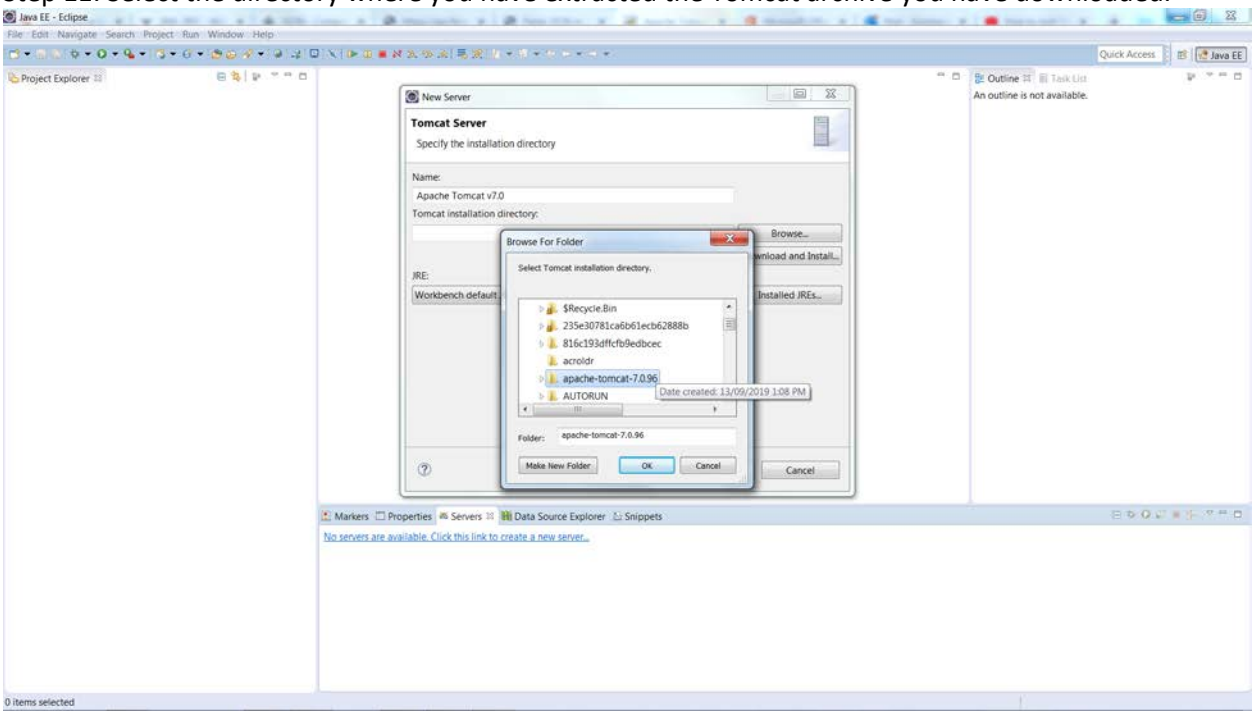


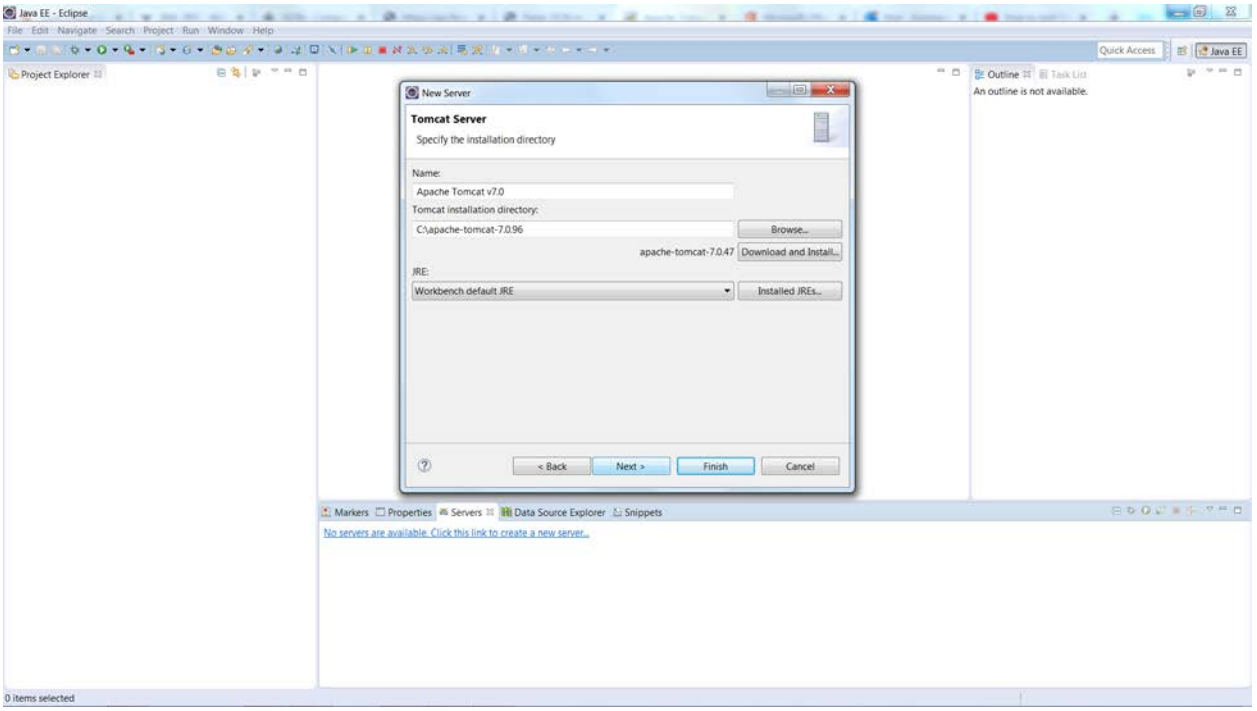
Step 10. Select the Apache Tomcat 7.0 server

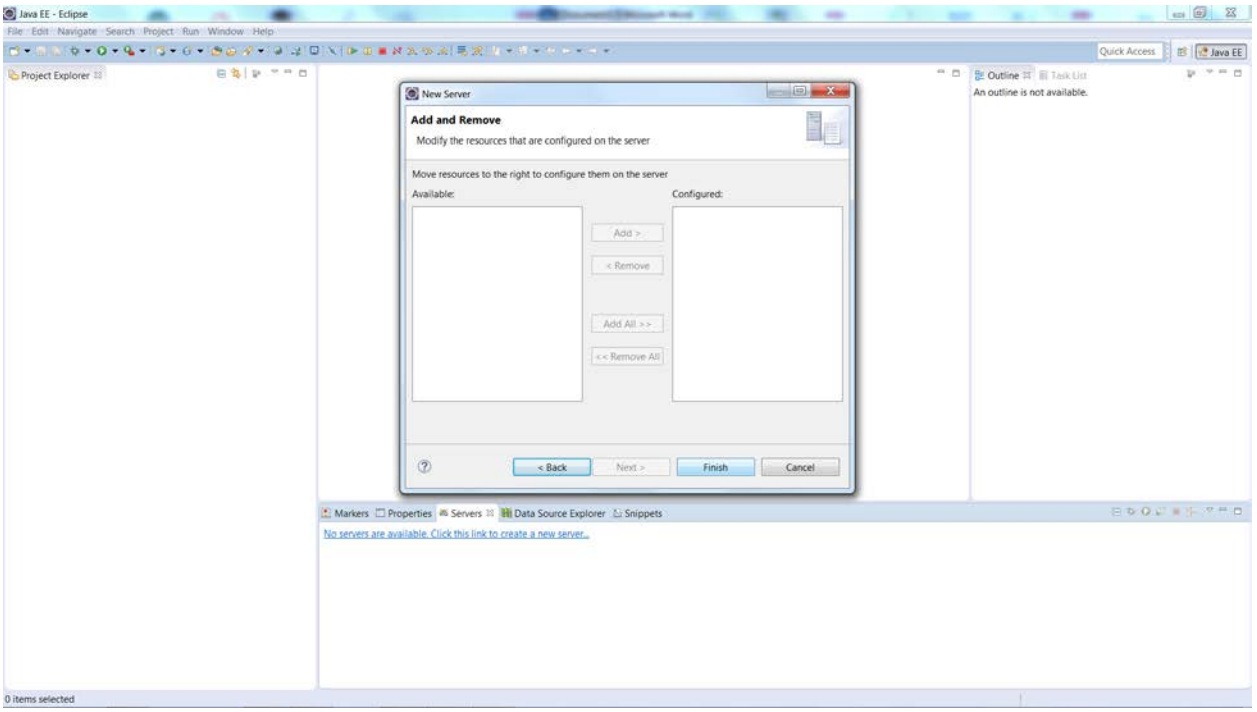




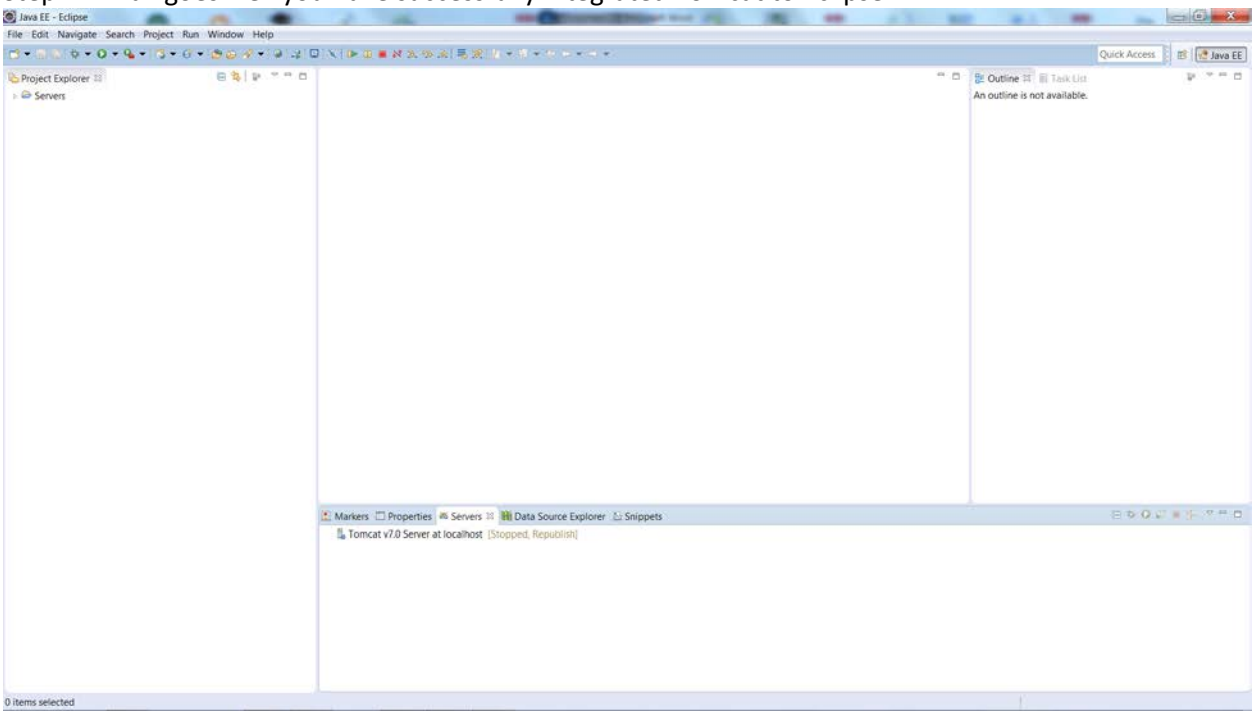
Step 11. Select the directory where you have extracted the Tomcat archive you have downloaded.







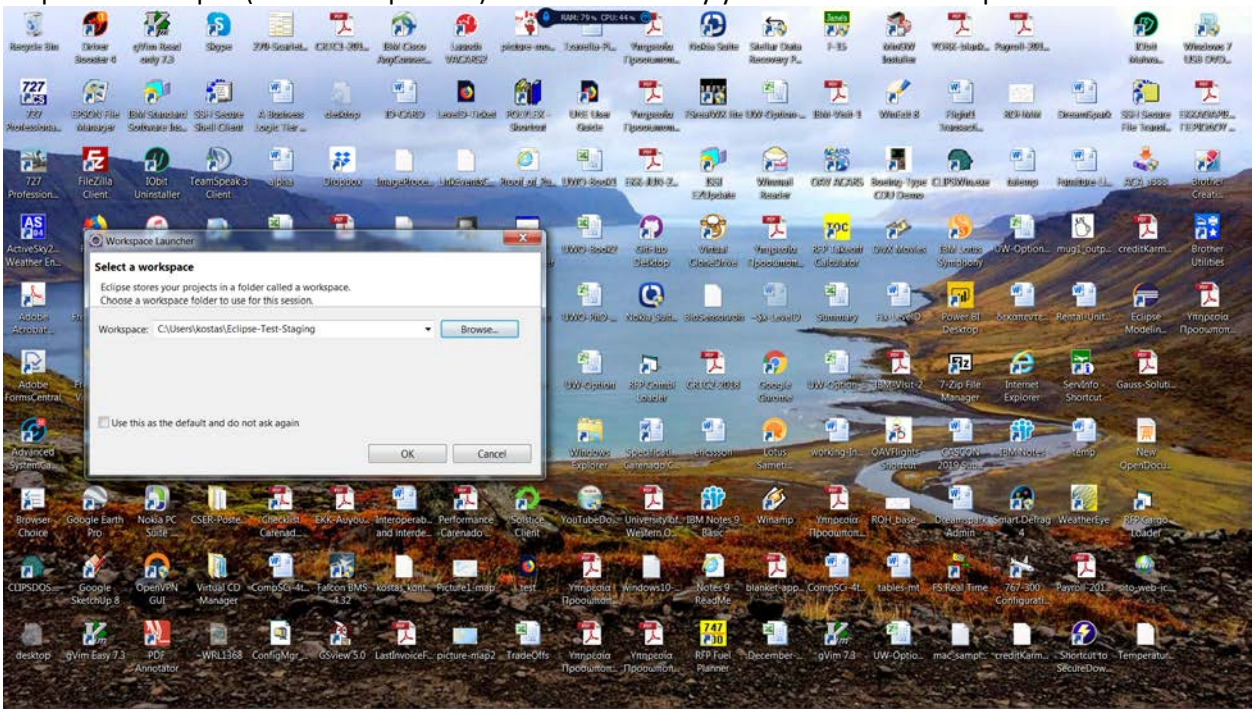
Step 12. If all goes well you have successfully integrated Tomcat to Eclipse.



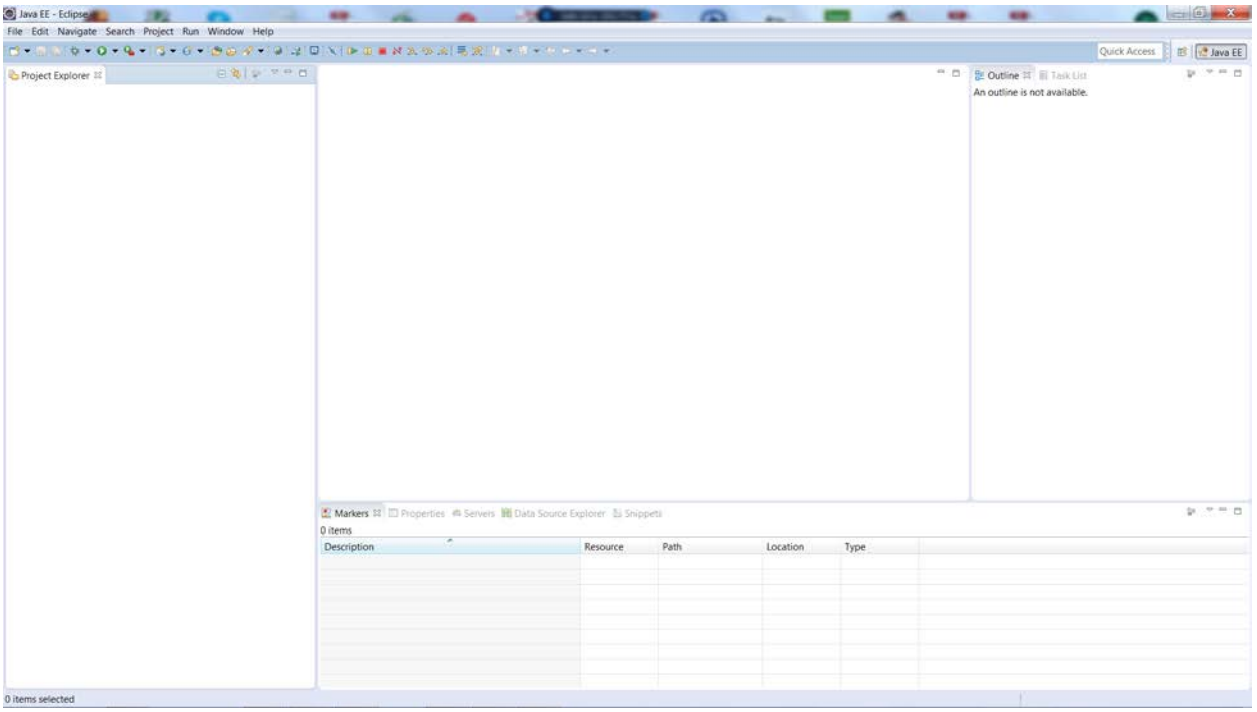
APPENDIX II

Start Eclipse and install the samples

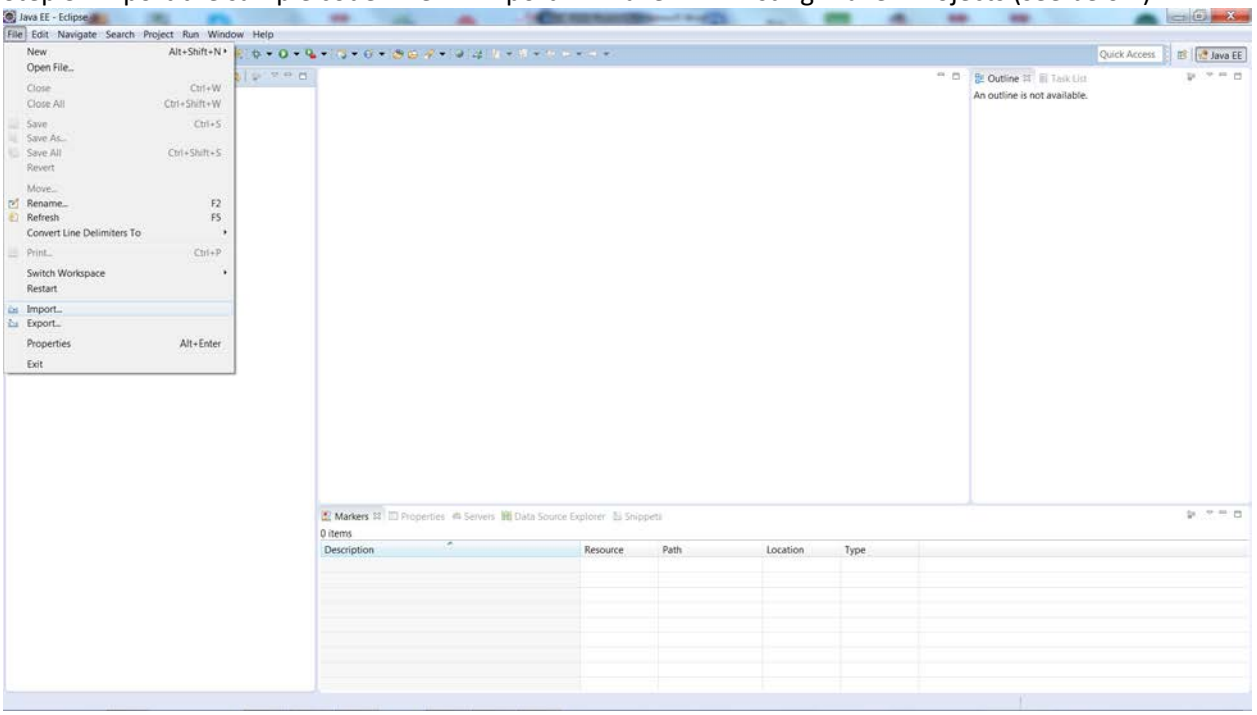
Step 1. Start Eclipse (run the eclipse.exe) from the directory you have installed Eclipse

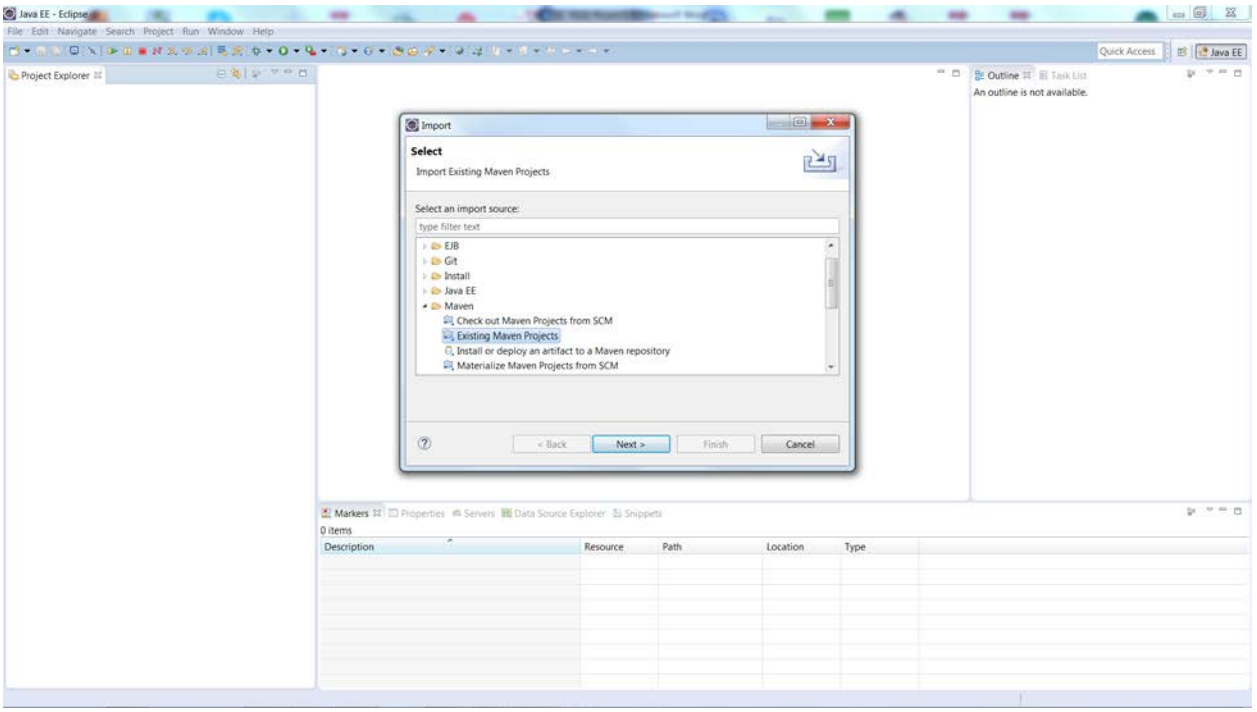


Step 2. Close the welcome page

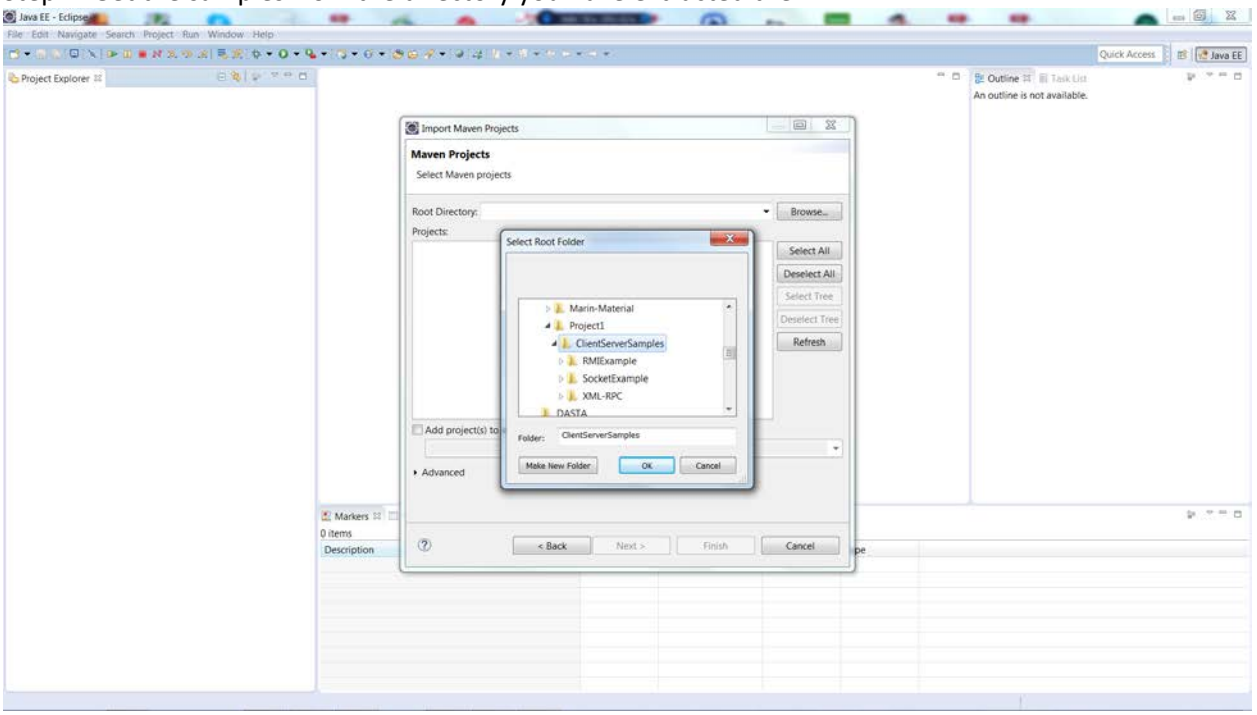


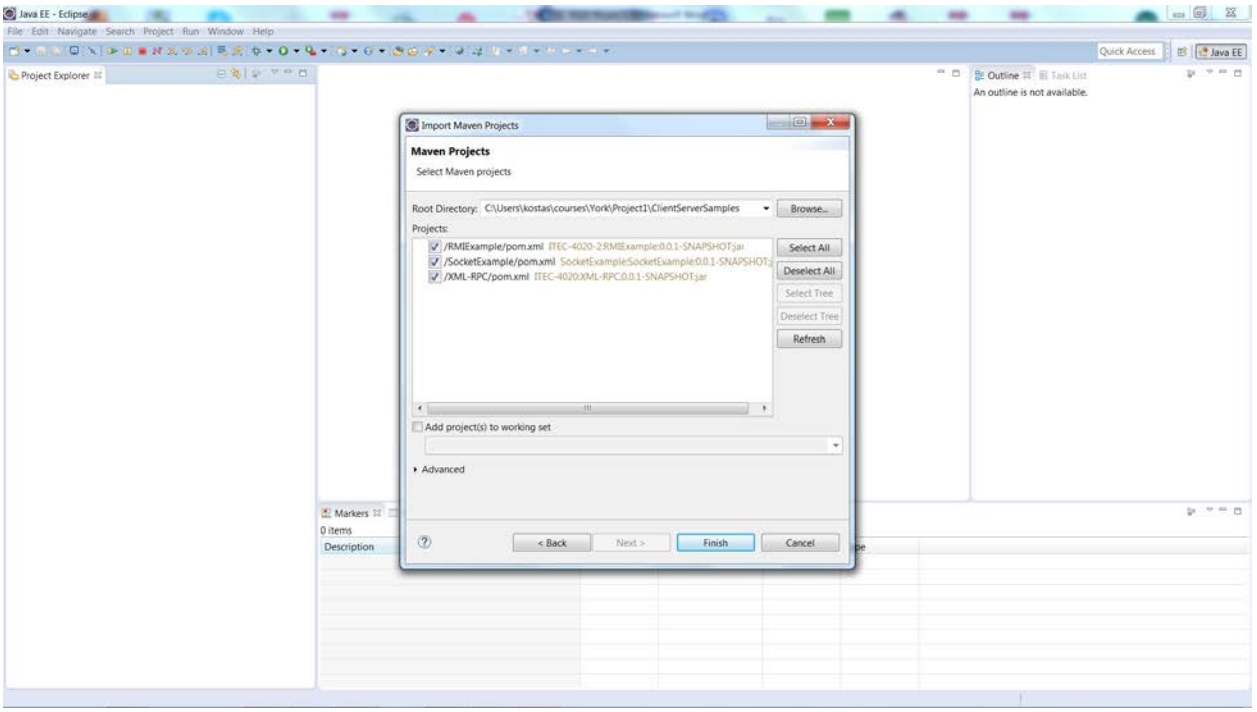
Step 3. Import the sample code. File → Import → Maven → Existing Maven Projects (see below).



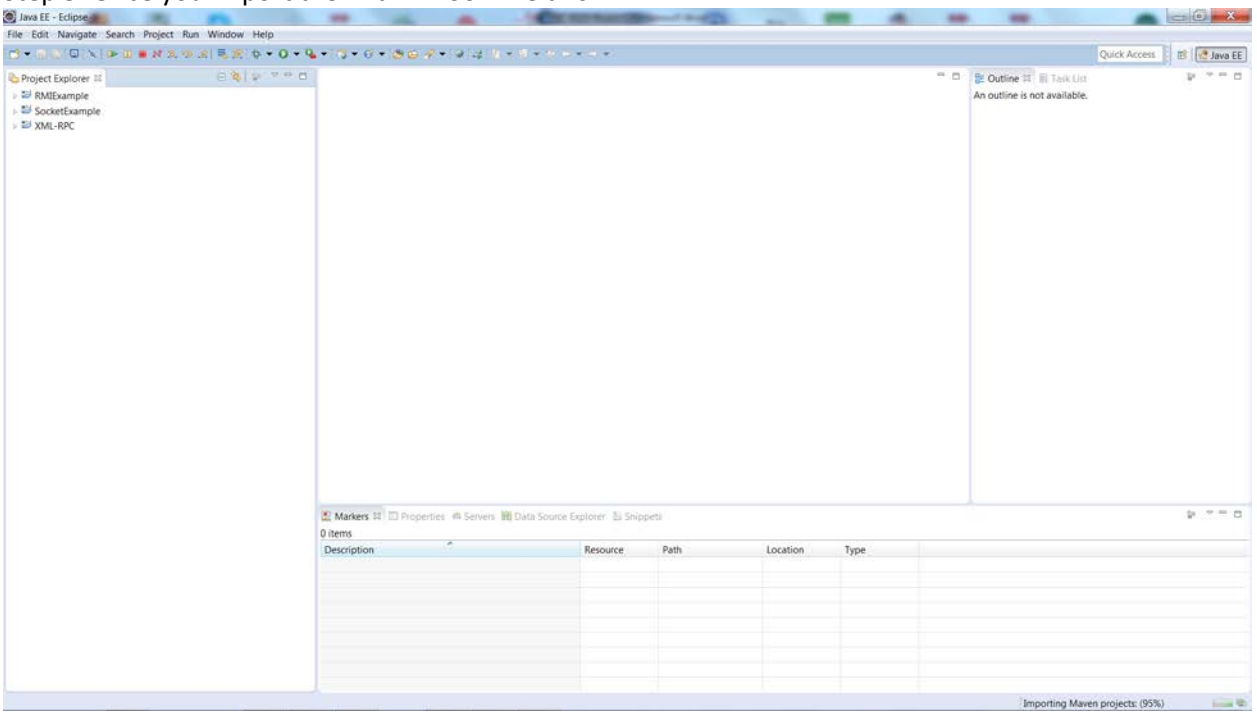


Step 4. Get the samples from the directory you have extracted them



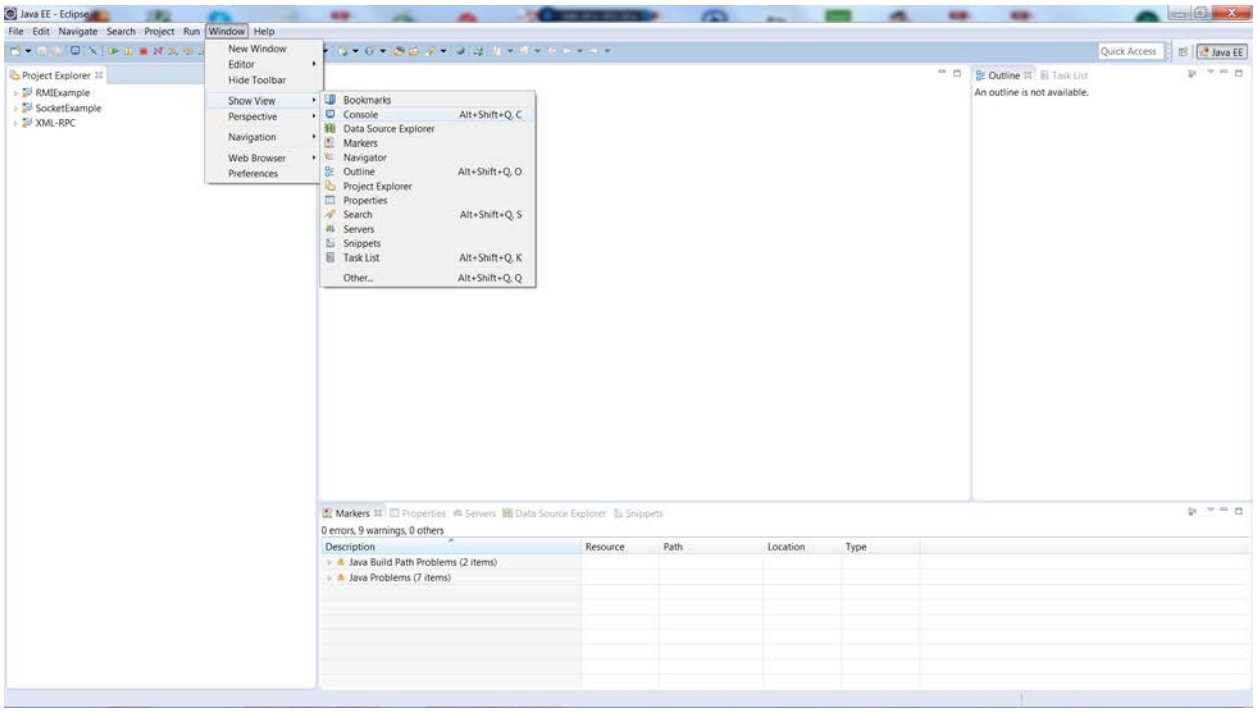


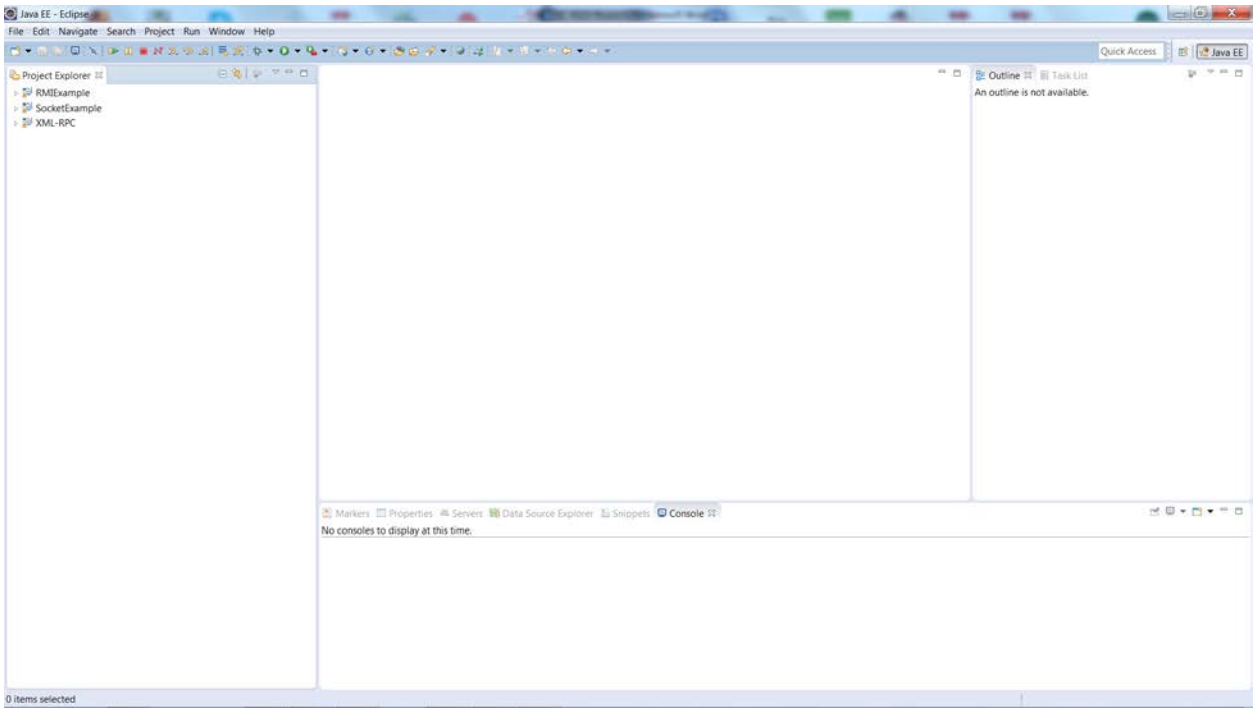
Step 5. Once you import them it will look like this.



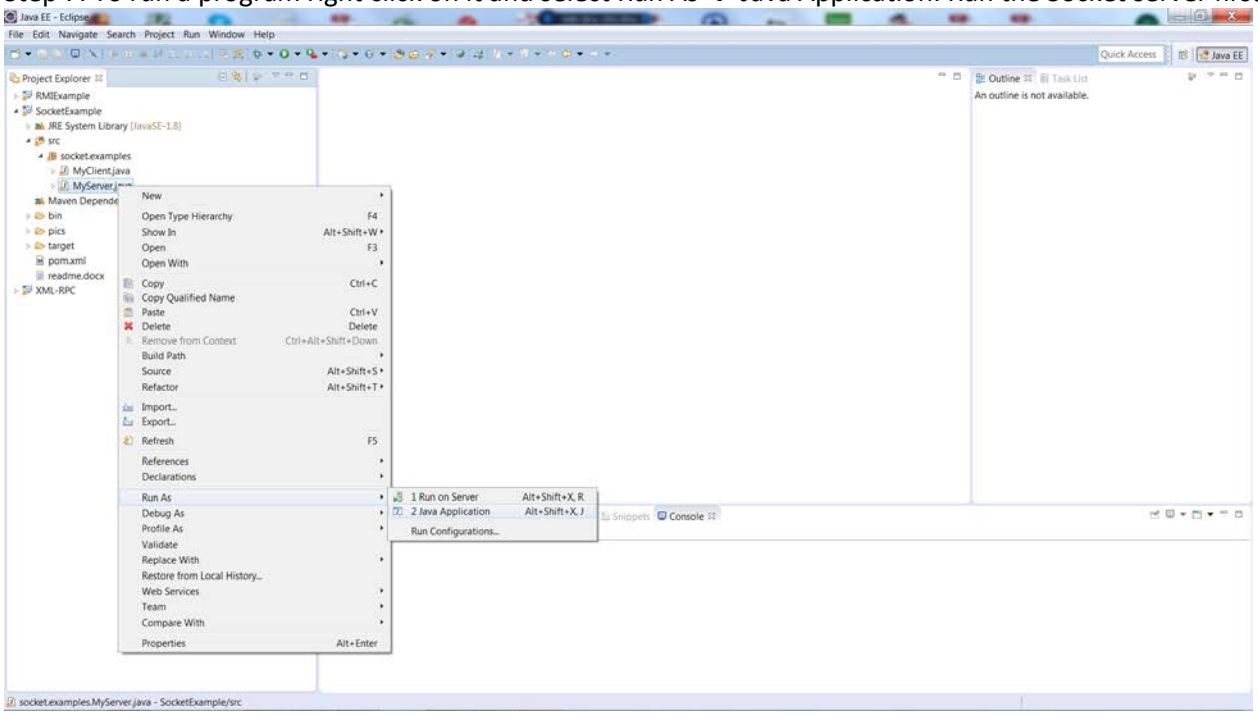
Step 6. Run the samples

A. SOCKETS EXAMPLE

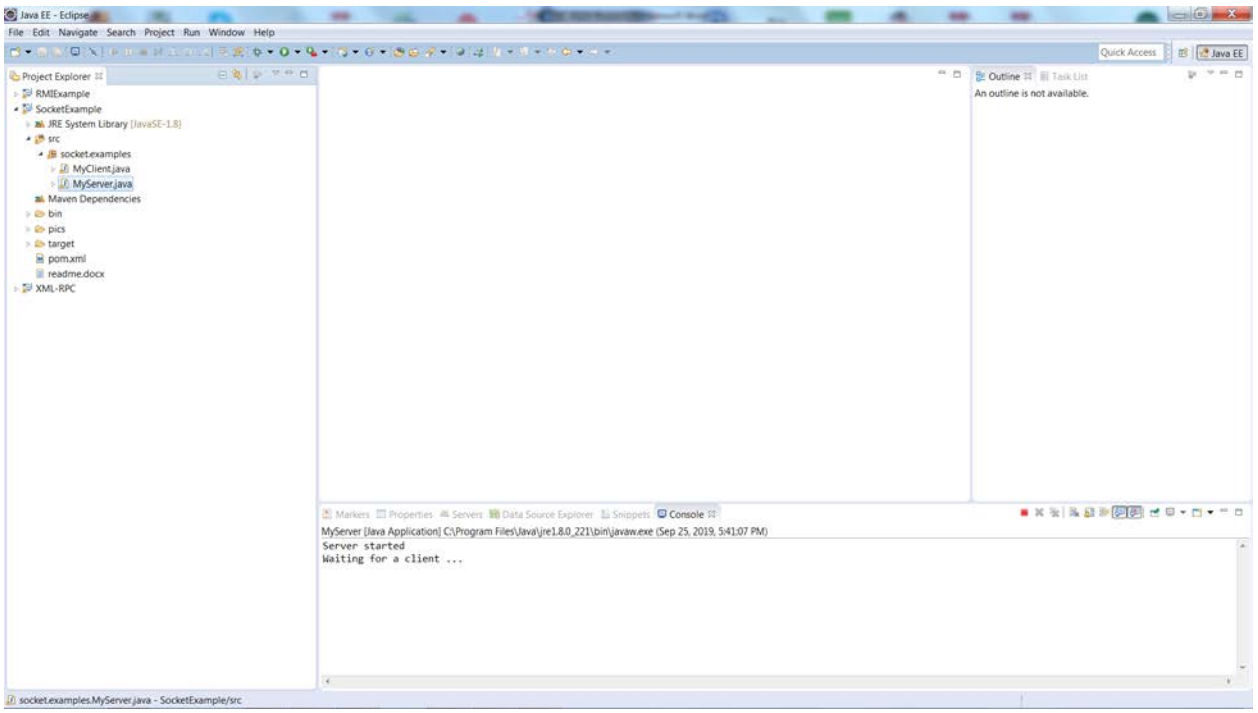




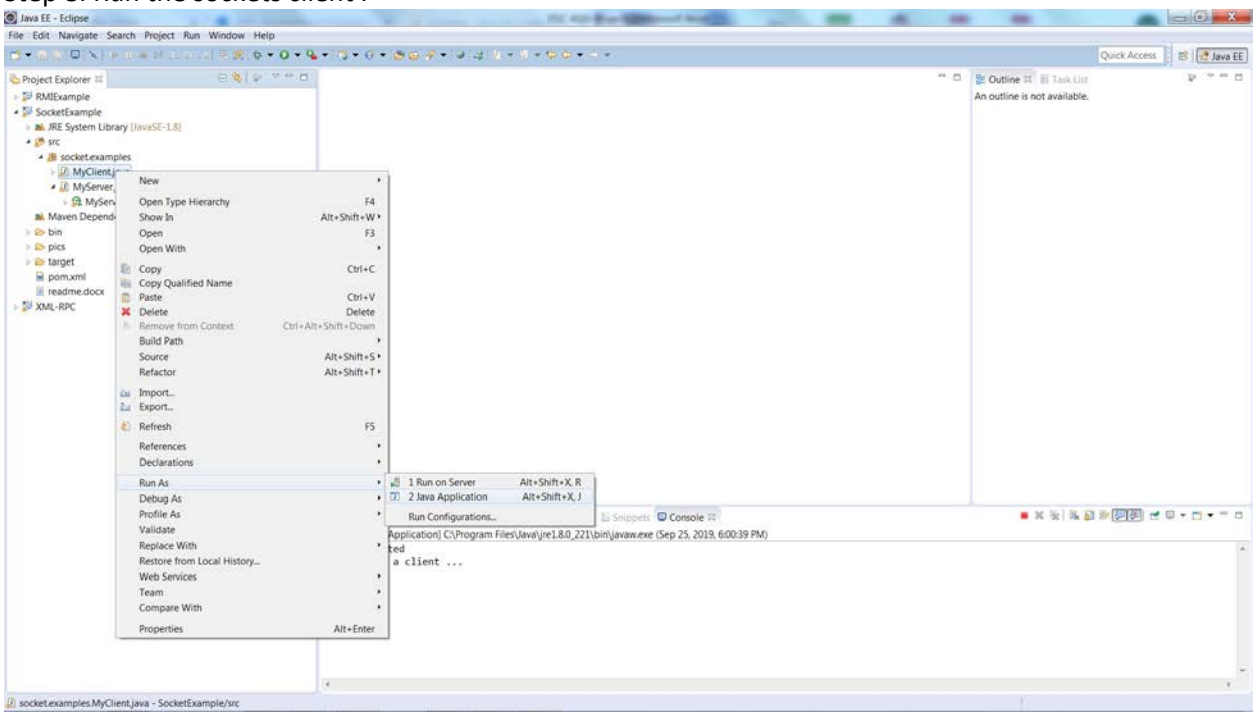
Step 7. To run a program right click on it and select Run As → Java Application. Run the Socket server first.



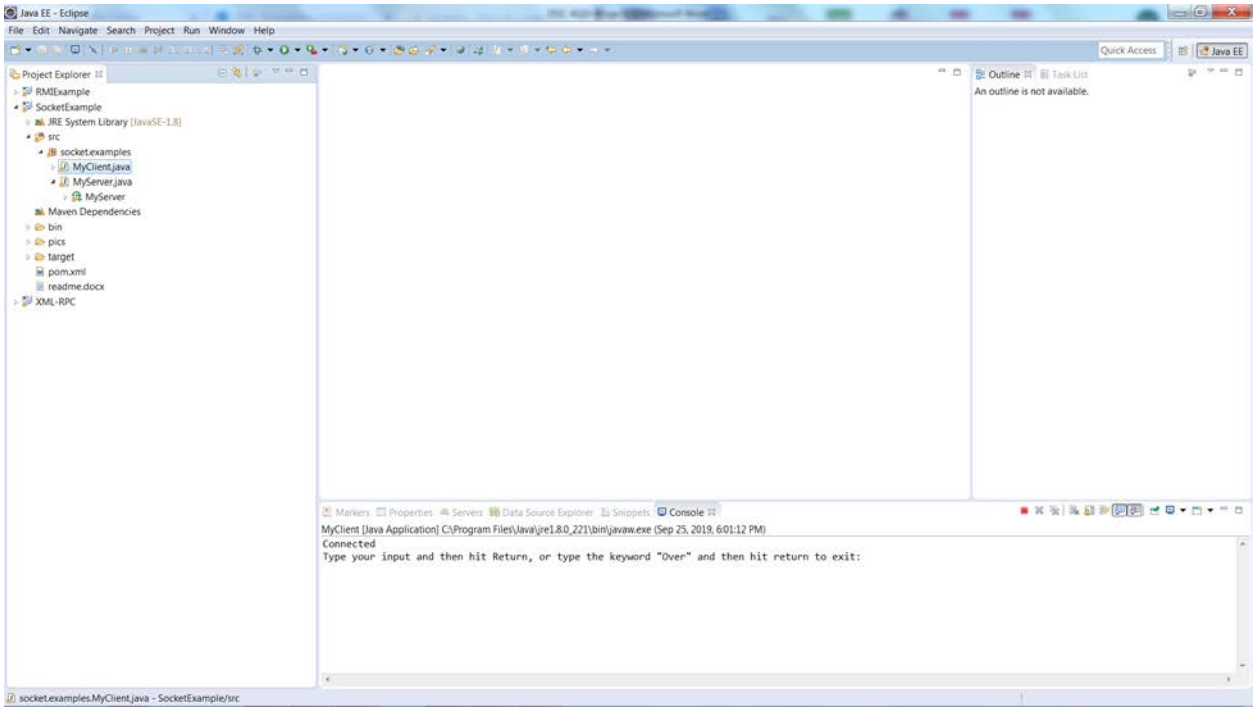
It look like this.



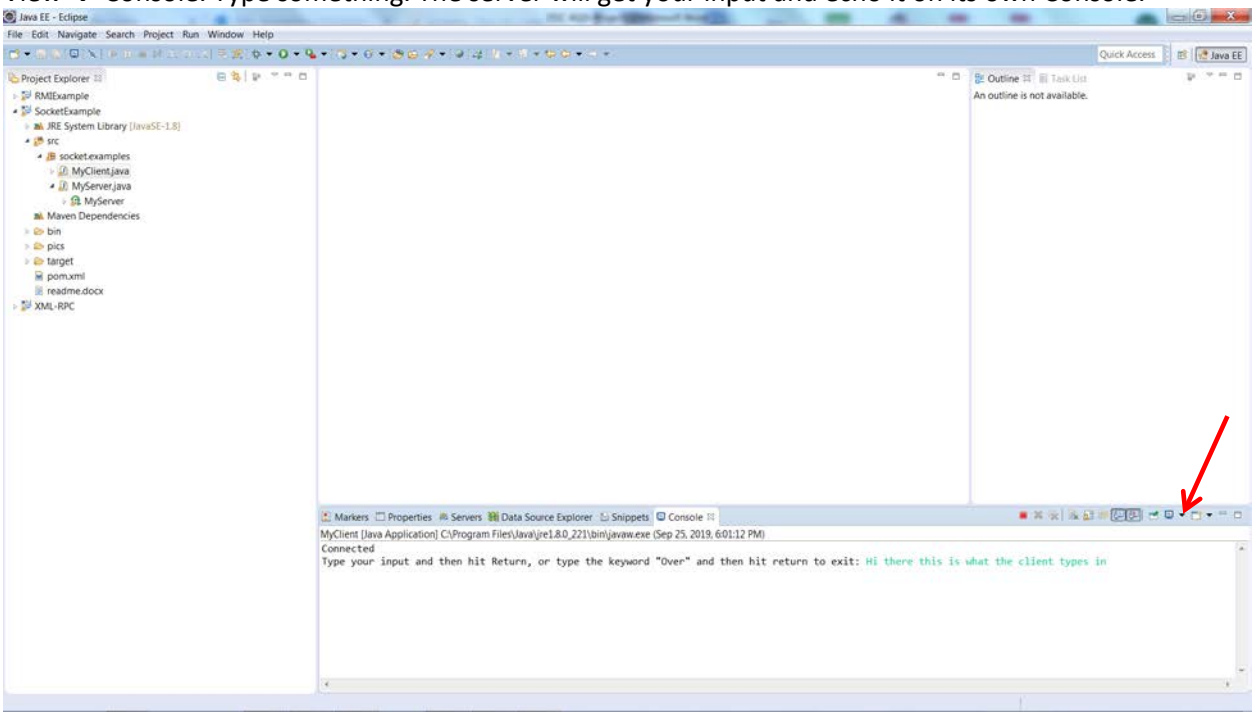
Step 8. Run the sockets client .

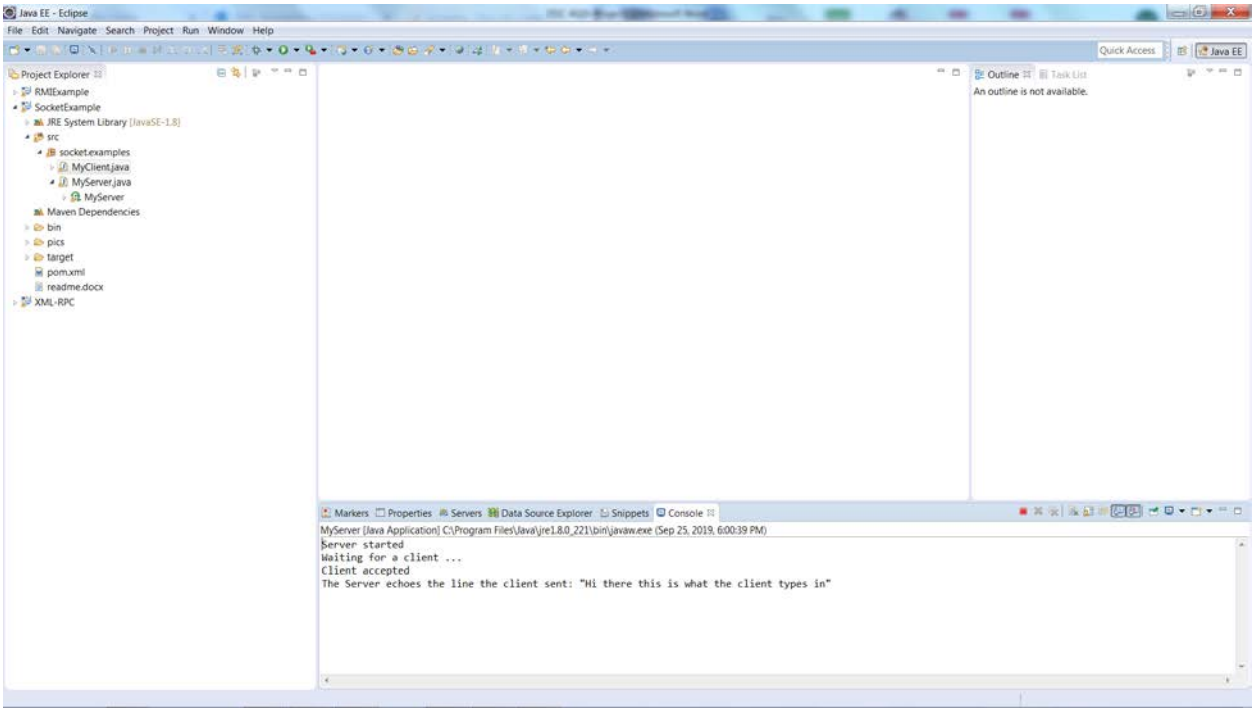


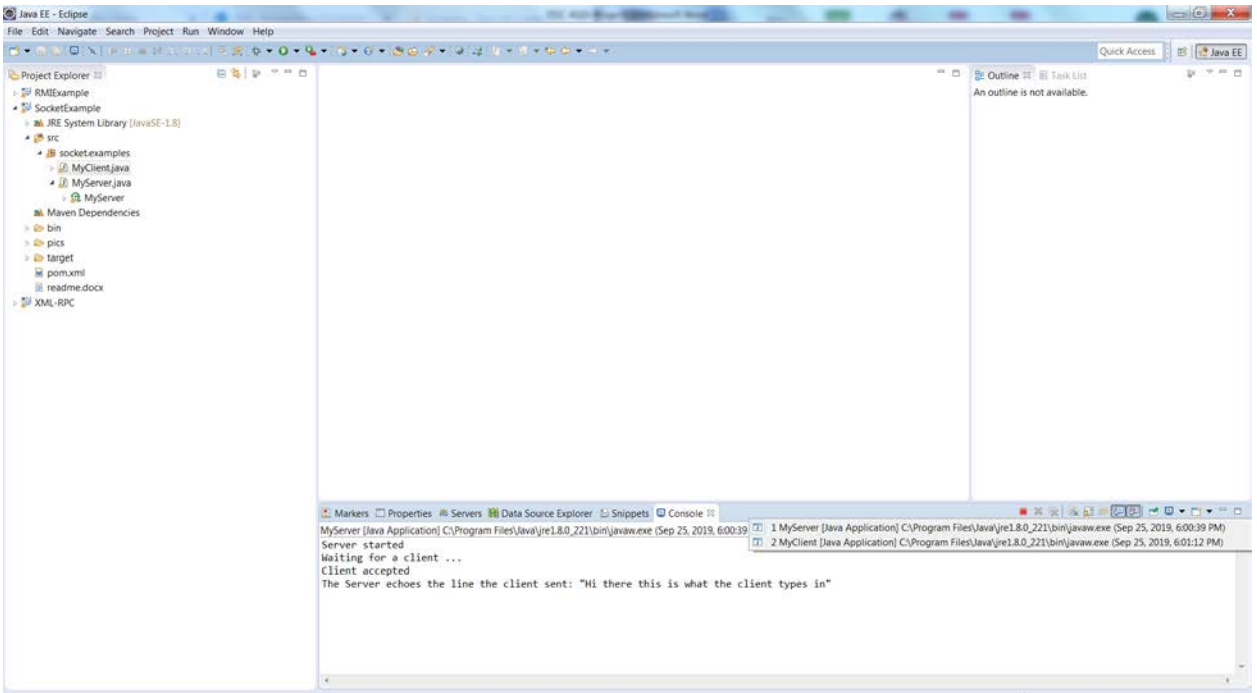
Looks like this.

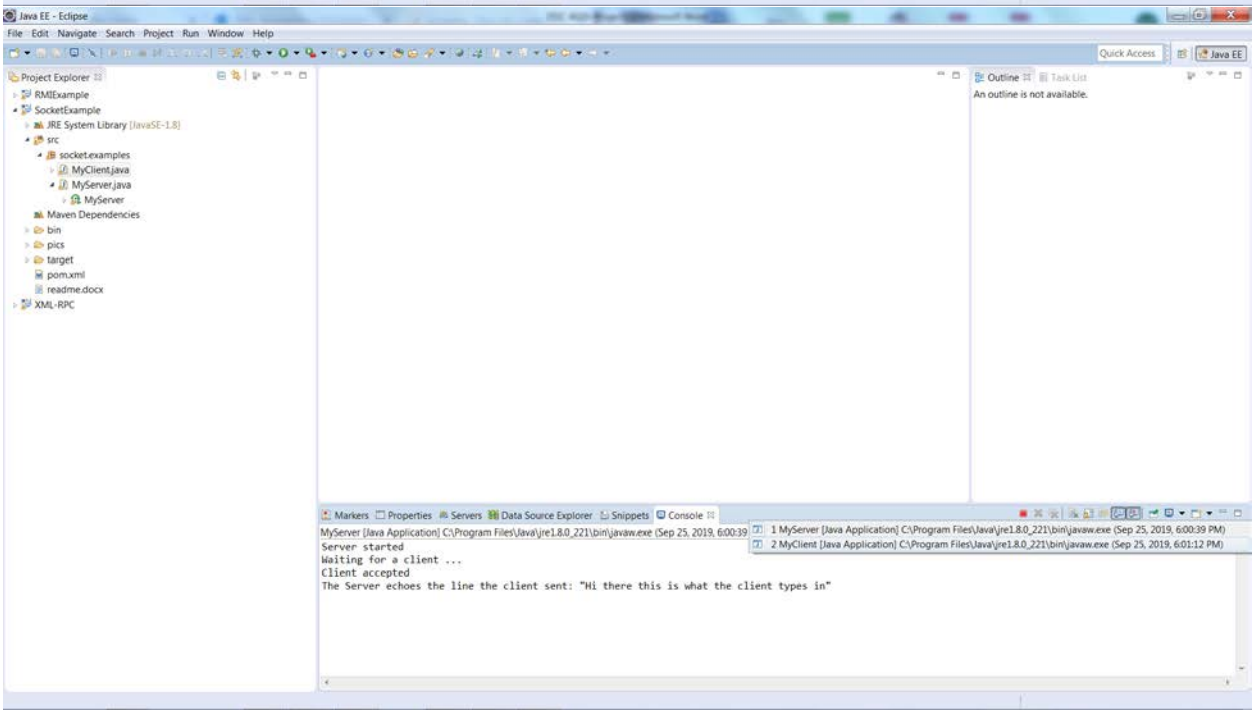


Step 9. Toggle to the client Console (use the drop down menu – see arrow). To have a Console use Window → Show View → Console. Type something. The server will get your input and echo it on its own Console.

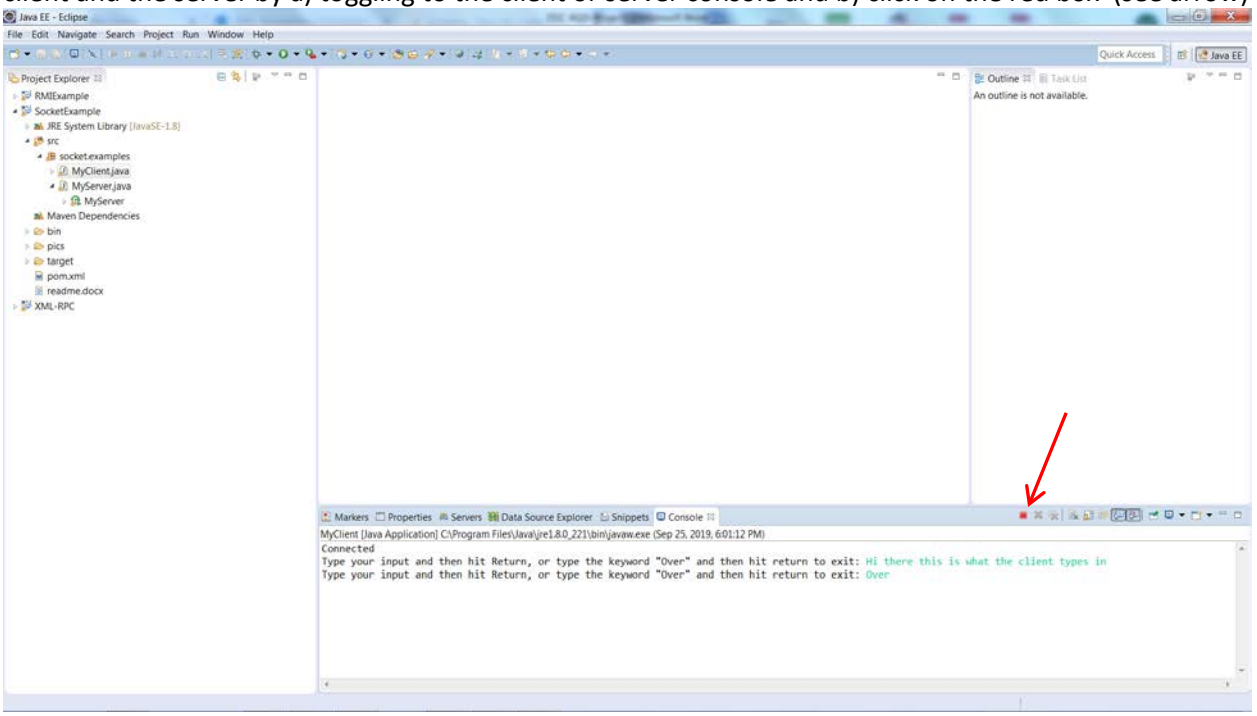


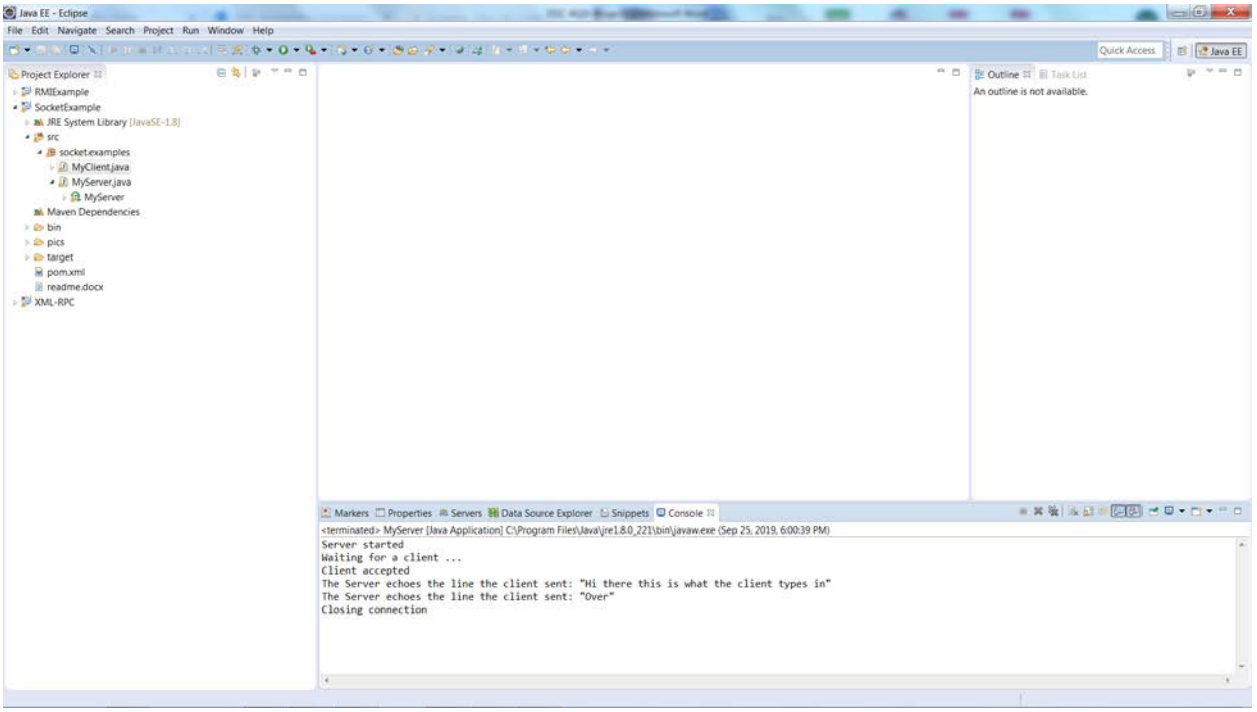



Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
SocketExample
JRE System Library [JavaSE-1.8]
src
socket.examples
MyClient.java
MyServer.java
MyServer
Maven Dependencies
bin
pics
target
pom.xml
readme.docx
XML-RPC
Outline
Task List
An outline is not available.
Markers Properties Servers Data Source Explorer Snippets Console
MyServer [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:00:39
1 MyServer [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:00:39 PM)
2 MyClient [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:01:12 PM)
Server started
Waiting for a client ...
Client accepted
The Server echoes the line the client sent: "Hi there this is what the client types in"


Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
SocketExample
JRE System Library [JavaSE-1.8]
src
socket.examples
MyClient.java
MyServer.java
MyServer
Maven Dependencies
bin
pics
target
pom.xml
readme.docx
XML-RPC
Outline
Task List
An outline is not available.
Markers Properties Servers Data Source Explorer Snippets Console
MyServer [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:00:39
1 MyServer [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:00:39 PM)
2 MyClient [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:01:12 PM)
Server started
Waiting for a client ...
Client accepted
The Server echoes the line the client sent: "Hi there this is what the client types in"

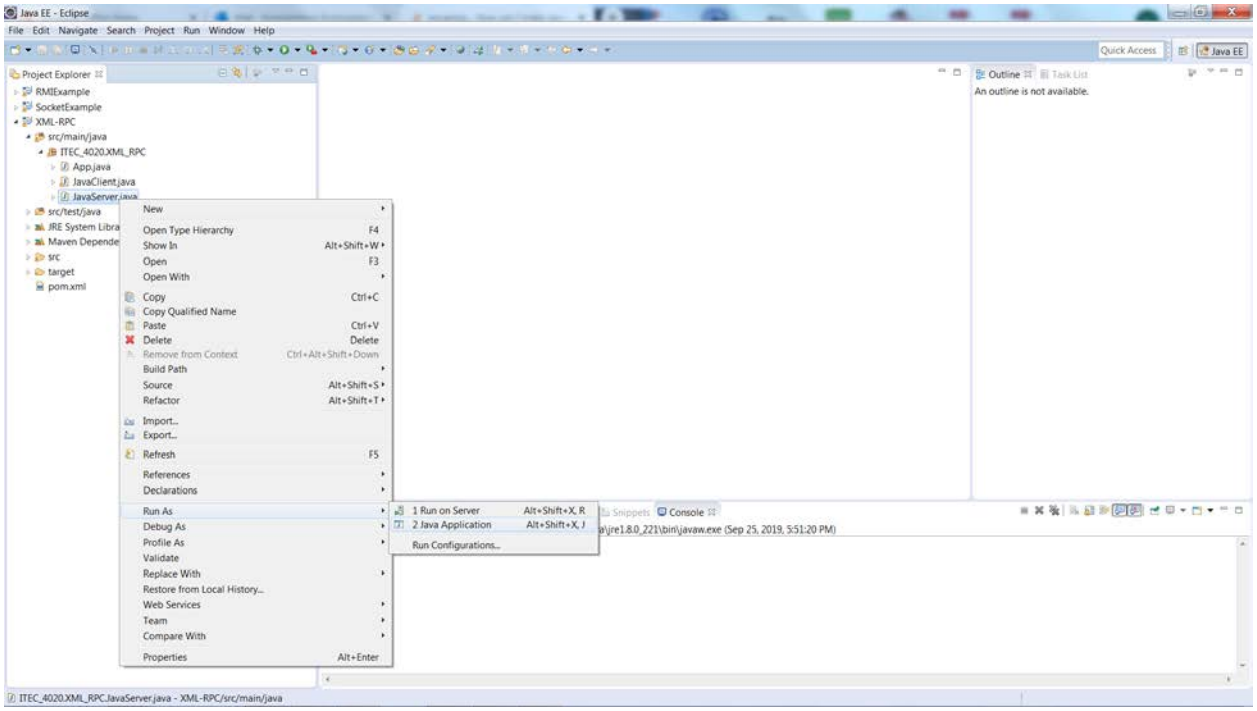
Step 10. Close the connection. You can type over on the client. In the other examples you will have to terminate the client and the server by a) toggling to the client or server console and b) click on the red box (see arrow)

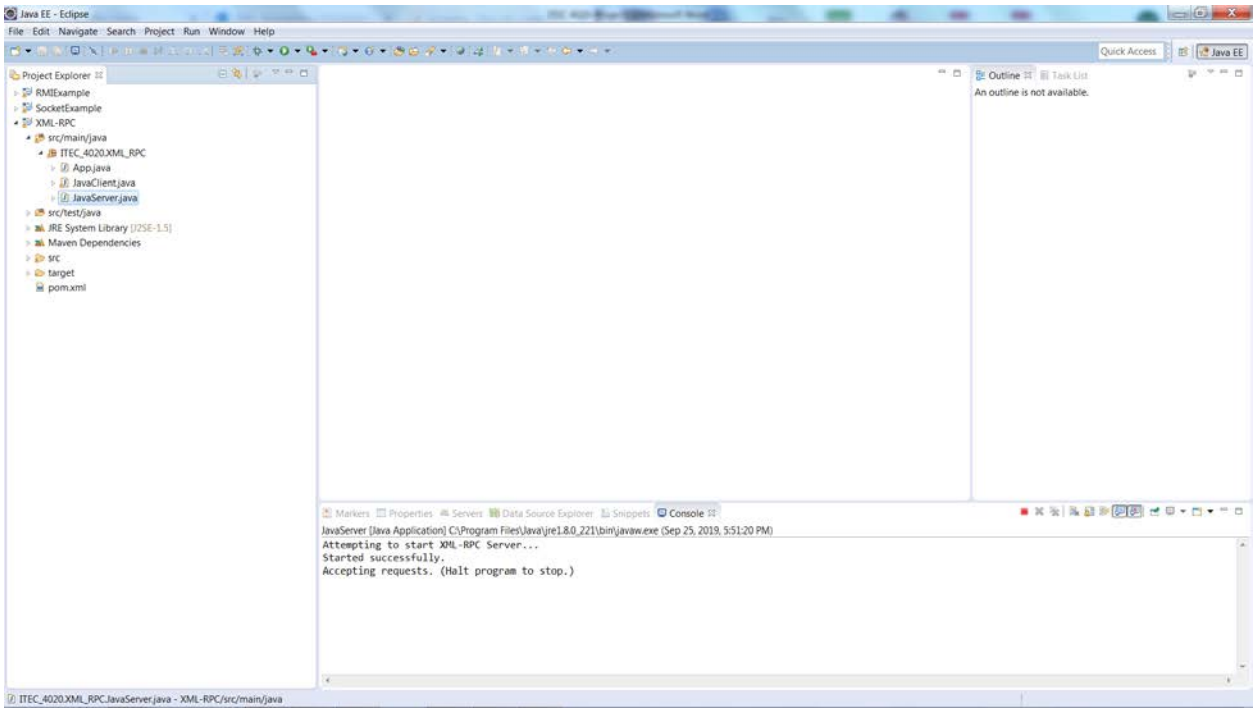


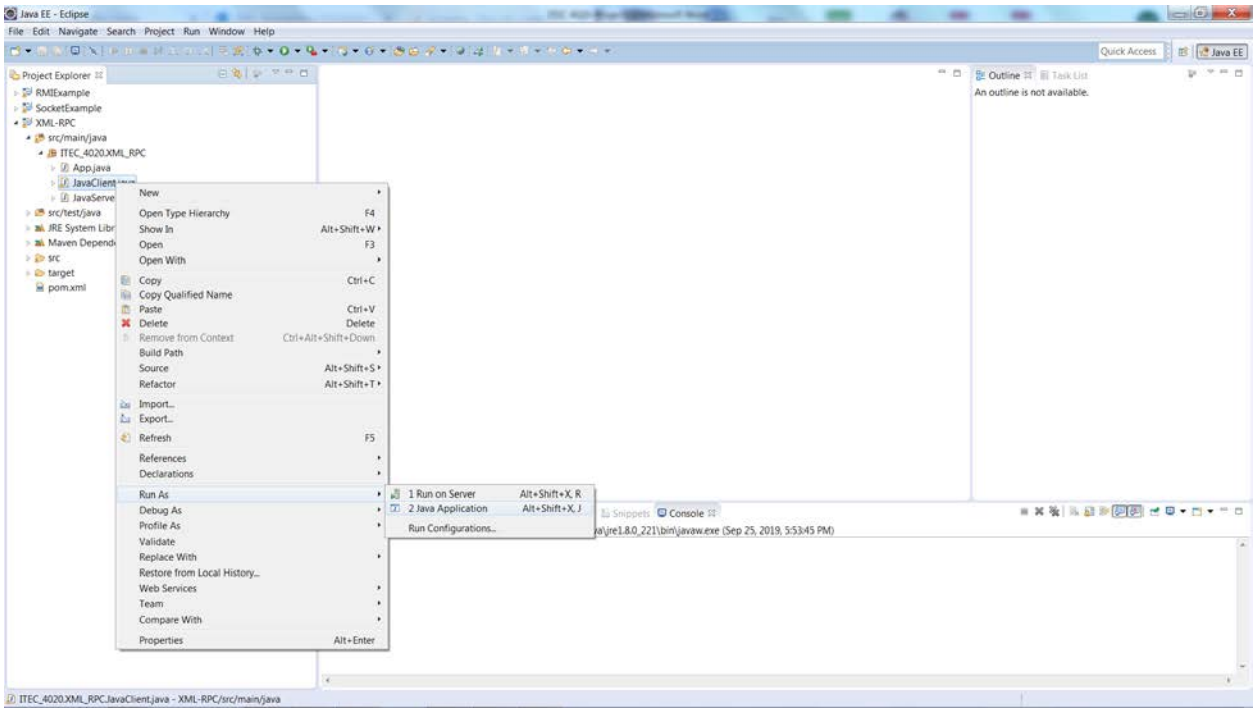


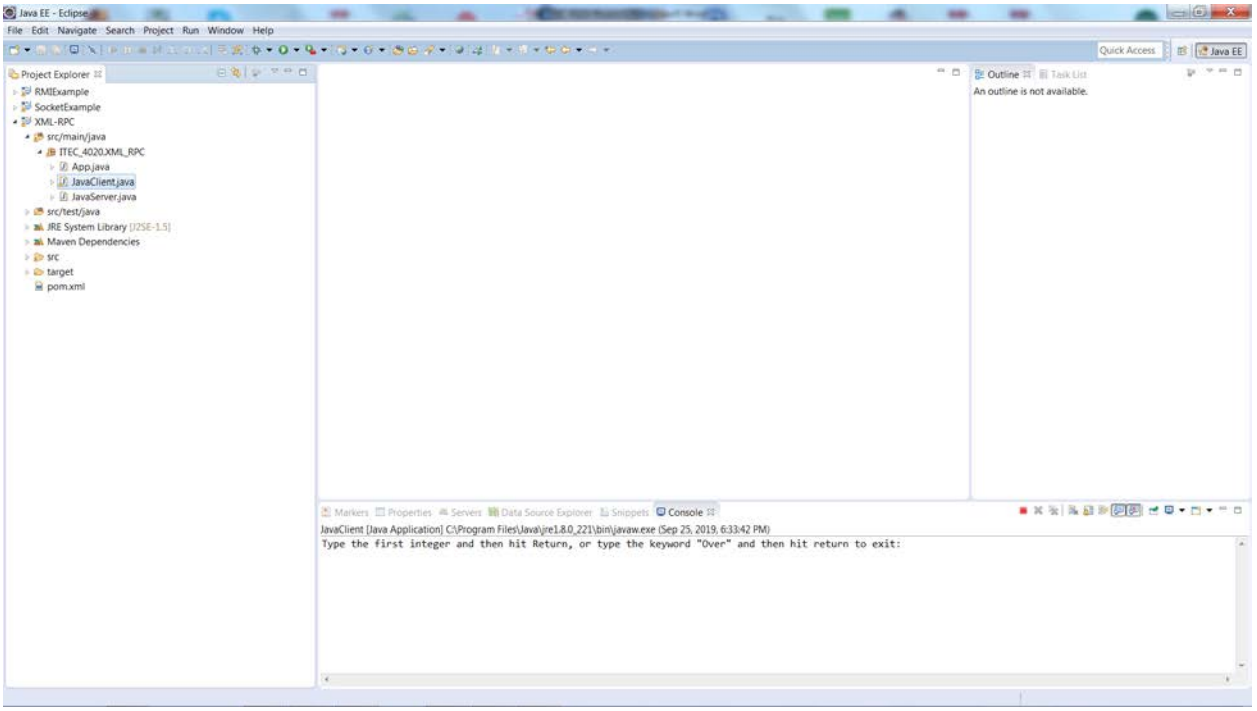
Do not forget to stop the client and the server programs using the red box termination button.

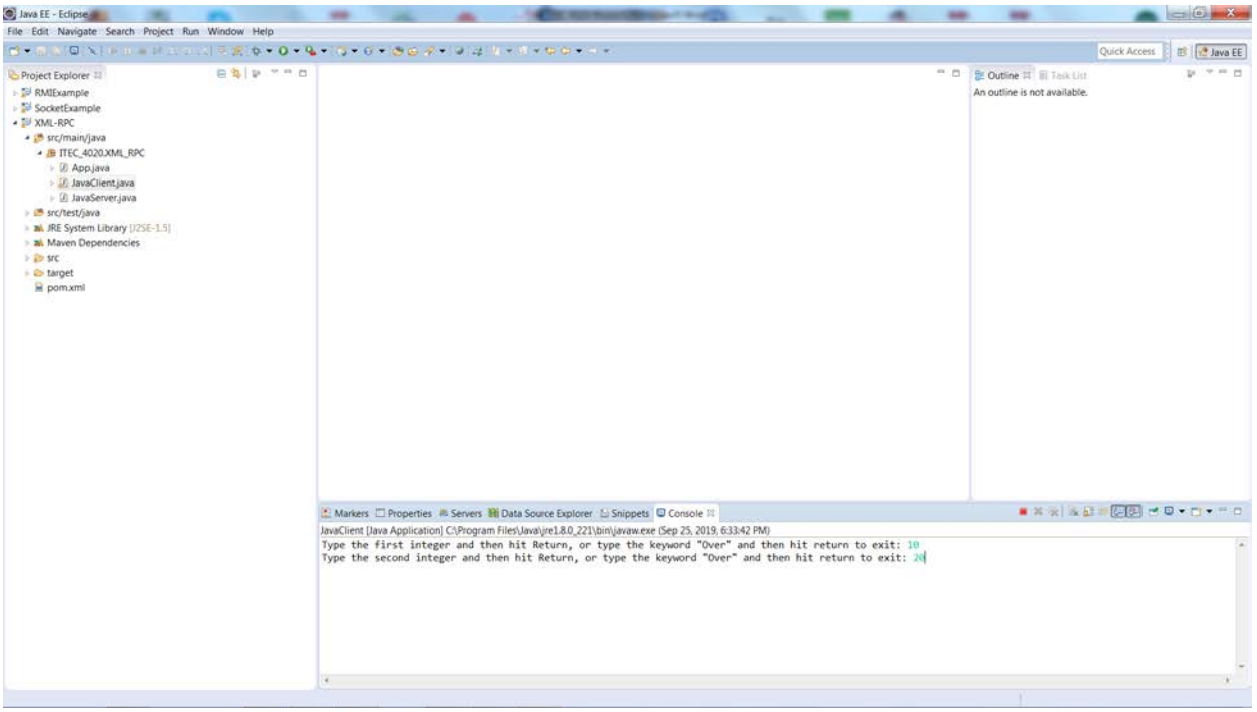
B. Running the XML RPC Example

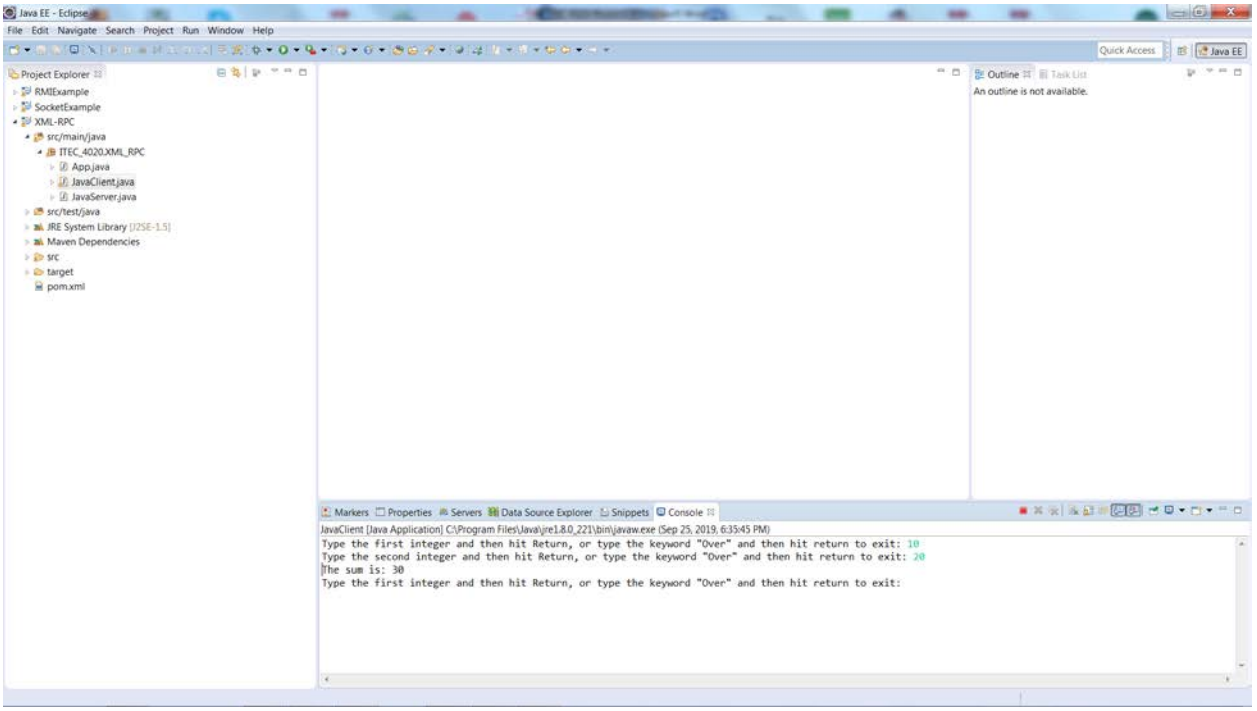


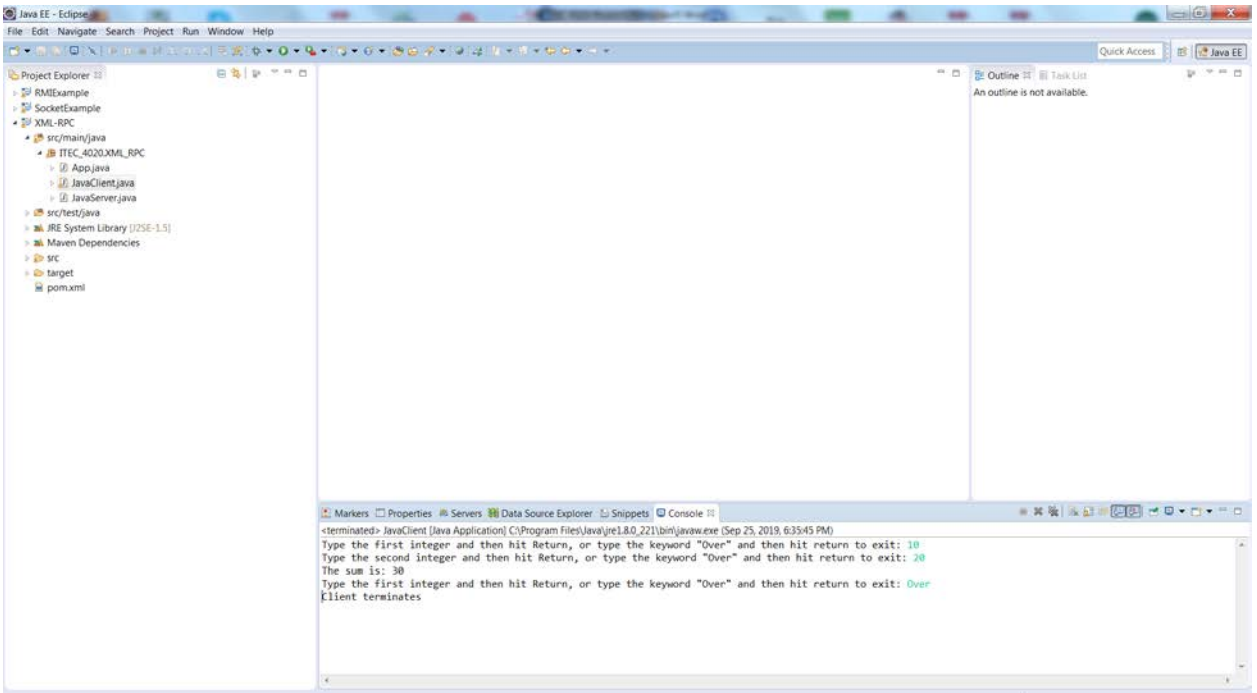


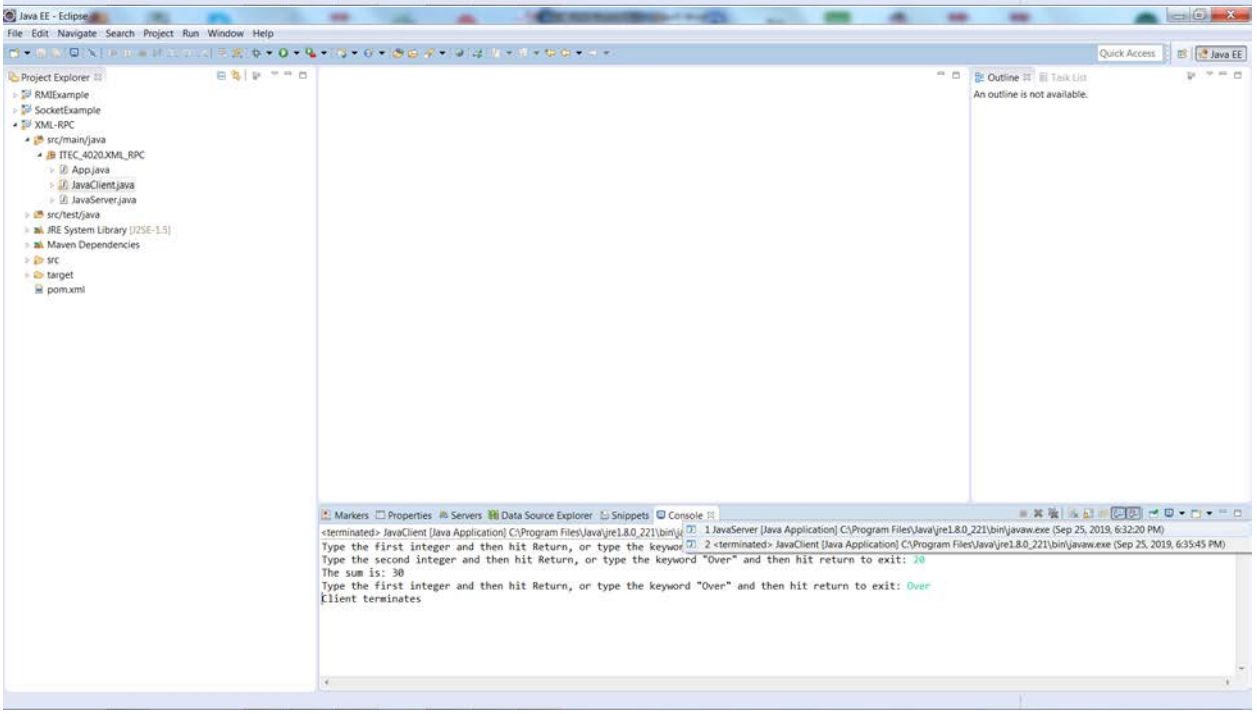

Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
SocketExample
XML-RPC
src/main/java
ITEC_4020.XML_RPC
App.java
JavaClient
JavaServe
src/test/java
JRE System Libr
Maven Depend
src
target
pom.xml
Outline
Task List
An outline is not available.
New
Open Type Hierarchy F4
Show In Alt+Shift+W
Open F3
Open With
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export...
Refresh F5
References
Declarations
Run As
Debug As
Profile As
Validate
Replace With
Restore from Local History...
Web Services
Team
Compare With
Properties Alt+Enter
1 Run on Server Alt+Shift+X, R
2 Java Application Alt+Shift+X, J
Run Configurations...
Snippets
Console
a\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 5:53:45 PM)
ITEC_4020.XML_RPC.JavaClient.java - XML-RPC/src/main/java

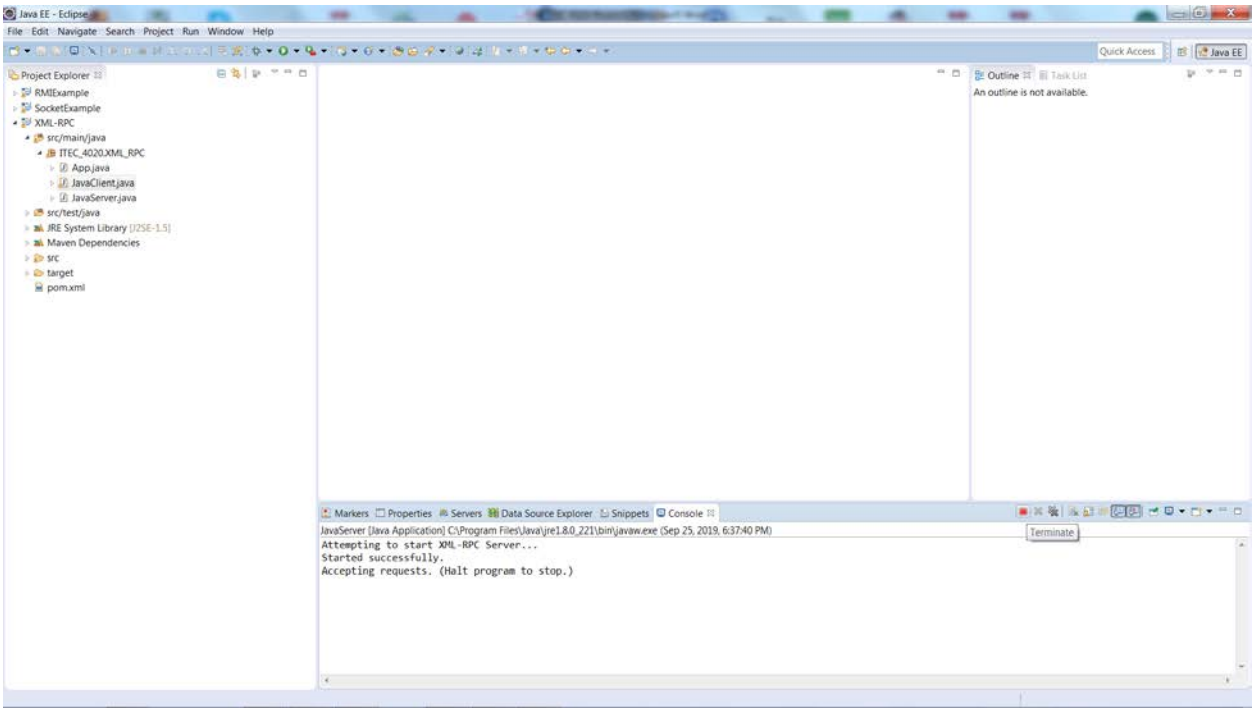

Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
SocketExample
XML-RPC
src/main/java
ITEC_4020.XML_RPC
App.java
JavaClient.java
JavaServer.java
src/test/java
JRE System Library [J2SE-1.5]
Maven Dependencies
src
target
pom.xml
Outline
Task List
An outline is not available.
Markers Properties Servers Data Source Explorer Snippets Console
JavaClient [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:33:42 PM)
Type the first integer and then hit Return, or type the keyword "Over" and then hit return to exit:


Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
SocketExample
XML-RPC
src/main/java
ITEC_4020.XML_RPC
App.java
JavaClient.java
JavaServer.java
src/test/java
JRE System Library [J2SE-1.5]
Maven Dependencies
src
target
pom.xml
Outline
Task List
An outline is not available.
Markers Properties Servers Data Source Explorer Snippets Console
JavaClient [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:33:42 PM)
Type the first integer and then hit Return, or type the keyword "Over" and then hit return to exit: 10
Type the second integer and then hit Return, or type the keyword "Over" and then hit return to exit: 20

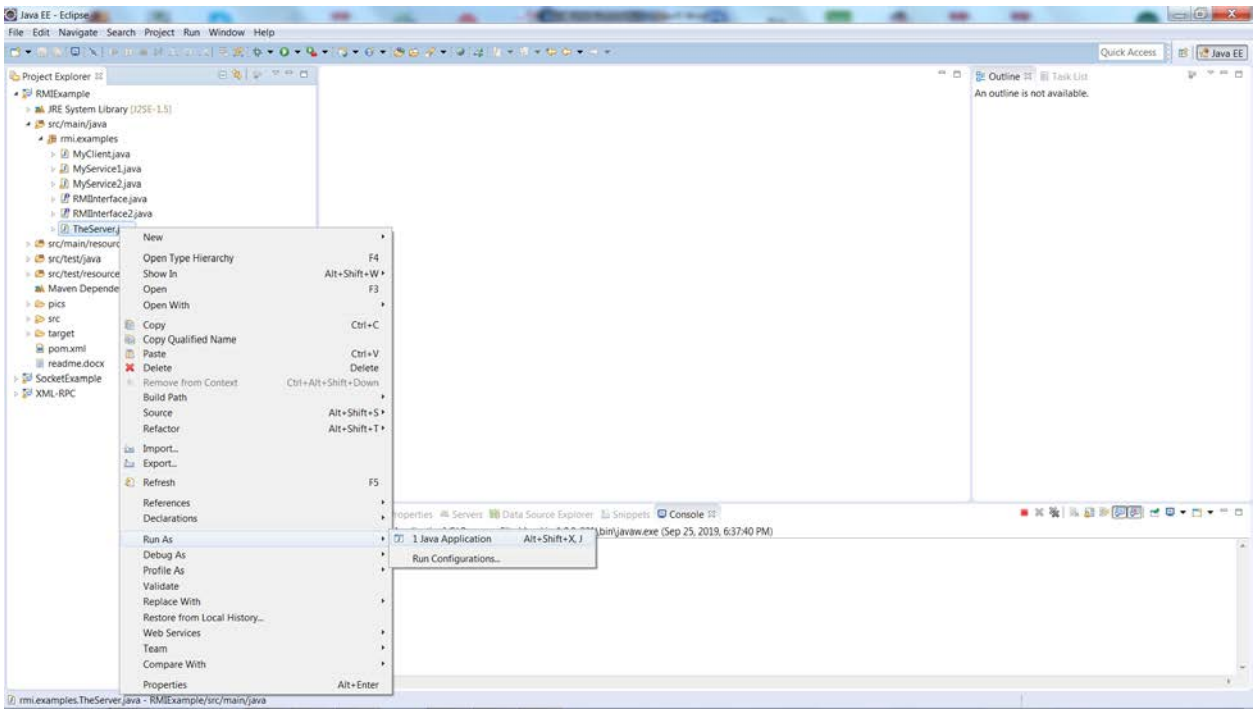

Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
SocketExample
XML-RPC
src/main/java
ITEC_4020.XML_RPC
App.java
JavaClient.java
JavaServer.java
src/test/java
JRE System Library [J2SE-1.5]
Maven Dependencies
src
target
pom.xml
Outline
Task List
An outline is not available.
Markers Properties Servers Data Source Explorer Snippets Console
JavaClient [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:35:45 PM)
Type the first integer and then hit Return, or type the keyword "Over" and then hit return to exit: 10
Type the second integer and then hit Return, or type the keyword "Over" and then hit return to exit: 20
The sum is: 30
Type the first integer and then hit Return, or type the keyword "Over" and then hit return to exit:

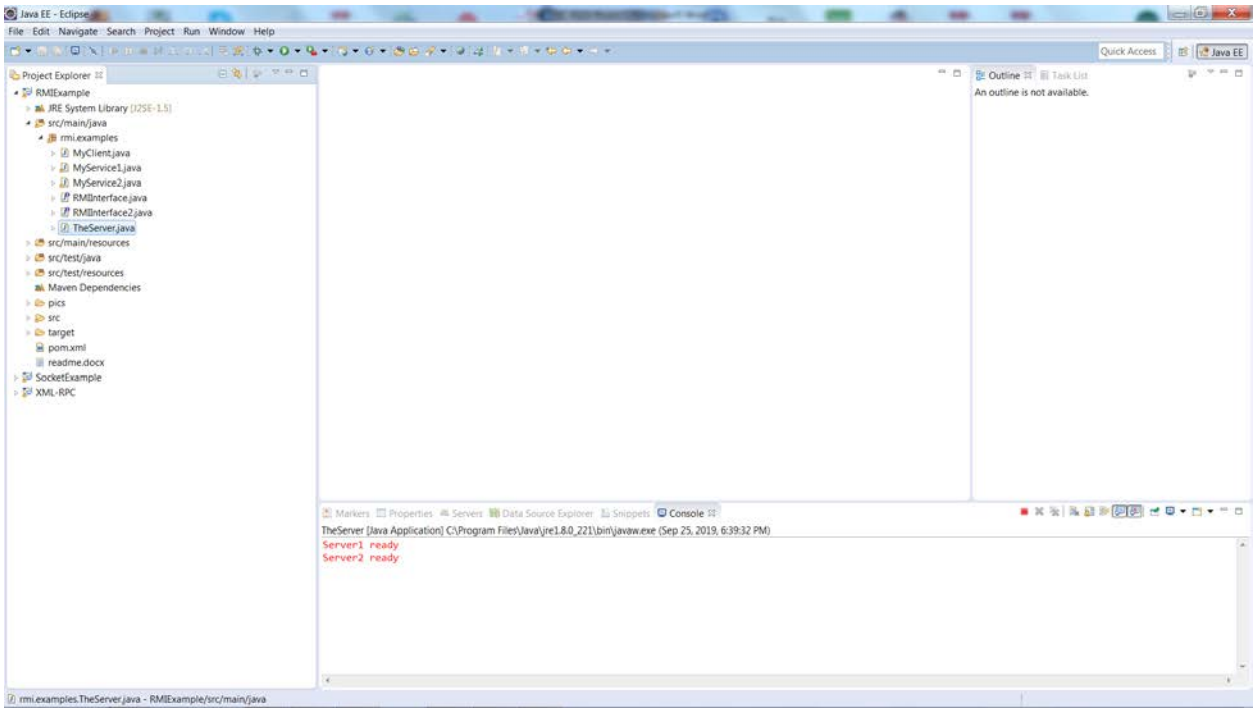

Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
SocketExample
XML-RPC
src/main/java
ITEC_4020.XML_RPC
App.java
JavaClient.java
JavaServer.java
src/test/java
JRE System Library [J2SE-1.5]
Maven Dependencies
src
target
pom.xml
Outline
Task List
An outline is not available.
Markers Properties Servers Data Source Explorer Snippets Console
<terminated> JavaClient [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:35:45 PM)
Type the first integer and then hit Return, or type the keyword "Over" and then hit return to exit: 10
Type the second integer and then hit Return, or type the keyword "Over" and then hit return to exit: 20
The sum is: 30
Type the first integer and then hit Return, or type the keyword "Over" and then hit return to exit: Over
Client terminates

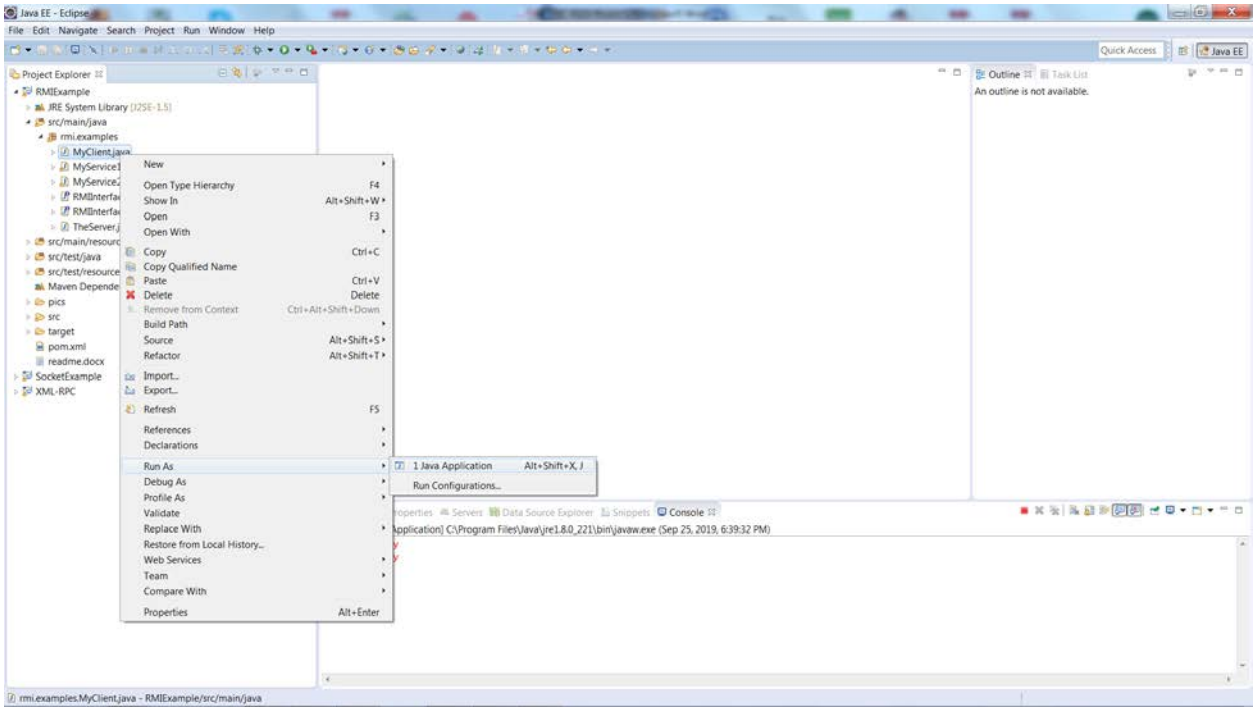

Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
SocketExample
XML-RPC
src/main/java
ITEC_4020.XML_RPC
App.java
JavaClient.java
JavaServer.java
src/test/java
JRE System Library [J2SE-1.5]
Maven Dependencies
src
target
pom.xml
Outline
Task List
An outline is not available.
Markers Properties Servers Data Source Explorer Snippets Console
1 JavaServer [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:32:20 PM)
2 <terminated> JavaClient [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:35:45 PM)
<terminated> JavaClient [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\j
Type the first integer and then hit Return, or type the keywor
Type the second integer and then hit Return, or type the keyword "Over" and then hit return to exit: 20
The sum is: 30
Type the first integer and then hit Return, or type the keyword "Over" and then hit return to exit: Over
Client terminates

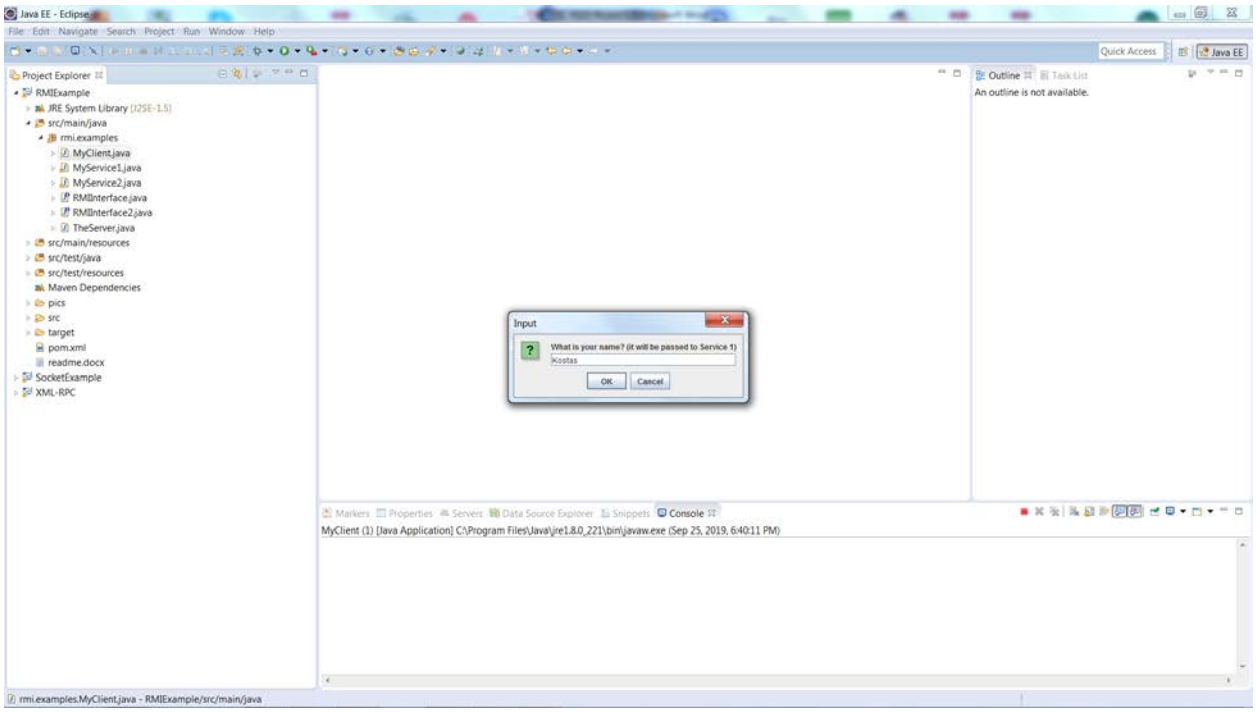


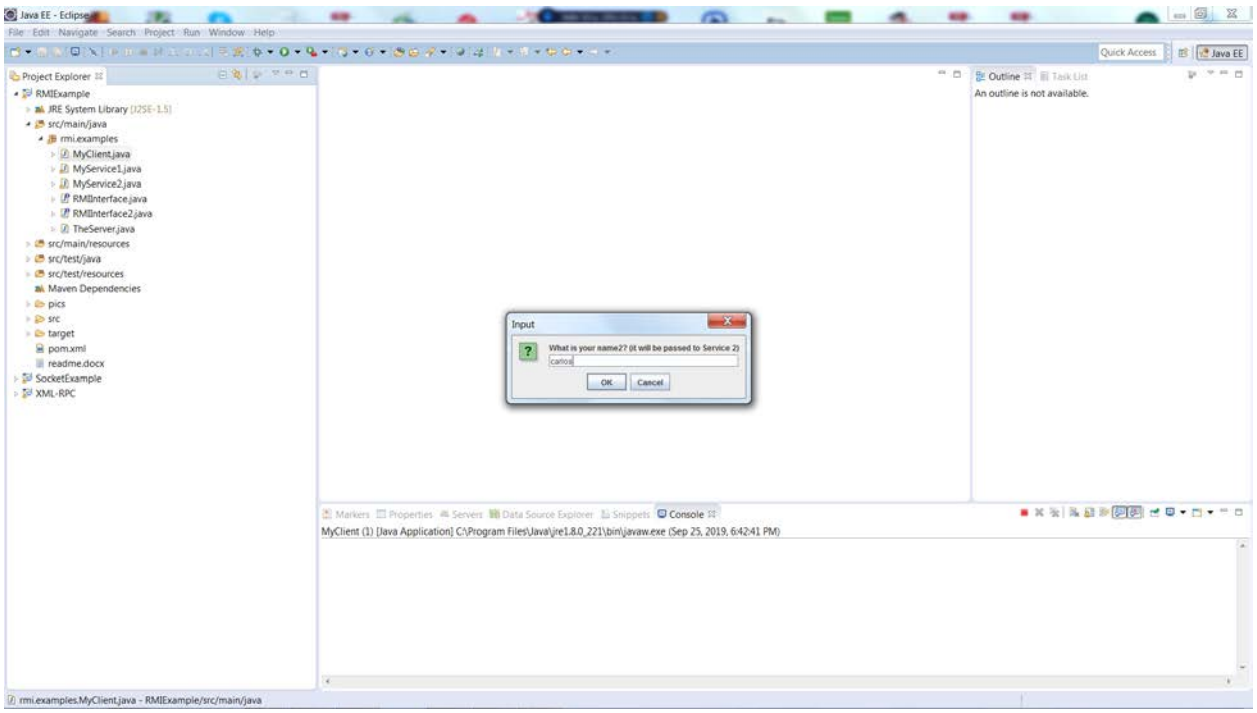
C. Running the RMI example

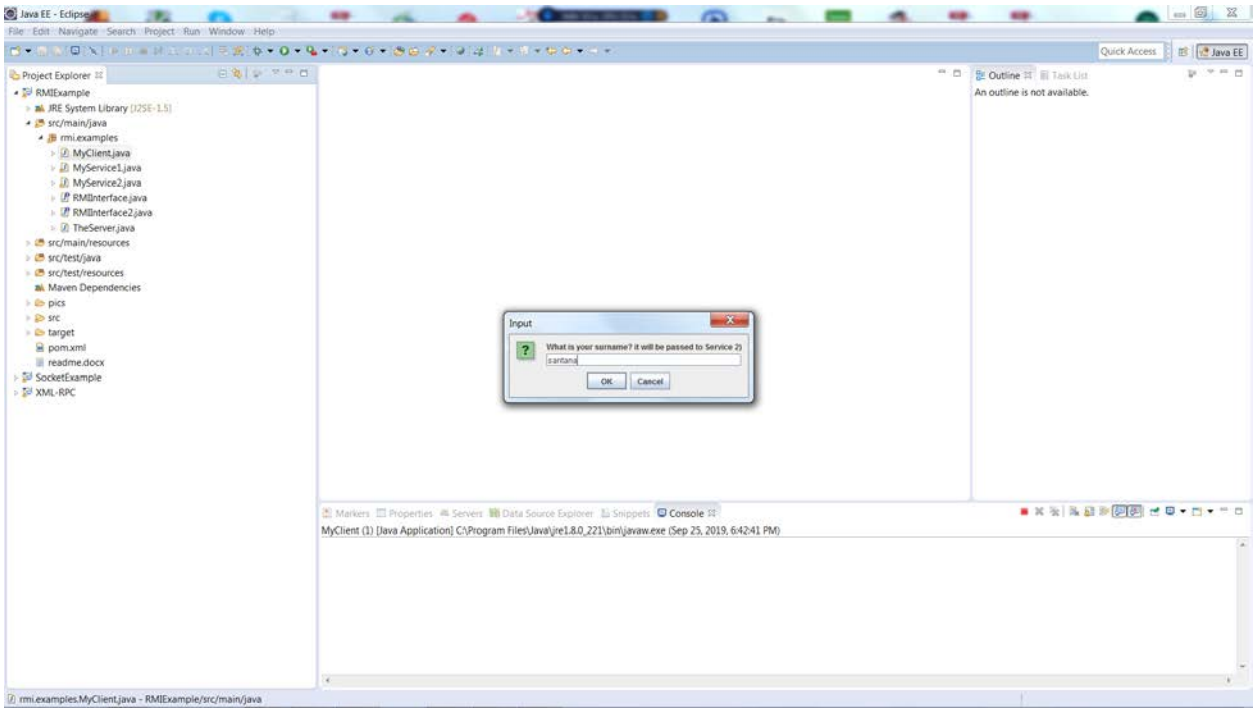


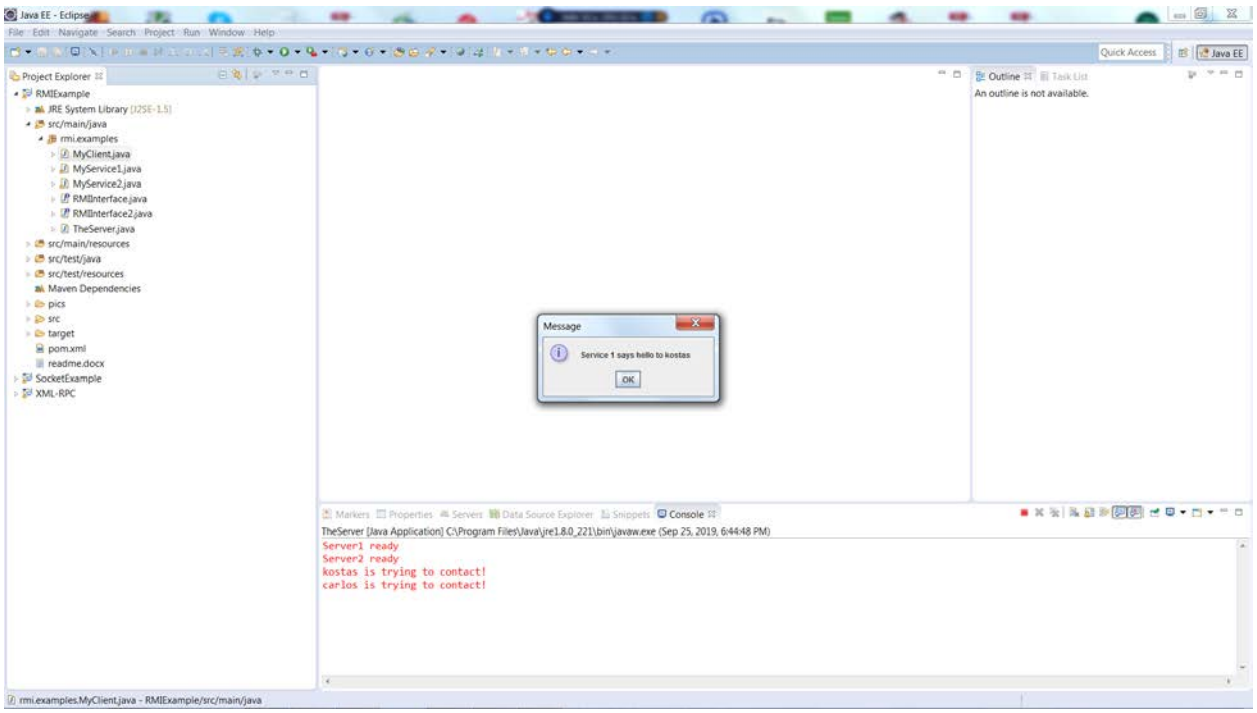

Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
JRE System Library [J2SE-1.5]
src/main/java
rmi.examples
MyClient.java
MyService1.java
MyService2.java
RMIInterface.java
RMIInterface2.java
TheServer.java
src/main/resources
src/test/java
src/test/resources
Maven Dependencies
pics
src
target
pom.xml
readme.docx
SocketExample
XML-RPC
Outline
Task List
An outline is not available.
Markers Properties Servers Data Source Explorer Snippets Console
TheServer [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:39:32 PM)
Server1 ready
Server2 ready
rmi.examples.TheServer.java - RMIExample/src/main/java

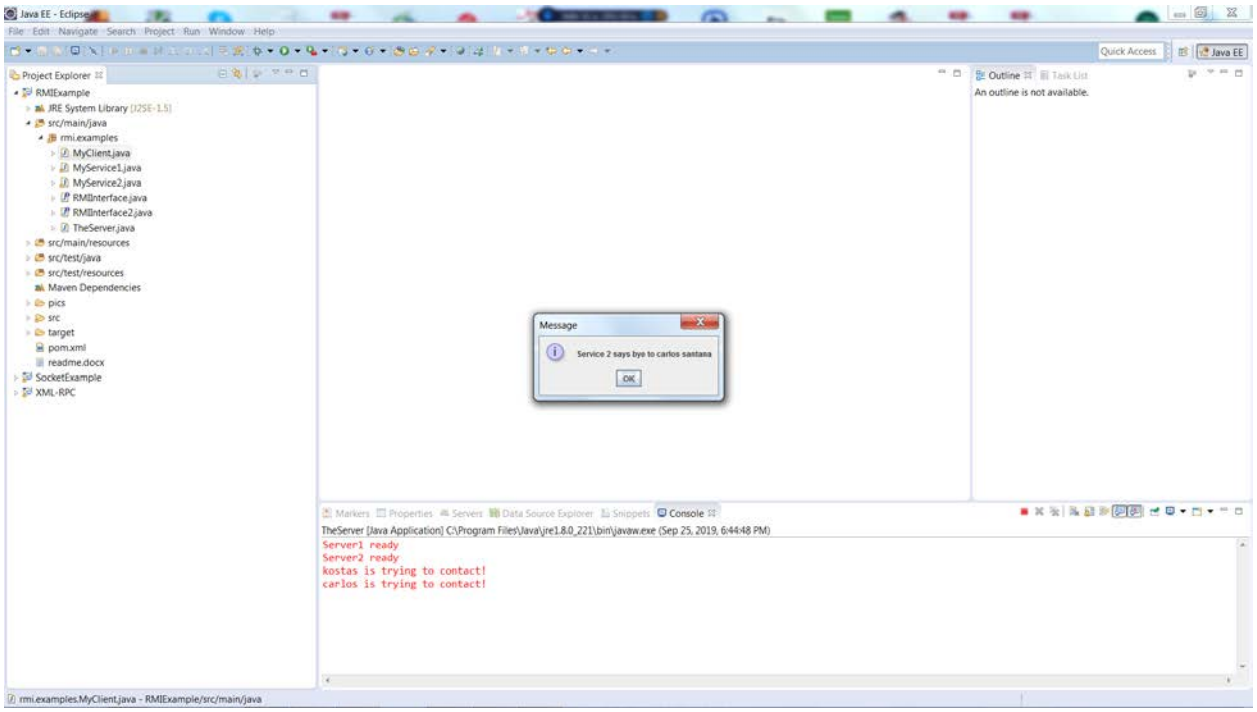

Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
JRE System Library [J2SE-1.5]
src/main/java
rmi.examples
MyClient.java
pom.xml
readme.docx
SocketExample
XML-RPC
Outline
Task List
An outline is not available.
New
Open Type Hierarchy F4
Show In Alt+Shift+W
Open F3
Open With
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export...
Refresh F5
References
Declarations
Run As
Debug As
Profile As
Validate
Replace With
Restore from Local History...
Web Services
Team
Compare With
Properties Alt+Enter
1 Java Application Alt+Shift+X, J
Run Configurations...
Console
rmi.examples.MyClient.java - RMIExample/src/main/java

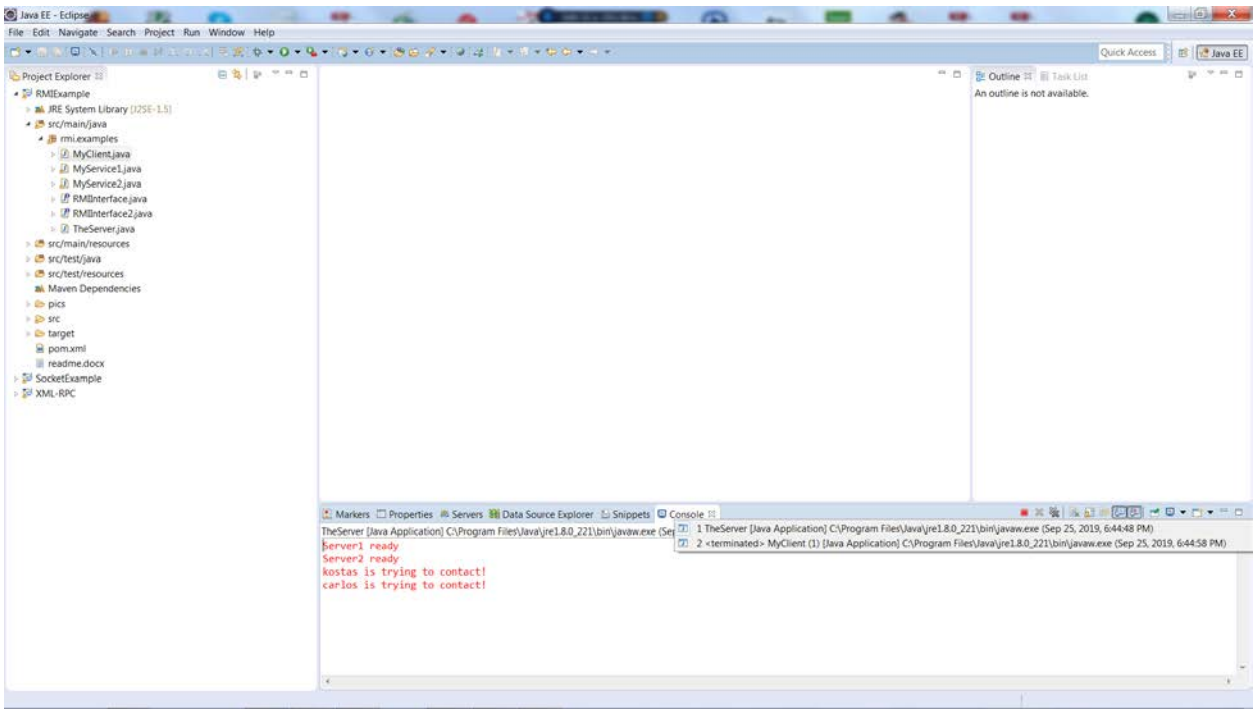

Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
JRE System Library [J2SE-1.5]
src/main/java
rmi.examples
MyClient.java
MyService1.java
MyService2.java
RMIInterface.java
RMIInterface2.java
TheServer.java
src/main/resources
src/test/java
src/test/resources
Maven Dependencies
pics
src
target
pom.xml
readme.docx
SocketExample
XML-RPC
Outline
Task List
An outline is not available.
Input
What is your name? (it will be passed to Service 1)
Kostas
OK
Cancel
Markers Properties Servers Data Source Explorer Snippets Console
MyClient (1) [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:40:11 PM)
rmi.examples.MyClient.java - RMIExample/src/main/java

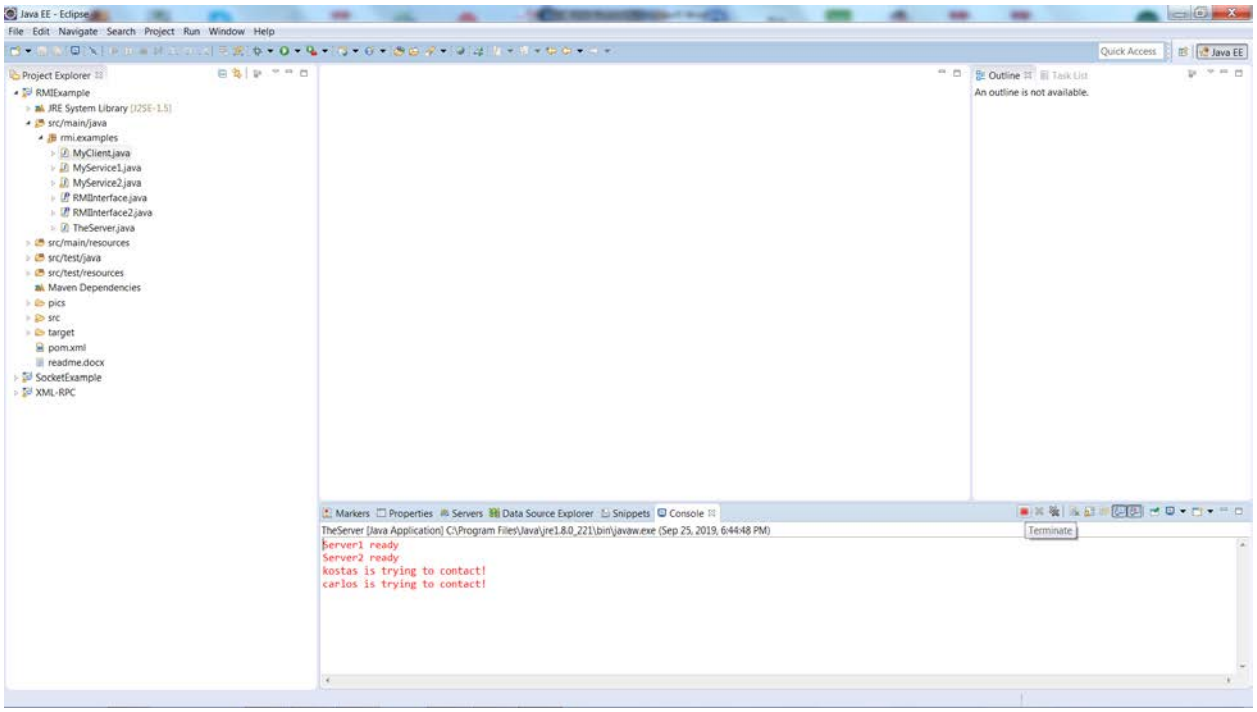

Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
JRE System Library [J2SE-1.5]
src/main/java
rmi.examples
MyClient.java
MyService1.java
MyService2.java
RMIInterface.java
RMIInterface2.java
TheServer.java
src/main/resources
src/test/java
src/test/resources
Maven Dependencies
pics
src
target
pom.xml
readme.docx
SocketExample
XML-RPC
Outline
Task List
An outline is not available.
Input
What is your name2? (it will be passed to Service 2)
Carlos
OK
Cancel
Markers Properties Servers Data Source Explorer Snippets Console
MyClient (1) [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:42:41 PM)
rmi.examples.MyClient.java - RMIExample/src/main/java


Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
JRE System Library [J2SE-1.5]
src/main/java
rmi.examples
MyClient.java
MyService1.java
MyService2.java
RMIInterface.java
RMIInterface2.java
TheServer.java
src/main/resources
src/test/java
src/test/resources
Maven Dependencies
pics
src
target
pom.xml
readme.docx
SocketExample
XML-RPC
Outline
Task List
An outline is not available.
Input
What is your surname? it will be passed to Service 2)
santana
OK
Cancel
Markers Properties Servers Data Source Explorer Snippets Console
MyClient (1) [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:42:41 PM)
rmi.examples.MyClient.java - RMIExample/src/main/java


Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
JRE System Library [J2SE-1.5]
src/main/java
rmi.examples
MyClient.java
MyService1.java
MyService2.java
RMIInterface.java
RMIInterface2.java
TheServer.java
src/main/resources
src/test/java
src/test/resources
Maven Dependencies
pics
src
target
pom.xml
readme.docx
SocketExample
XML-RPC
Outline
Task List
An outline is not available.
Message
Service 1 says hello to kostas
OK
Markers Properties Servers Data Source Explorer Snippets Console
TheServer [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:44:48 PM)
Server1 ready
Server2 ready
kostas is trying to contact!
carlos is trying to contact!
rmi.examples.MyClient.java - RMIExample/src/main/java

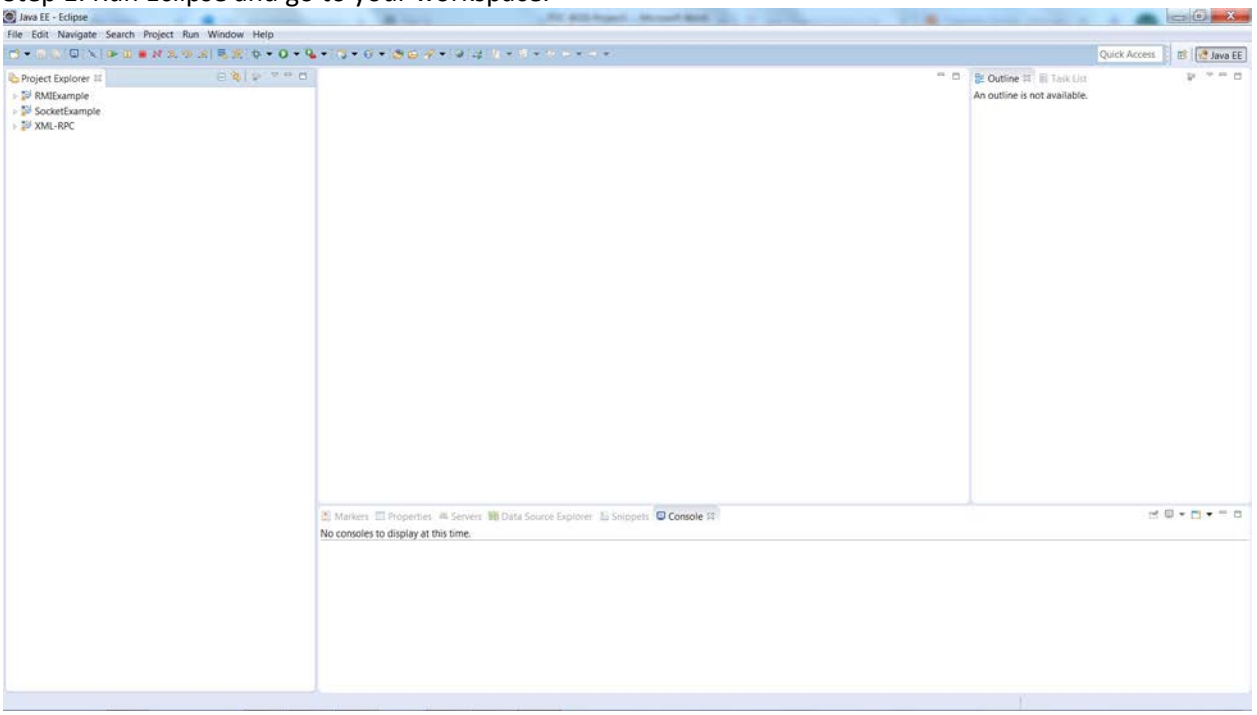

Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
JRE System Library [J2SE-1.5]
src/main/java
rmi.examples
MyClient.java
MyService1.java
MyService2.java
RMIInterface.java
RMIInterface2.java
TheServer.java
src/main/resources
src/test/java
src/test/resources
Maven Dependencies
pics
src
target
pom.xml
readme.docx
SocketExample
XML-RPC
Outline
Task List
An outline is not available.
Message
Service 2 says bye to carlos santana
OK
Markers Properties Servers Data Source Explorer Snippets Console
TheServer [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:44:48 PM)
Server1 ready
Server2 ready
kostas is trying to contact!
carlos is trying to contact!
rmi.examples.MyClient.java - RMIExample/src/main/java


Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
JRE System Library [J2SE-1.5]
src/main/java
rmi.examples
MyClient.java
MyService1.java
MyService2.java
RMIInterface.java
RMIInterface2.java
TheServer.java
src/main/resources
src/test/java
src/test/resources
Maven Dependencies
pics
src
target
pom.xml
readme.docx
SocketExample
XML-RPC
Outline
Task List
An outline is not available.
Markers Properties Servers Data Source Explorer Snippets Console
TheServer [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:44:48 PM)
Server1 ready
Server2 ready
kostas is trying to contact!
carlos is trying to contact!
1 TheServer [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:44:48 PM)
2 <terminated> MyClient (1) [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:44:58 PM)

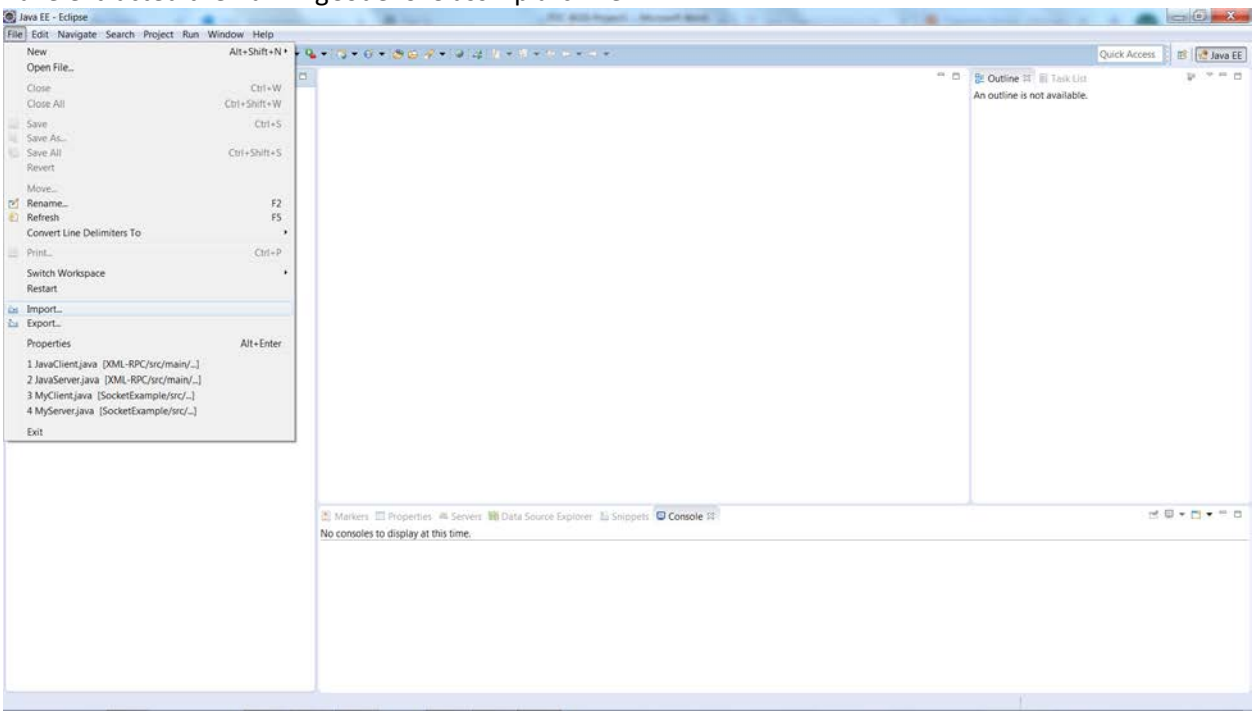

Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
JRE System Library [J2SE-1.5]
src/main/java
rmi.examples
MyClient.java
MyService1.java
MyService2.java
RMIInterface.java
RMIInterface2.java
TheServer.java
src/main/resources
src/test/java
src/test/resources
Maven Dependencies
pics
src
target
pom.xml
readme.docx
SocketExample
XML-RPC
Outline
Task List
An outline is not available.
Markers
Properties
Servers
Data Source Explorer
Snippets
Console
TheServer [Java Application] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (Sep 25, 2019, 6:44:48 PM)
Terminate
Server1 ready
Server2 ready
kostas is trying to contact!
carlos is trying to contact!

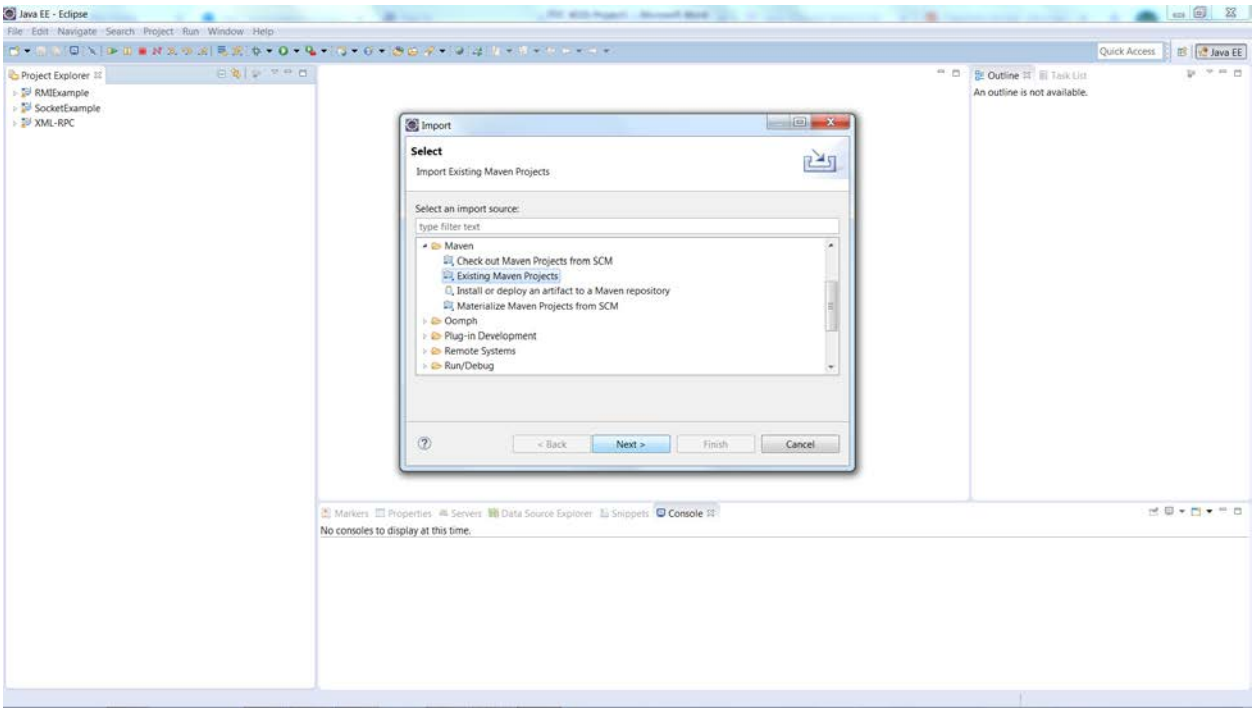
Appendix III
Import to Eclipse the Banking application

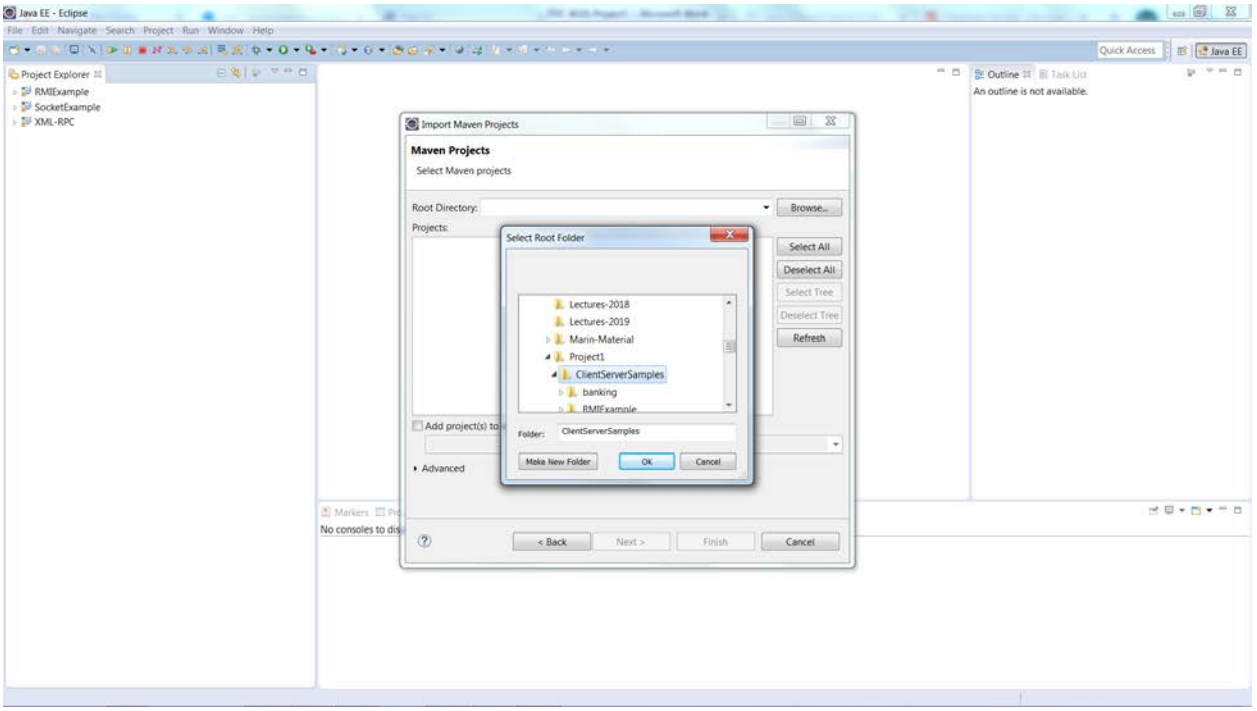
Step 1. Run Eclipse and go to your workspace.

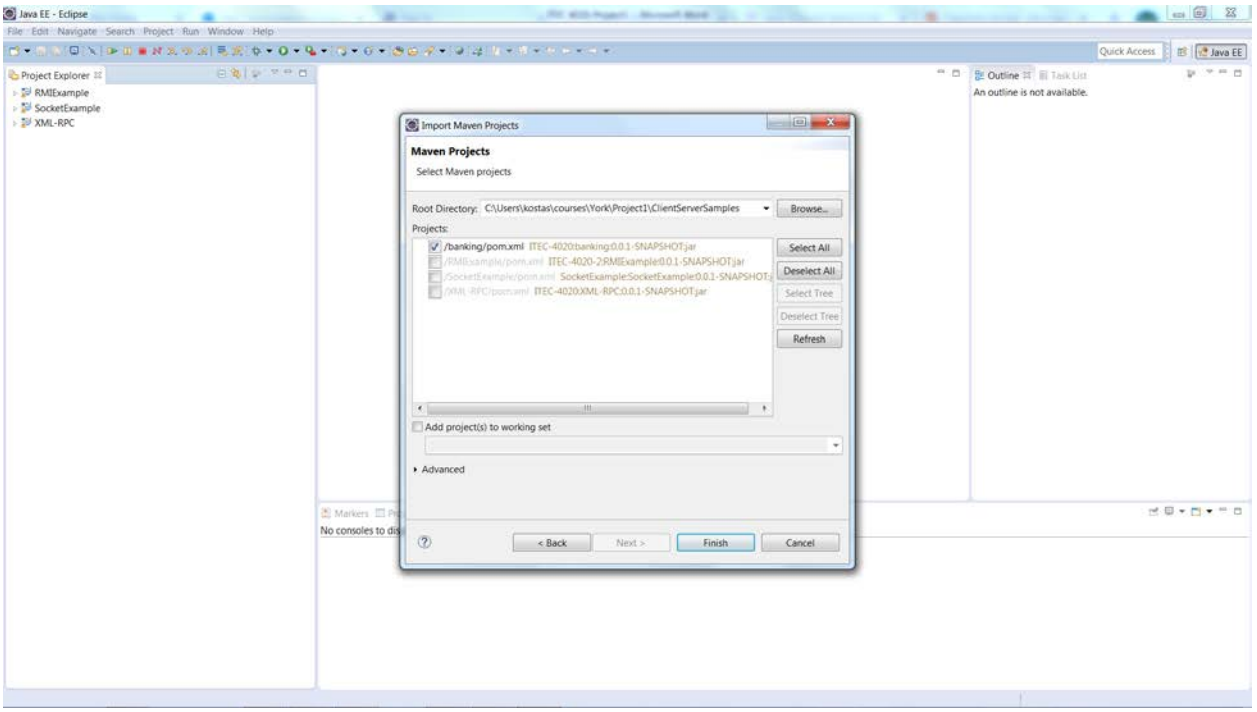


Step 2. Import the banking code using Import → Maven → Existing Maven Projects. Select the directory where yo have extracted the BankingCodeForClass.zip archive.

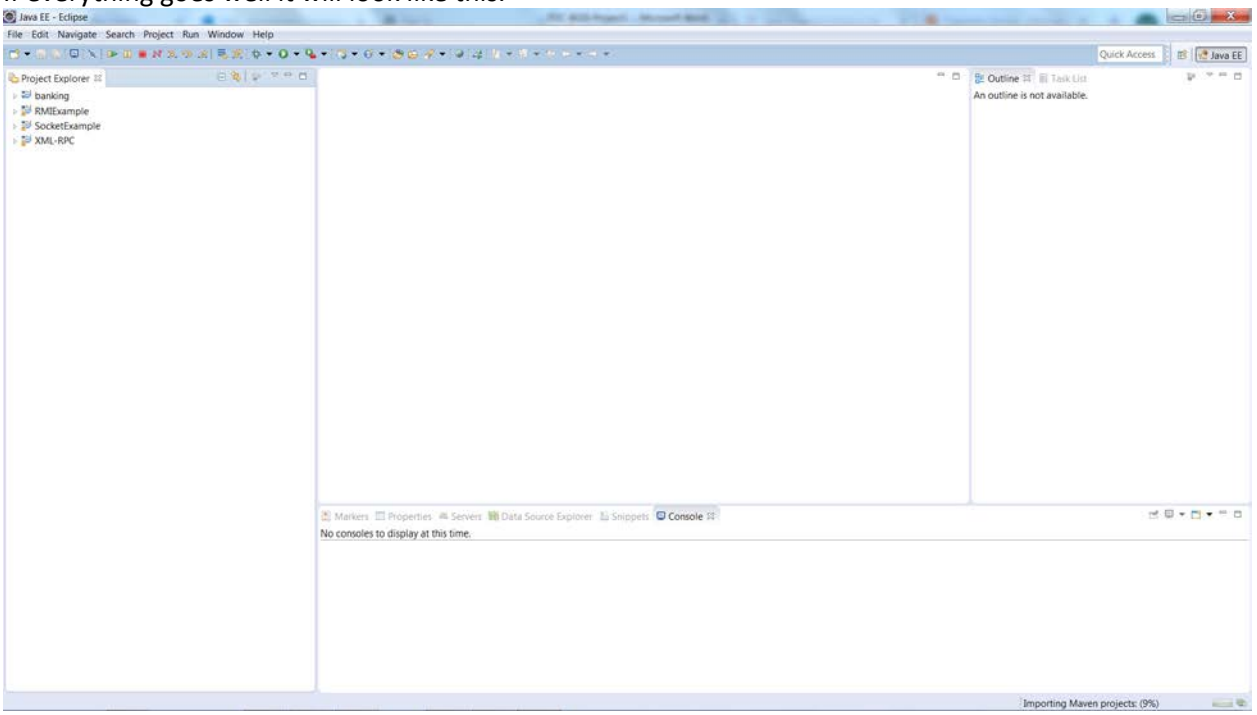


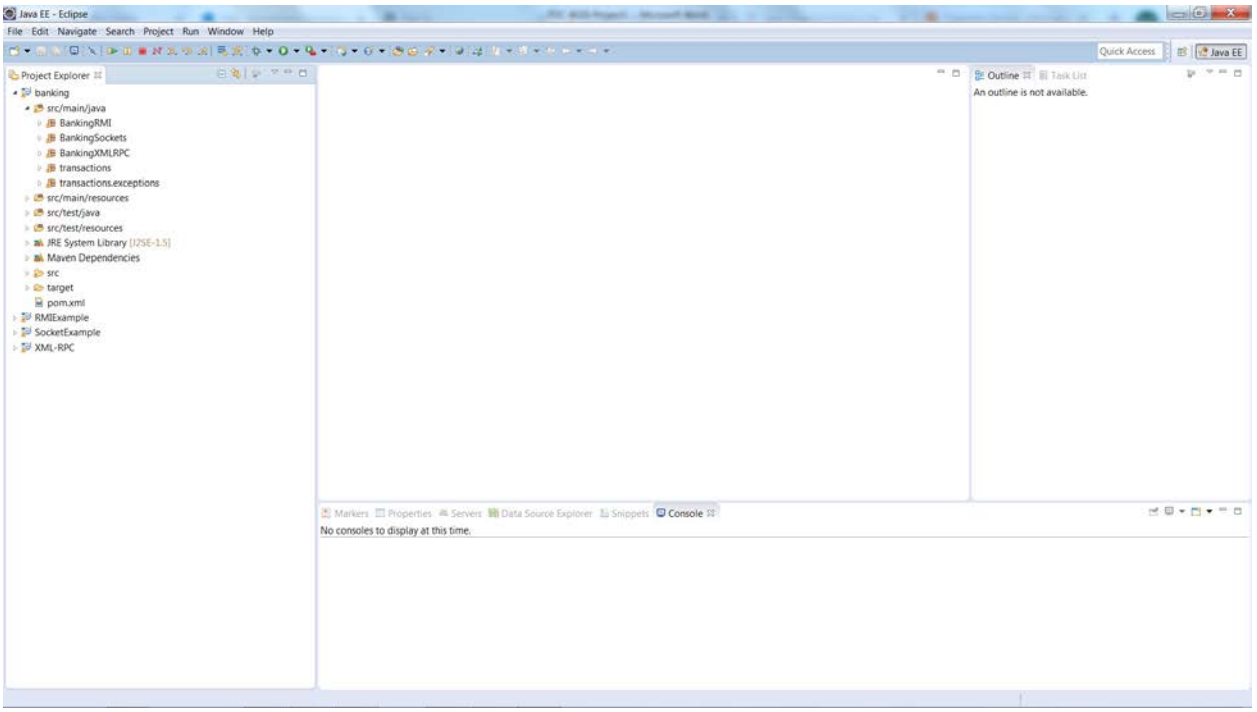

Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
SocketExample
XML-RPC
Outline
Task List
An outline is not available.
Import
Select
Import Existing Maven Projects
Select an import source:
type filter text
Maven
Check out Maven Projects from SCM
Existing Maven Projects
Install or deploy an artifact to a Maven repository
Materialize Maven Projects from SCM
Oomph
Plug-in Development
Remote Systems
Run/Debug
< Back
Next >
Finish
Cancel
Markers
Properties
Servers
Data Source Explorer
Snippets
Console
No consoles to display at this time.


Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
RMIExample
SocketExample
XML-RPC
Outline
Task List
An outline is not available.
Import Maven Projects
Maven Projects
Select Maven projects
Root Directory:
Browse...
Projects:
Select All
Deselect All
Select Tree
Deselect Tree
Refresh
Select Root Folder
Lectures-2018
Lectures-2019
Marin-Material
Project1
ClientServerSamples
banking
RMIExample
Folder:
ClientServerSamples
Make New Folder
OK
Cancel
Add project(s) to
Advanced
< Back
Next >
Finish
Cancel



If everything goes well it will look like this.




Java EE - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java EE
Project Explorer
banking
src/main/java
BankingRMI
BankingSockets
BankingXMLRPC
transactions
transactions.exceptions
src/main/resources
src/test/java
src/test/resources
JRE System Library [J2SE-1.5]
Maven Dependencies
src
target
pom.xml
RMIExample
SocketExample
XML-RPC
Outline
Task List
An outline is not available.
Markers
Properties
Servers
Data Source Explorer
Snippets
Console
No consoles to display at this time.