Internet Engineering

Tomasz Babczyński, Zofia Kruczkiewicz
Tomasz Kubik

Information systems modelling– UML and service description languages

Laboratory 3

Choose yourself and new technologies

HUMAN CAPITAL
HUMAN – BEST INVESTMENT!

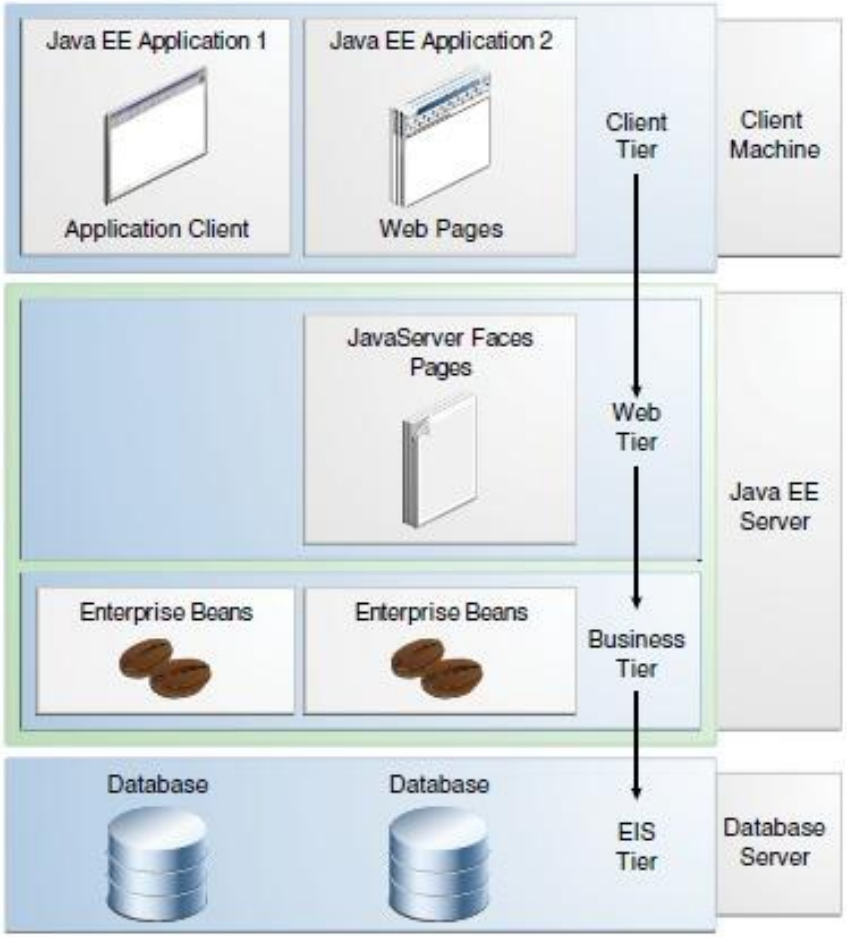
Wrocław University of Technology

EUROPEAN SOCIAL FUND

Project co-financed from the EU European Social Fund

Examplary multitiered Information System

(Java EE 7 – Tutorial Java EE 7)



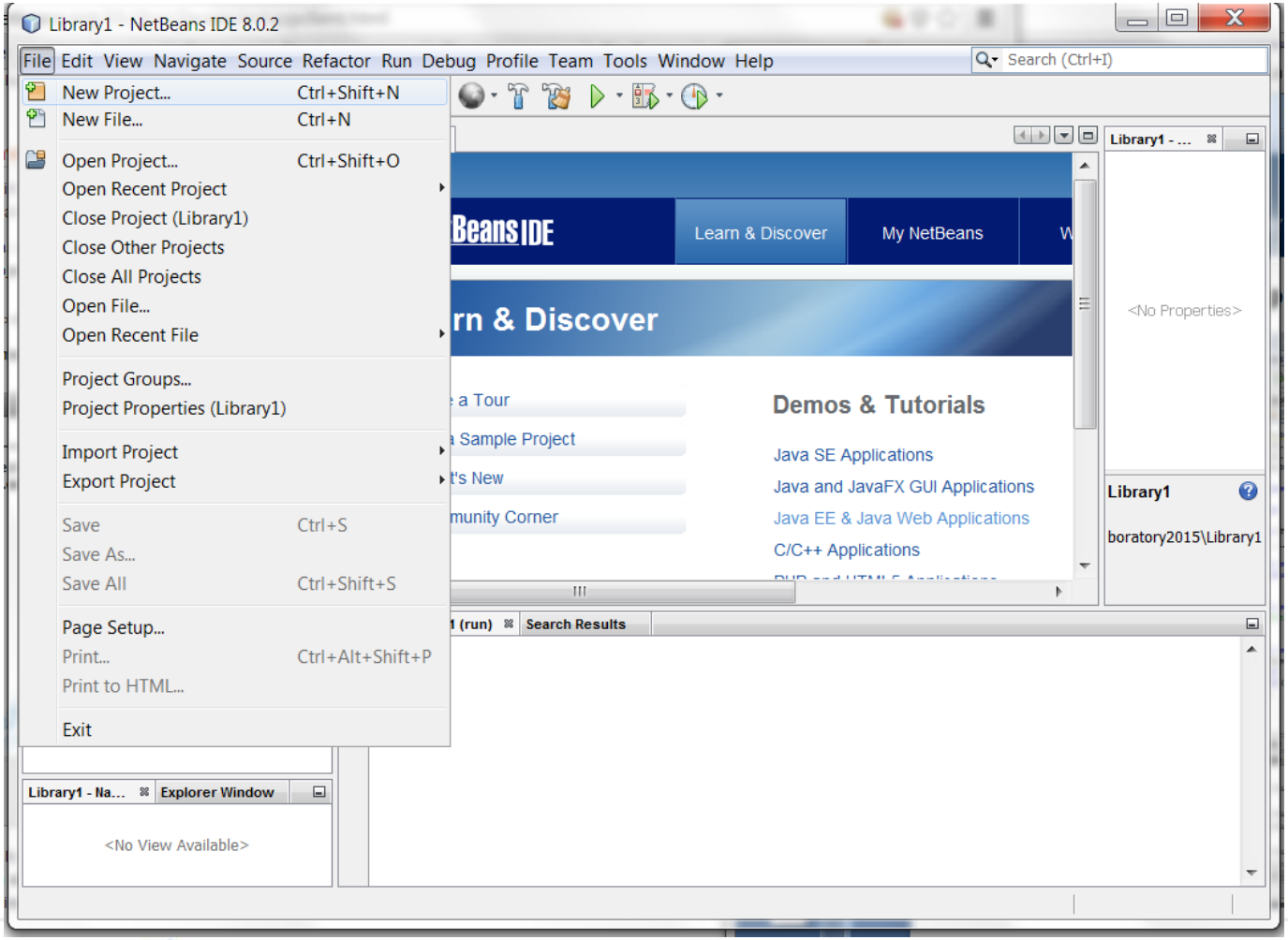
Steps for building the client tier of a multitier application

1. **Create the Java EE project - right click File/New Project**
2. **Creation the library project for interface of EJB**
3. **Adding the EJB as the facade**
4. **Creation the library project for code from lab1-2**
5. **Adding the library project (p.4) to the EJB module of EE project and its interface**
6. **Creation of EE client tier**
7. **Build, deploy and run the created software**
8. **Some information about updating Java EE projects**

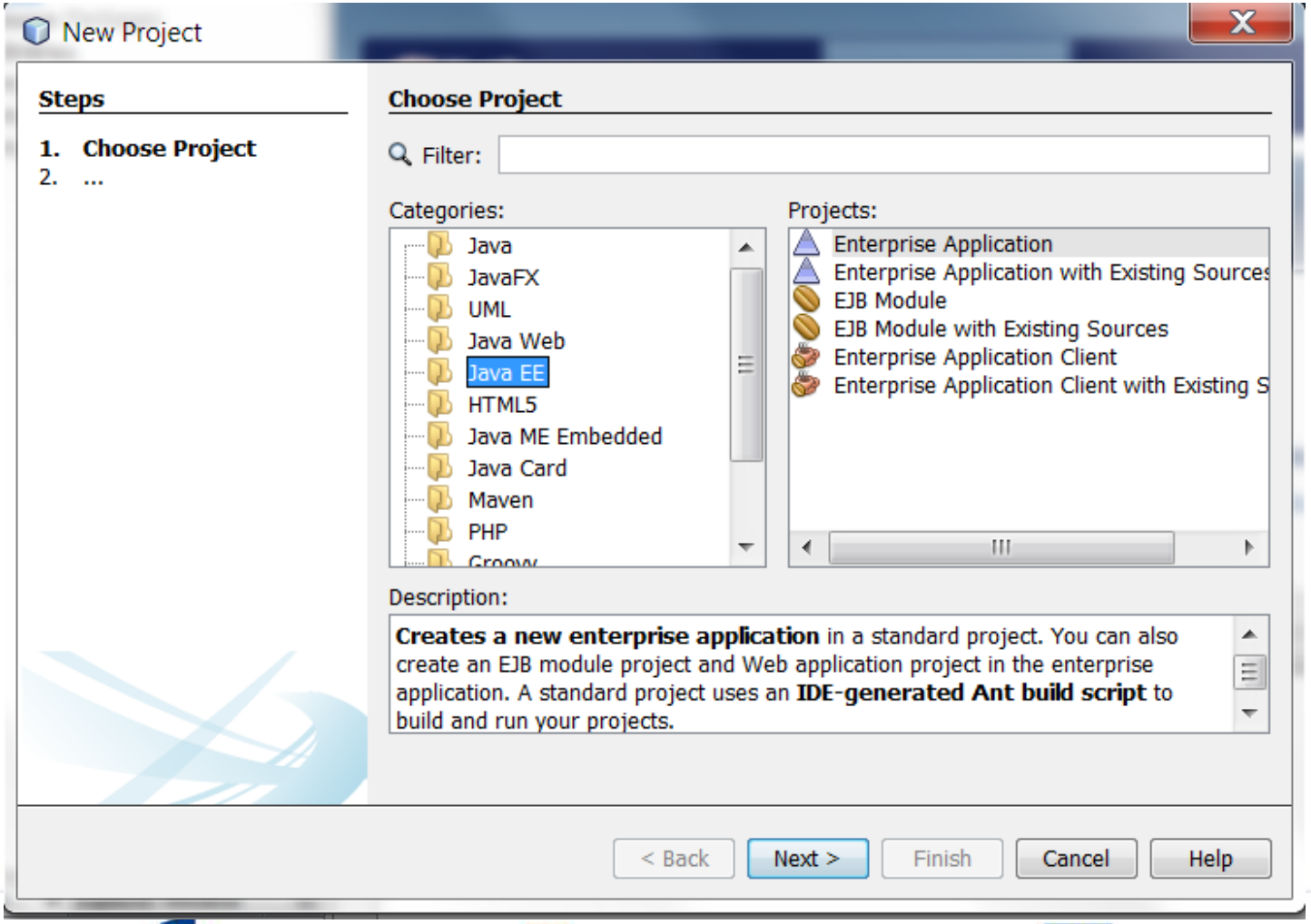
Add your own results from the lab2 to this application in the 2, 6 steps and develop some changes in the Java Class Library project made in the 4 step (the Library1 project).

Example (Java EE & Java Web Learning Trail**)**
https://netbeans.org/kb/docs/javaee/entappclient.html
from
https://netbeans.org/kb/trails/java-ee.html?utm_source=netbeans&utm_campaign=welcomepage

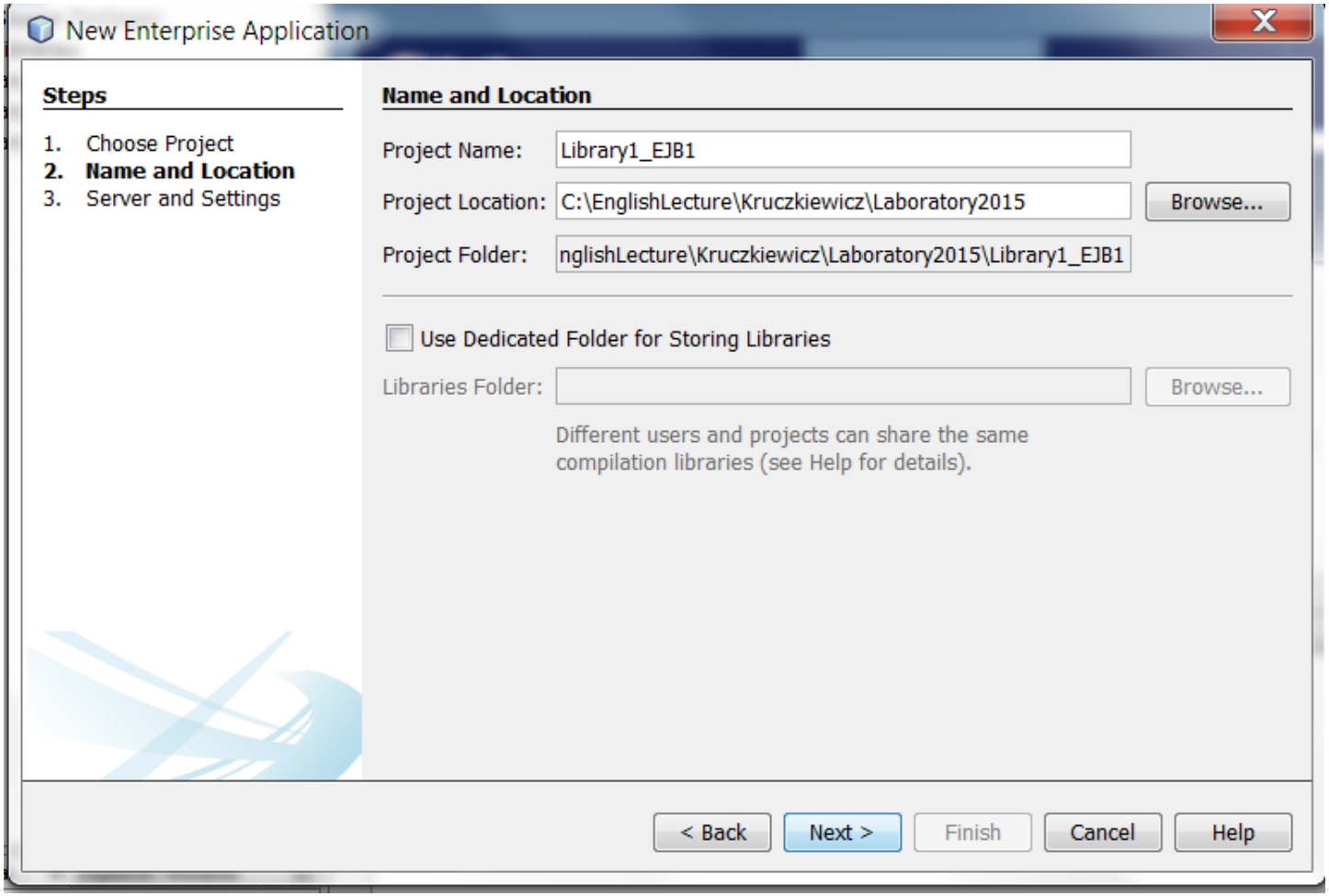
1. / 1.1.Create the Java EE project - right click File/New Project



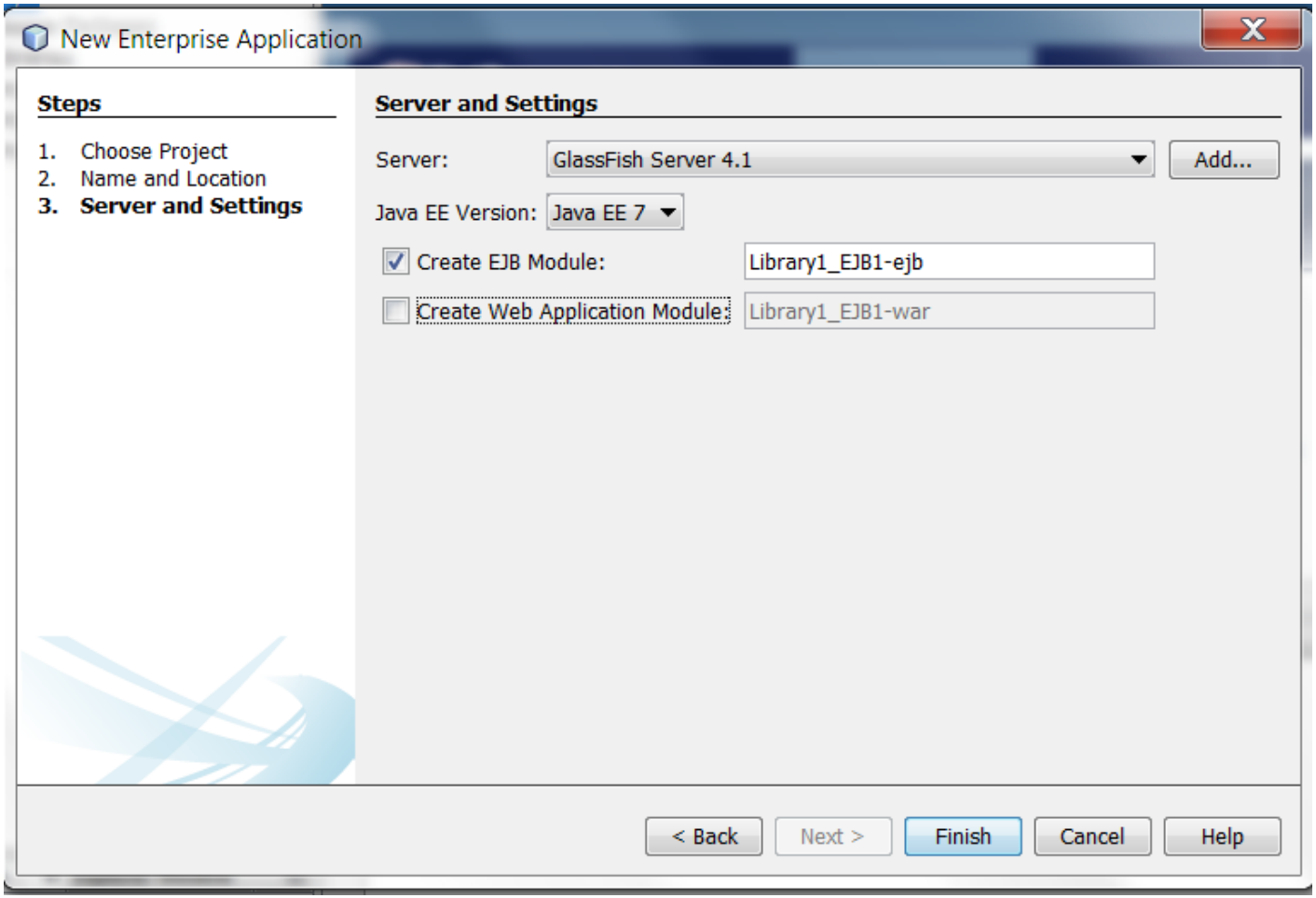
1.2. Select Java EE\ Enterprise Application and click Next



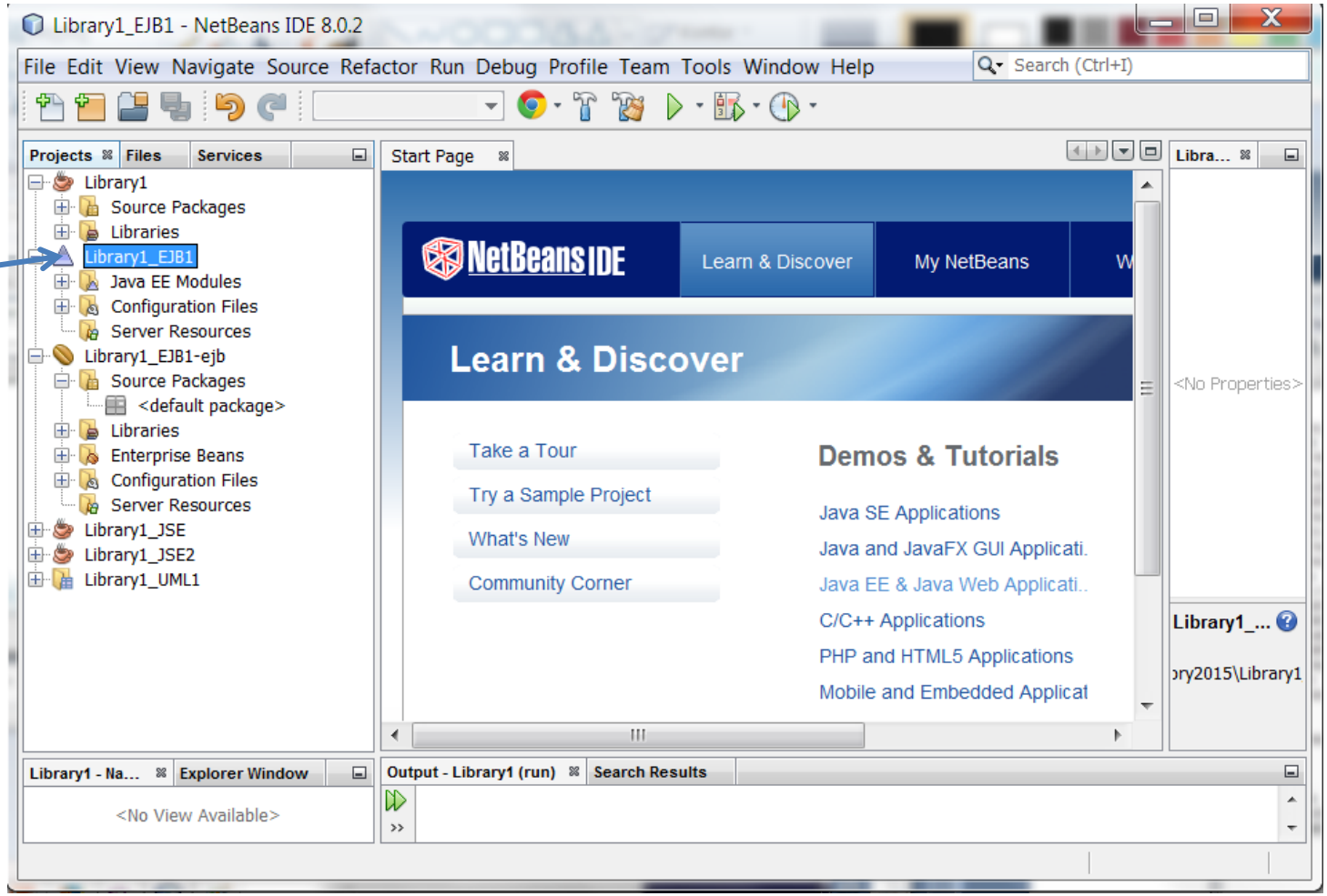
1.3. Name and Location of EE project and next



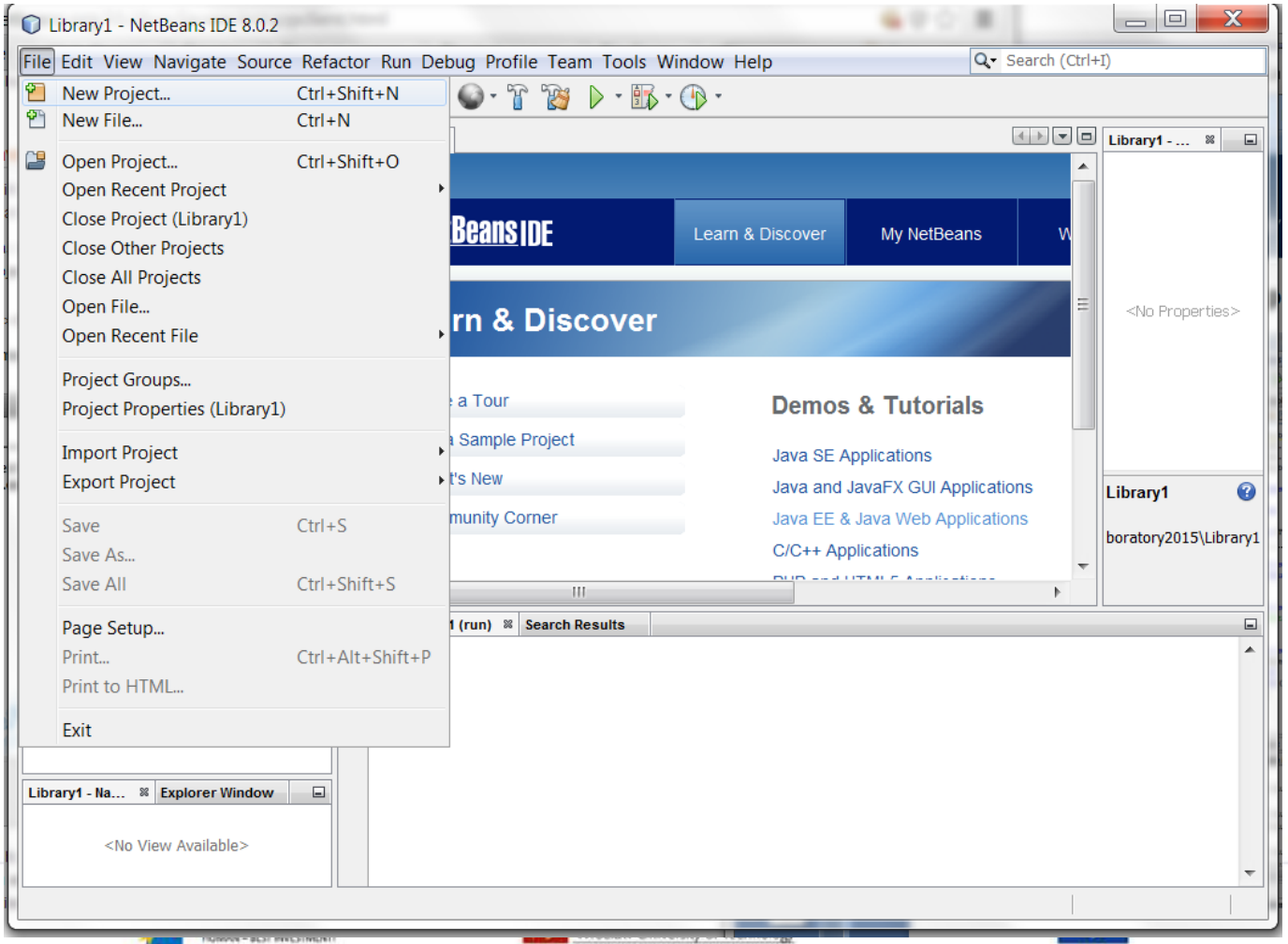
1.4. Server and Settings of EE project and next



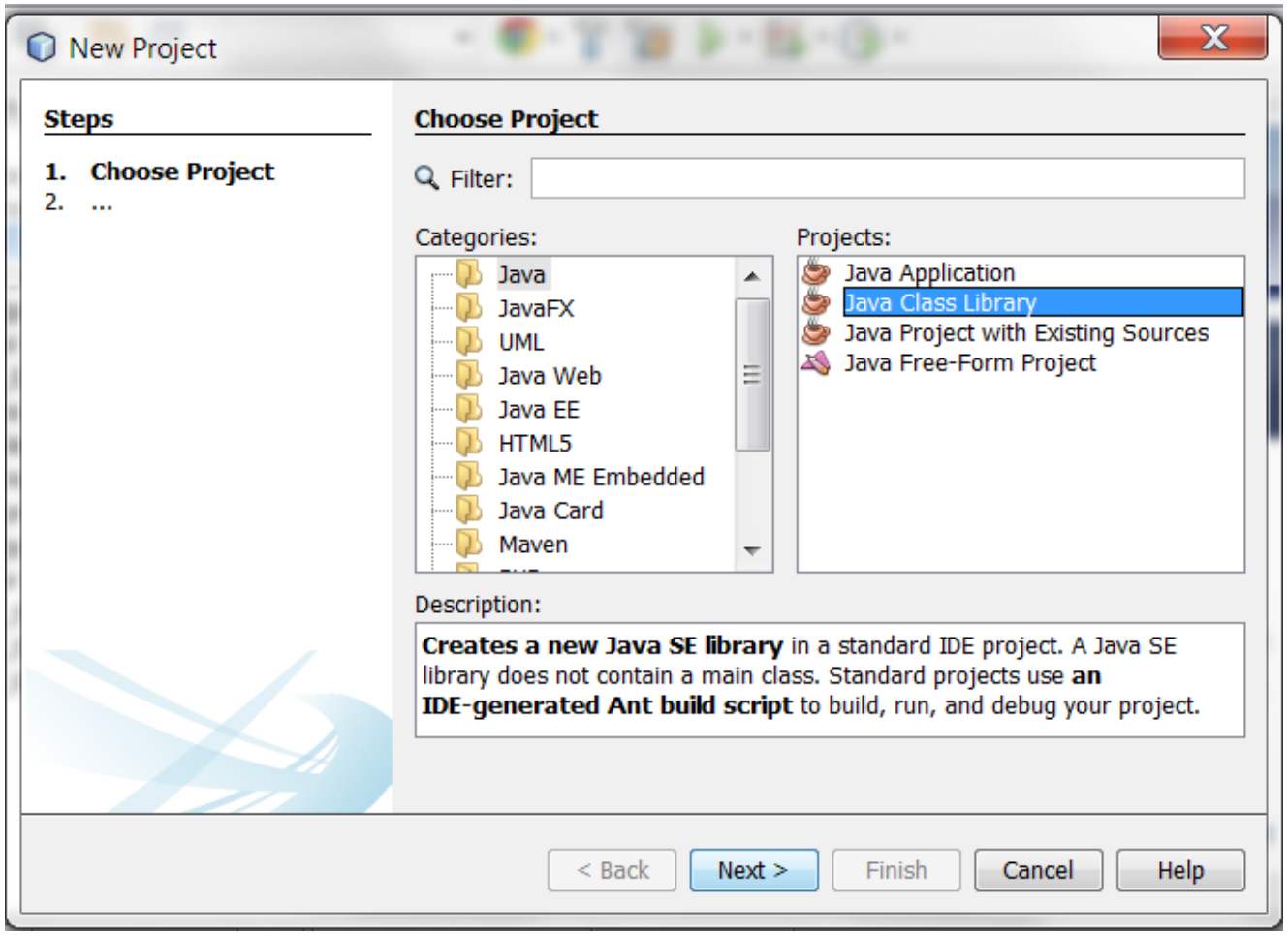
1.5. The new EE project



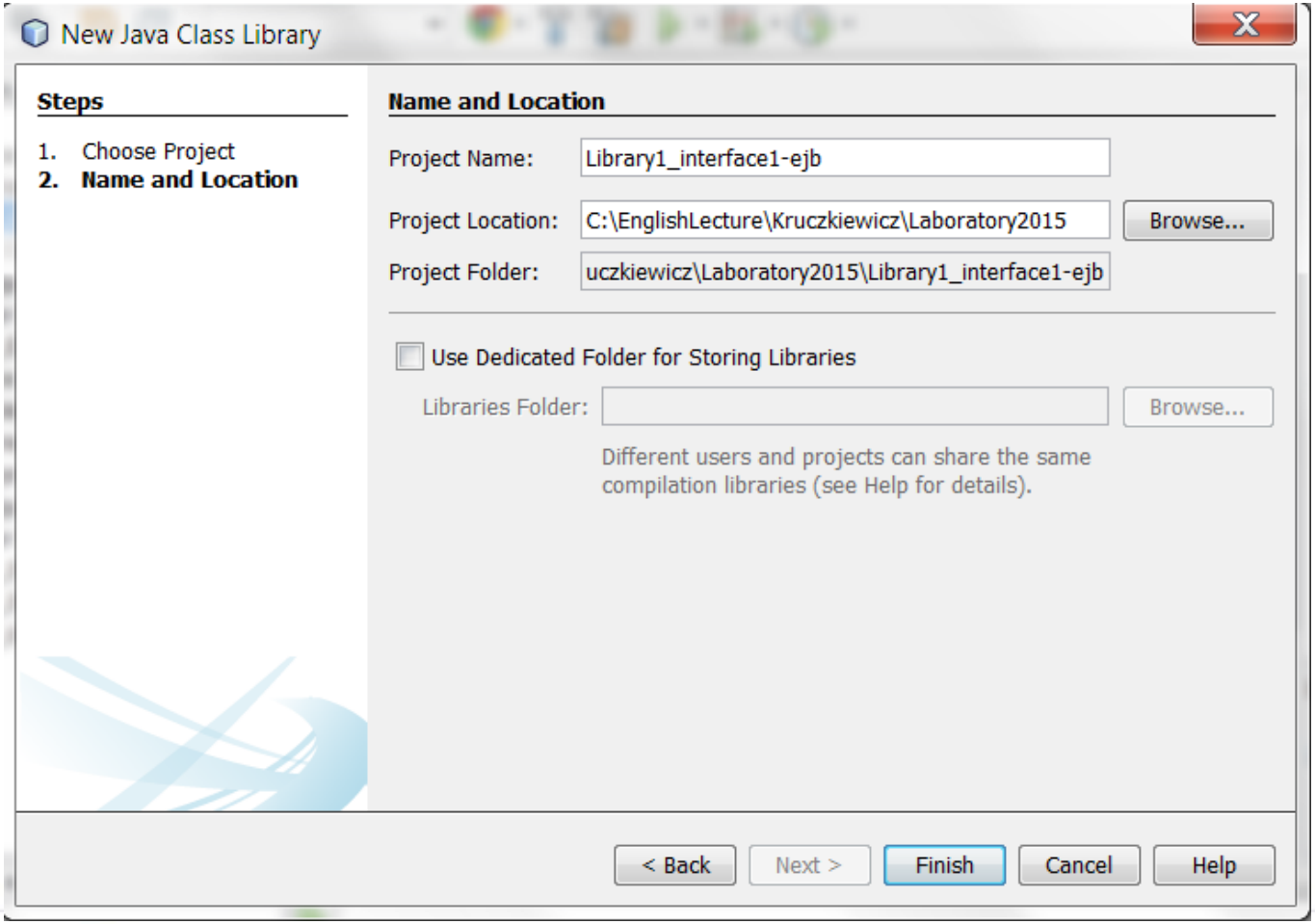
2. / 2.1. Creation the library project for interface of EJB



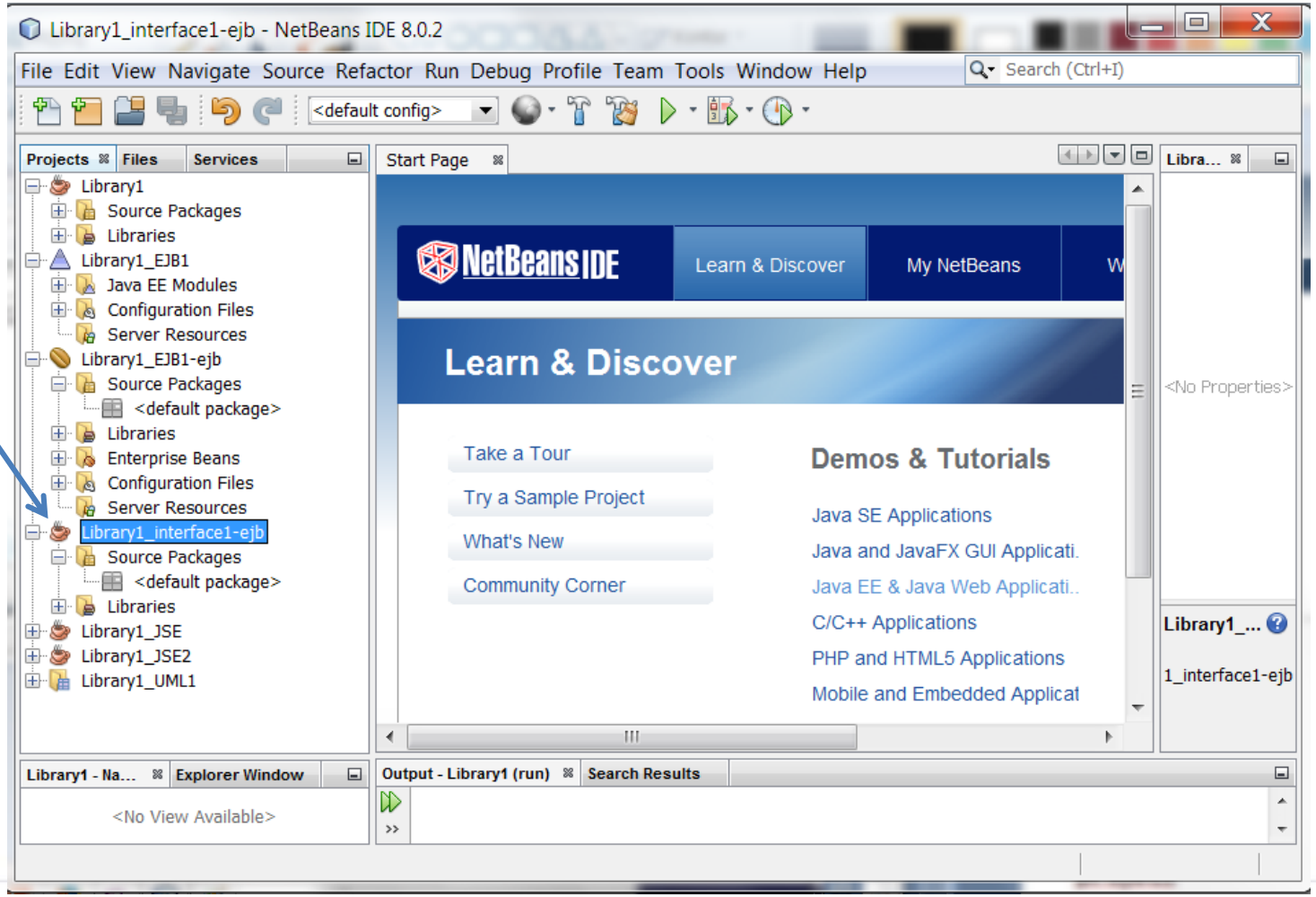
2.2. Creation the library project for interface of EJB – select Java/Java Class Library items



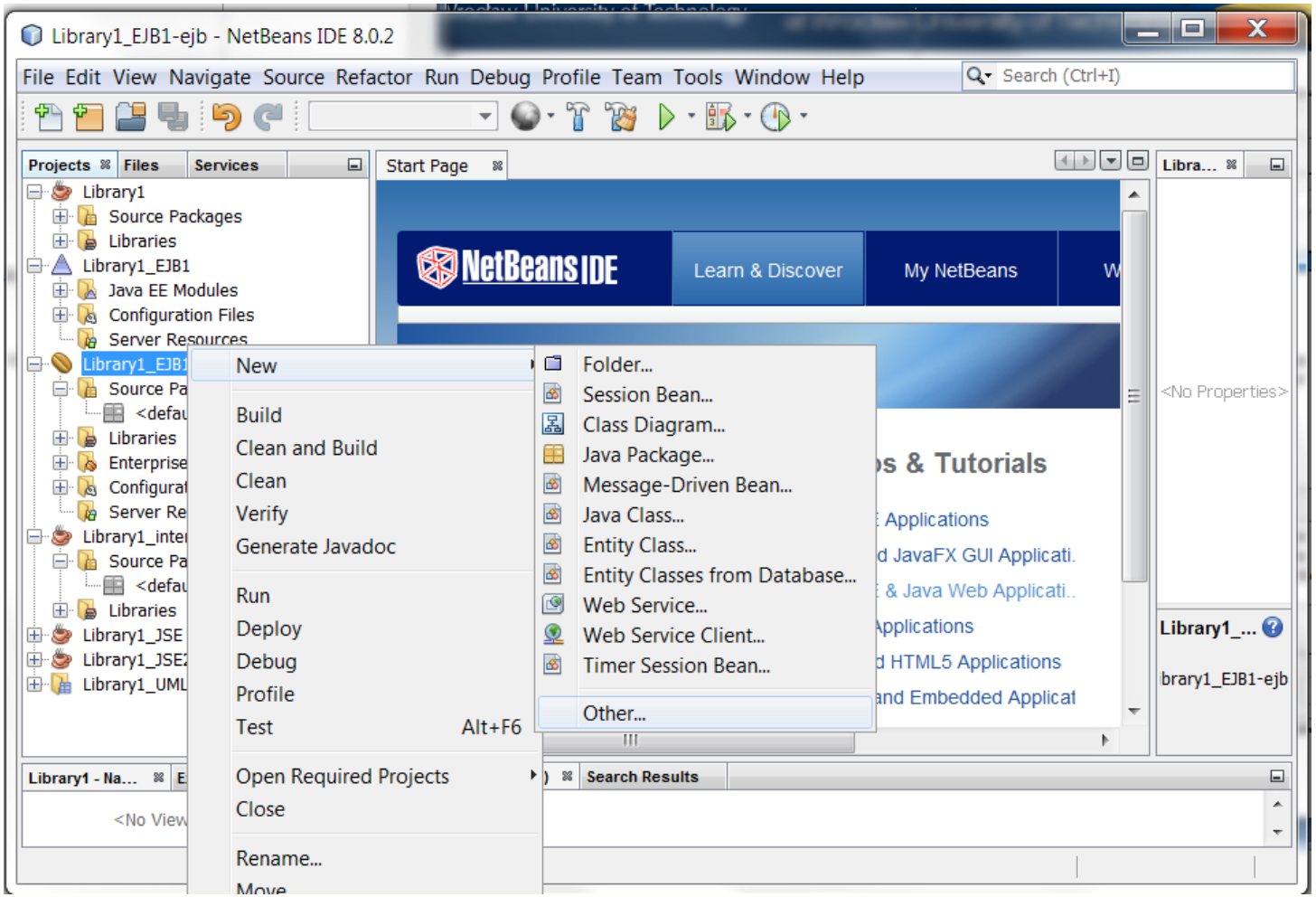
2.3. Creation the library project for interface of EJB – fill the Project Name and select the Project Location



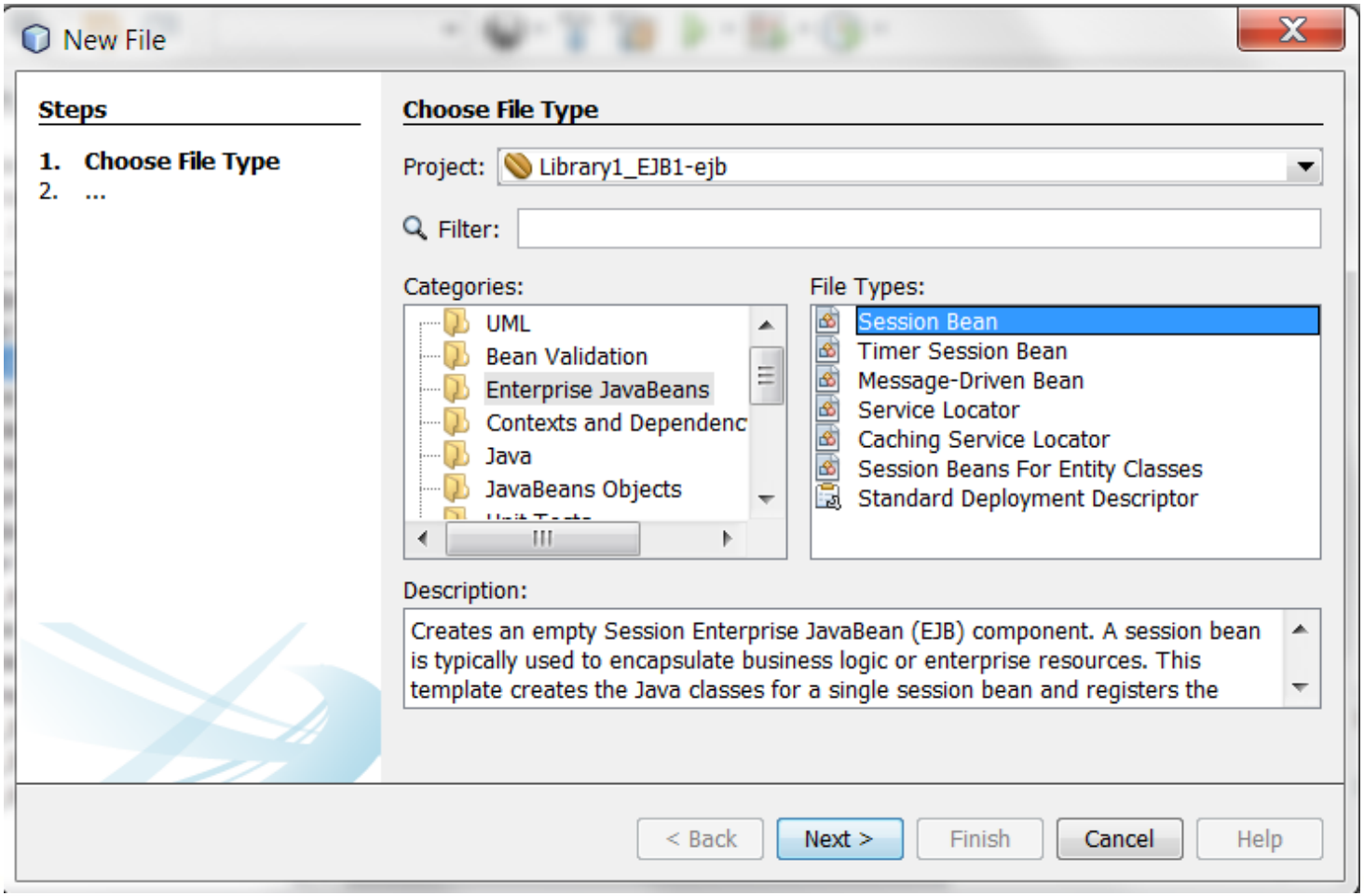
2.4. The new empty library project



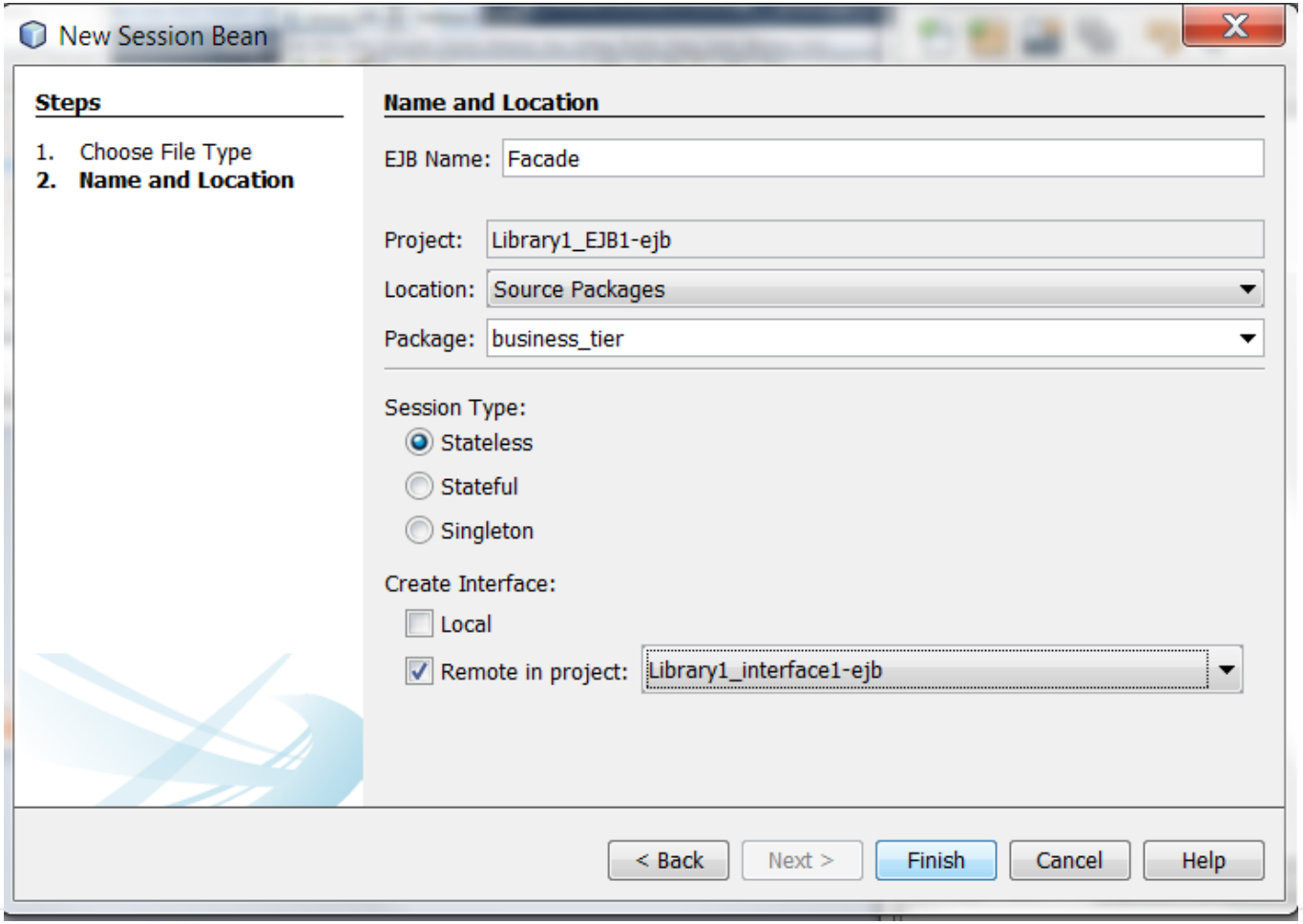
3. / 3.1. Adding the EJB as the Session Facade – Right click/New/Other



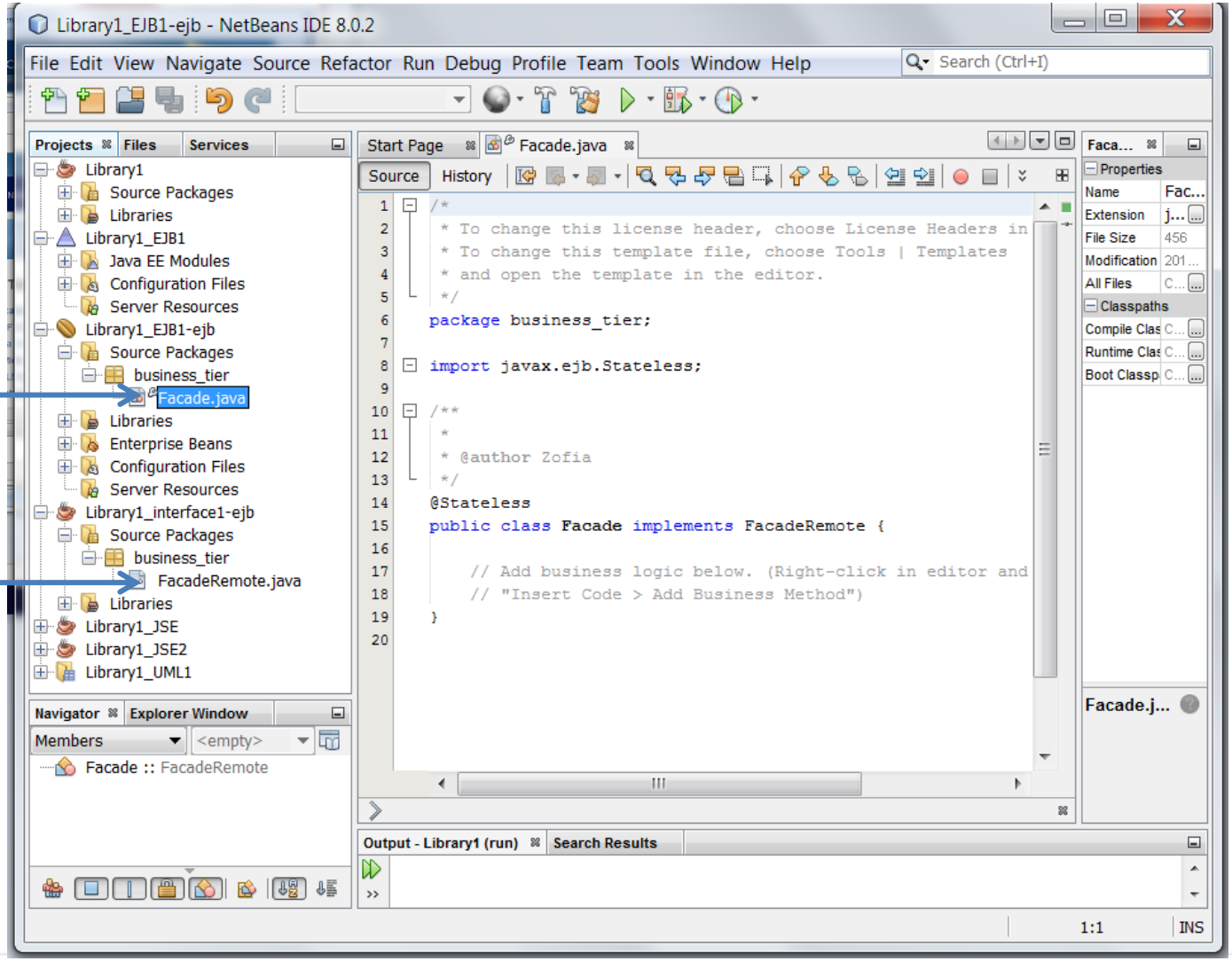
3.2. Adding the EJB as the Session Facade – select Enterprise JavaBeans/ Session Bean



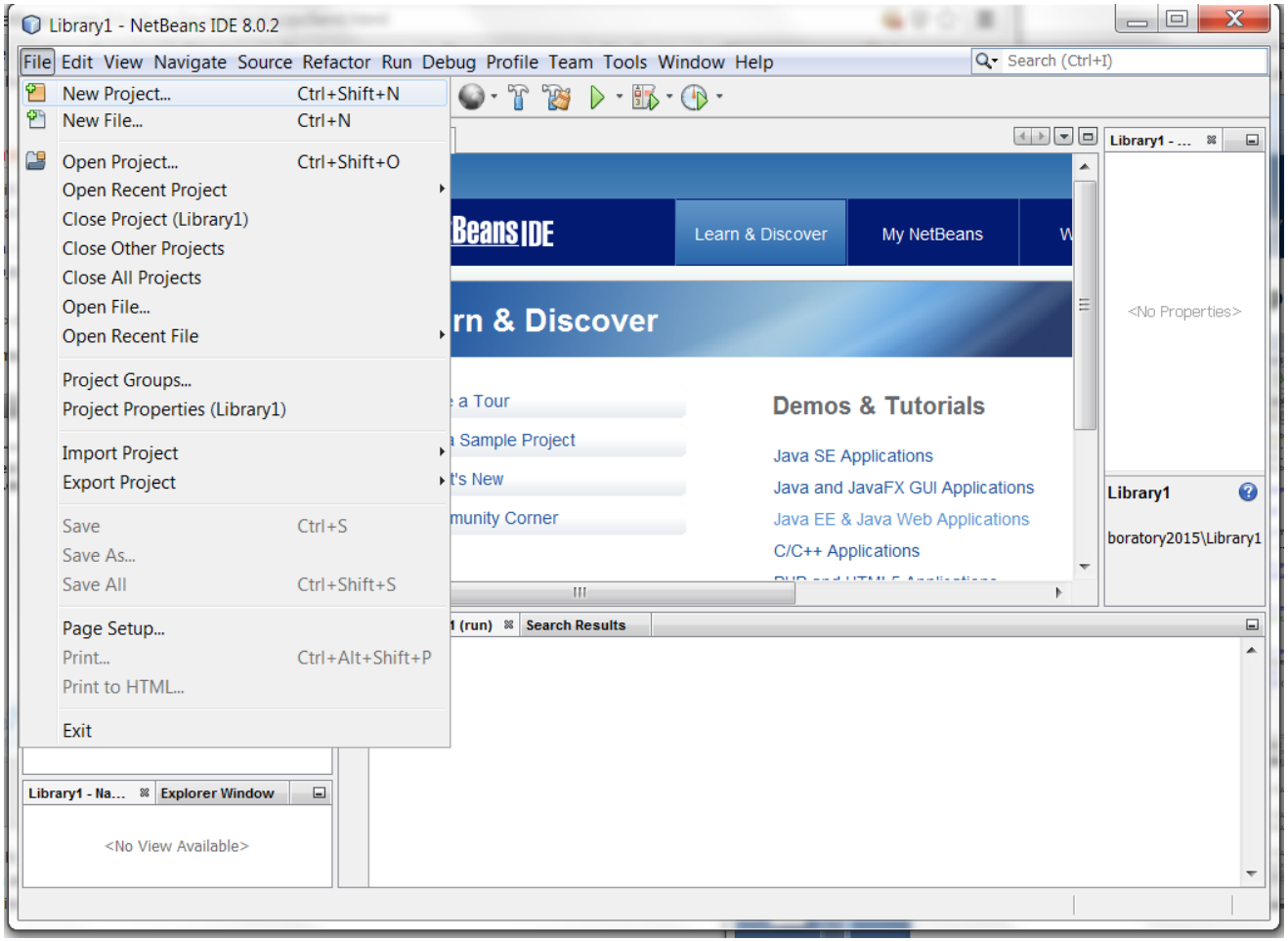
3.3. Adding the EJB as the Session Facade – fill the EJB Name, name of package, select the Session Type and create interafce for Remote EJB



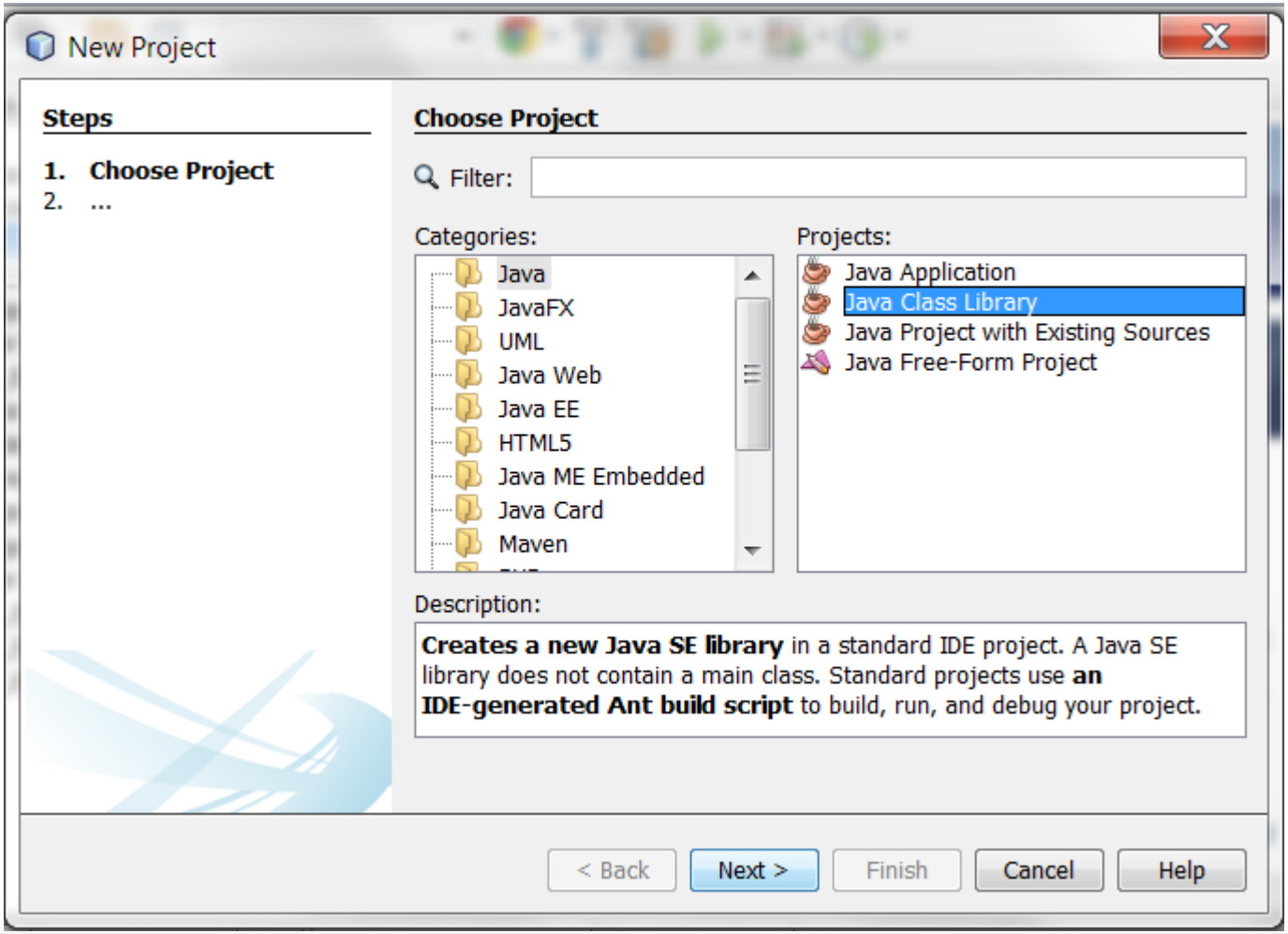
3.4. The new empty EJB and its empty interface



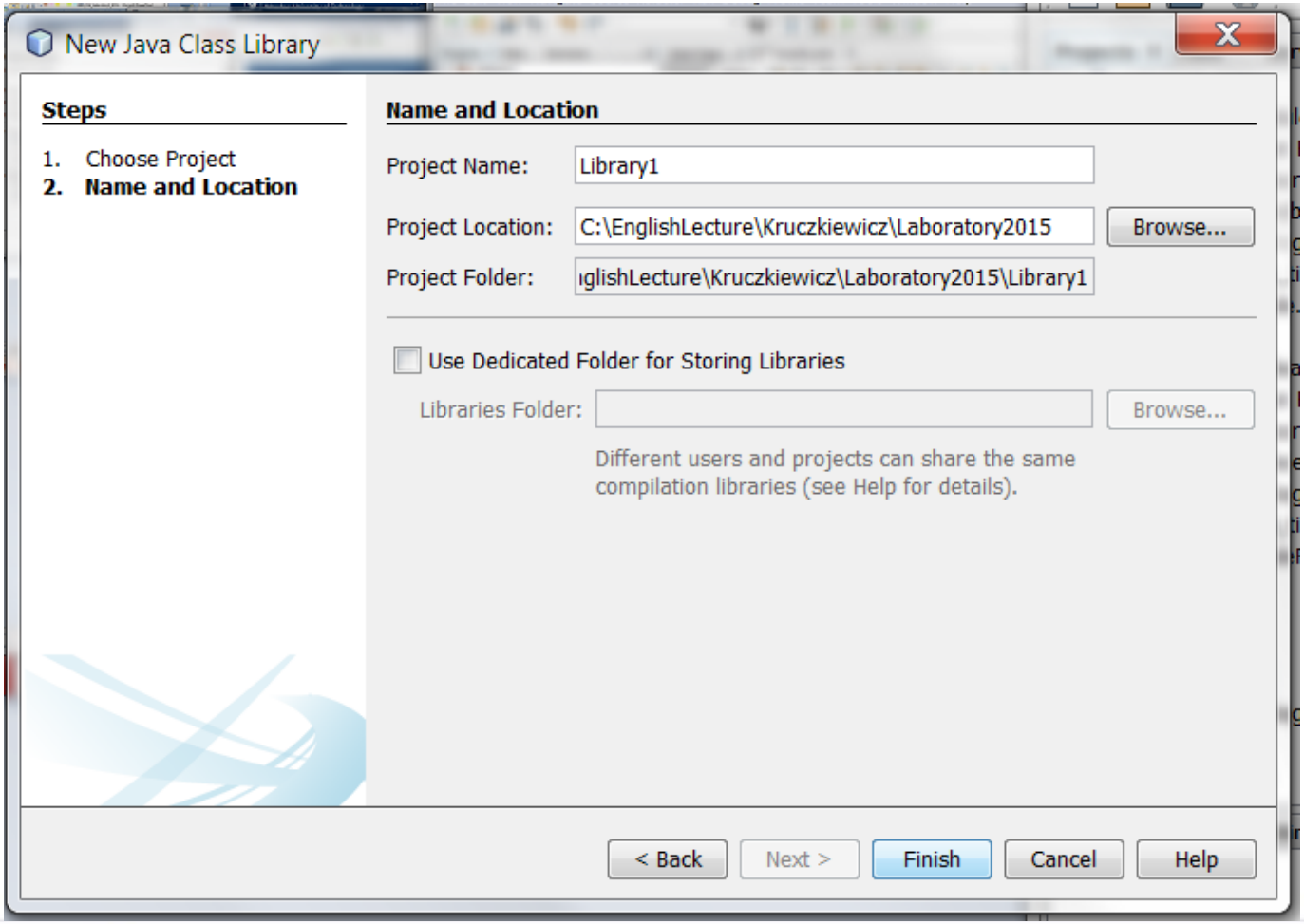
4. / 4.1./4.1.1. Creation the library project for code from lab1-2



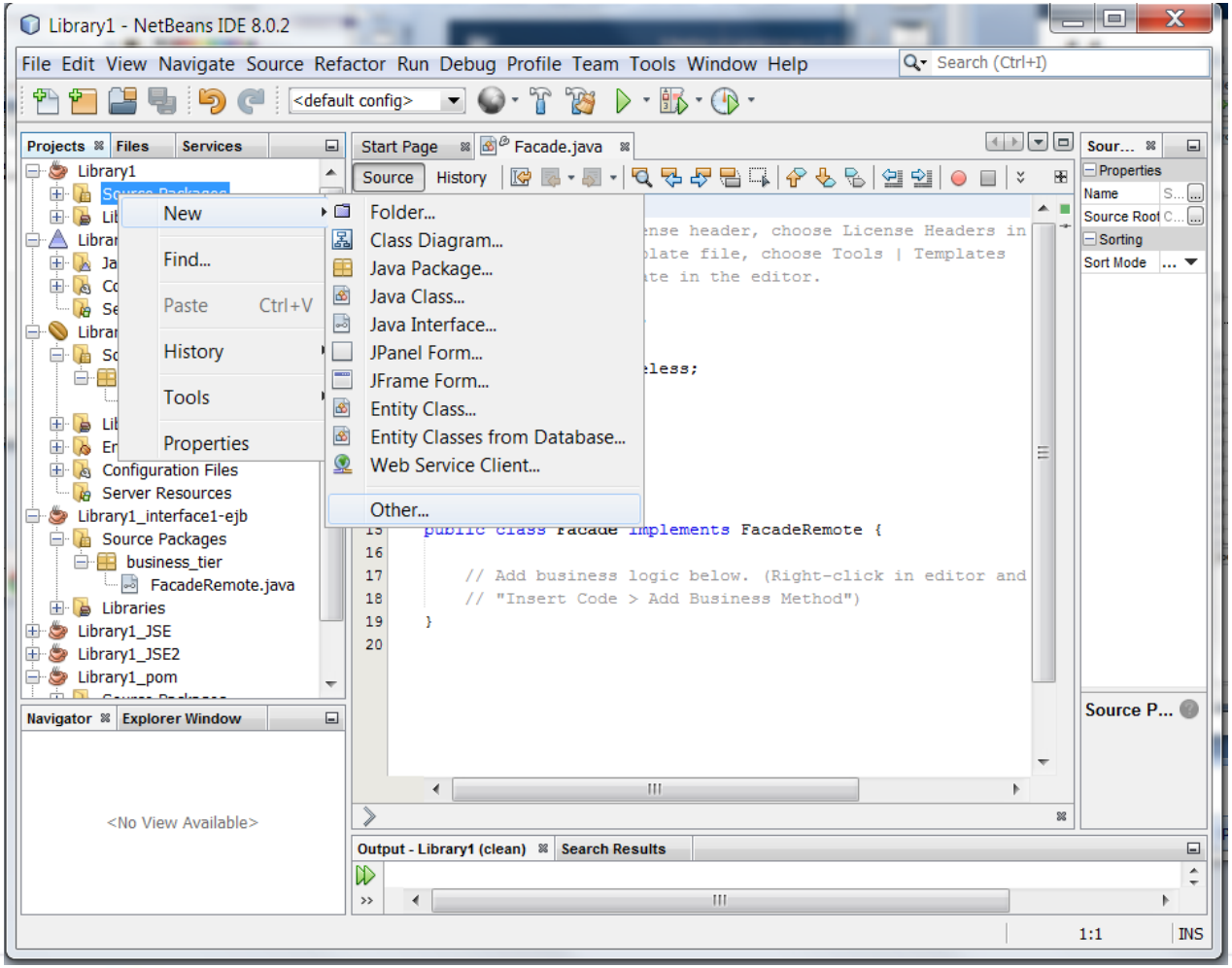
4.1.2. Creation the library project for code from lab1-2 - select Java/Java Class Library items



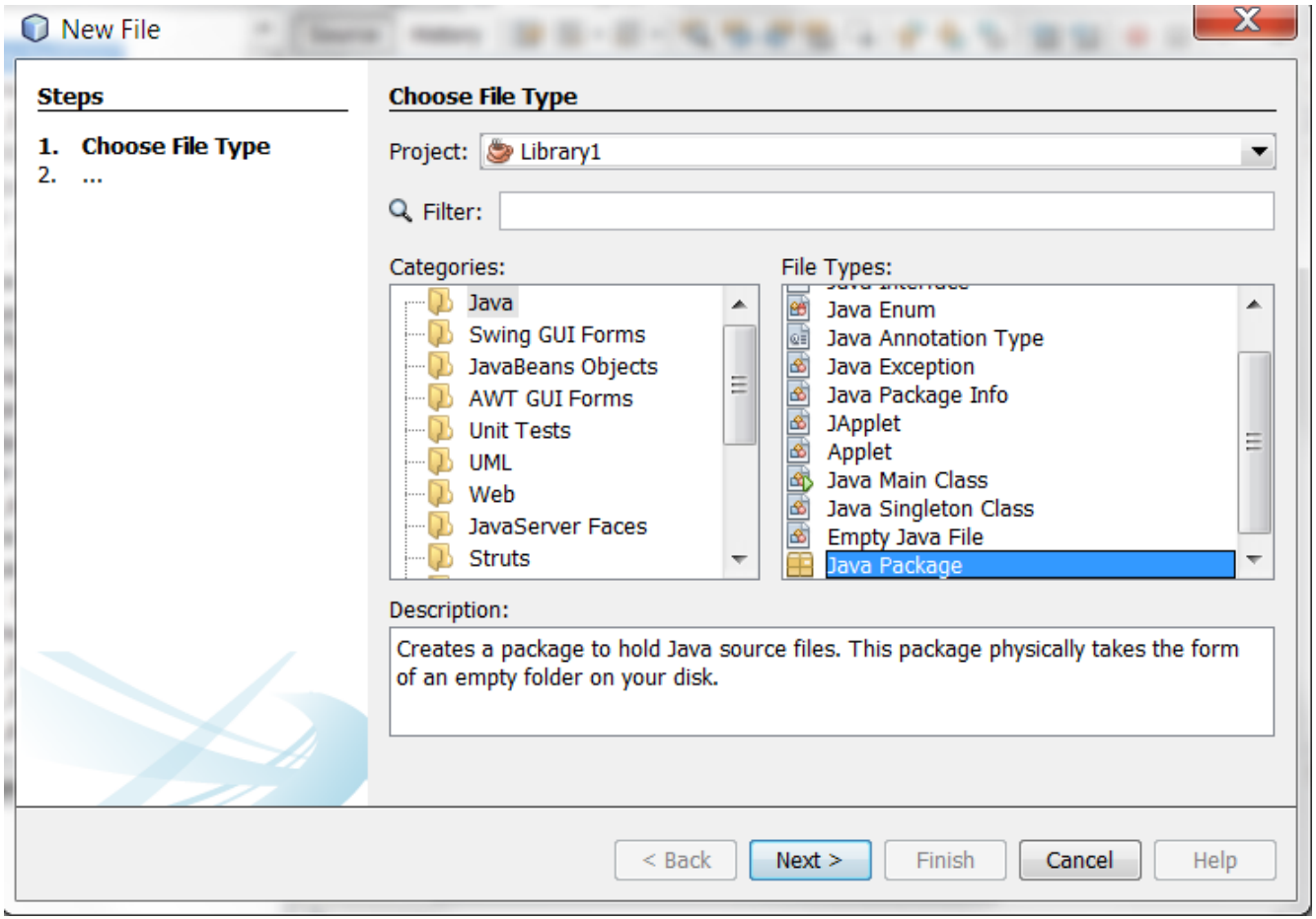
4.1.3. Creation the library project for code from lab1-2 - fill the Project Name and select the Project Location



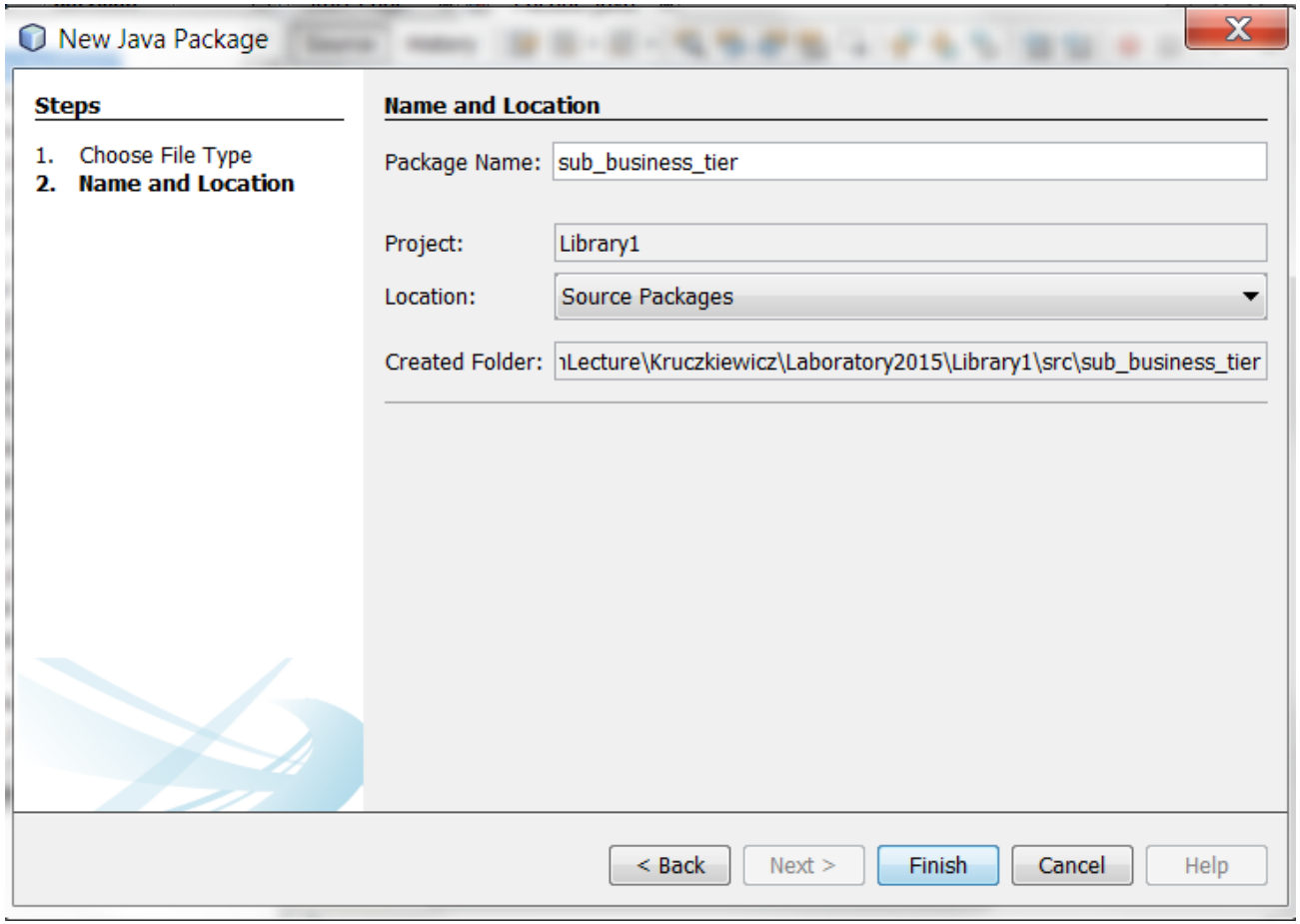
4.2. /4.2.1 Creation the package for TFacade and TFactory classes

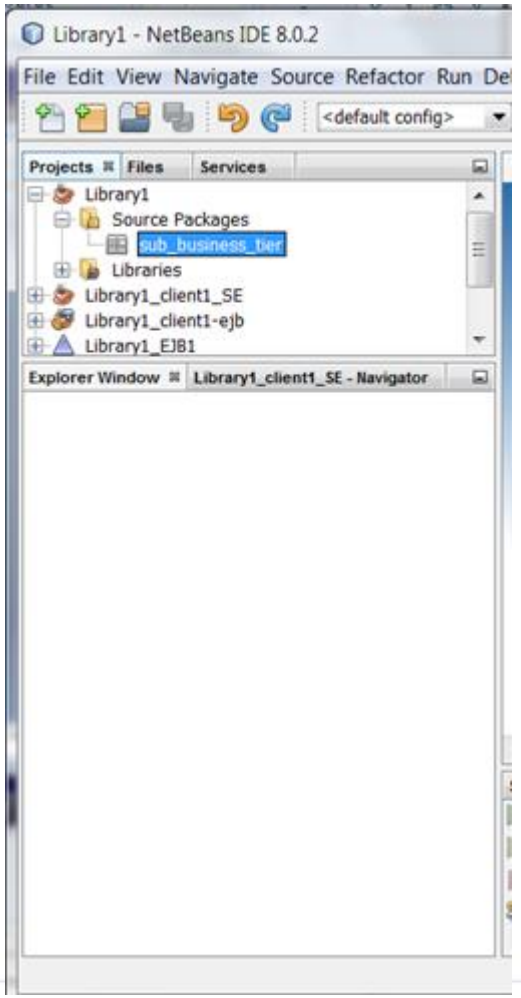


4.2.2. Creation the package for TFacade and TFactory classes – select Java/ Java Package

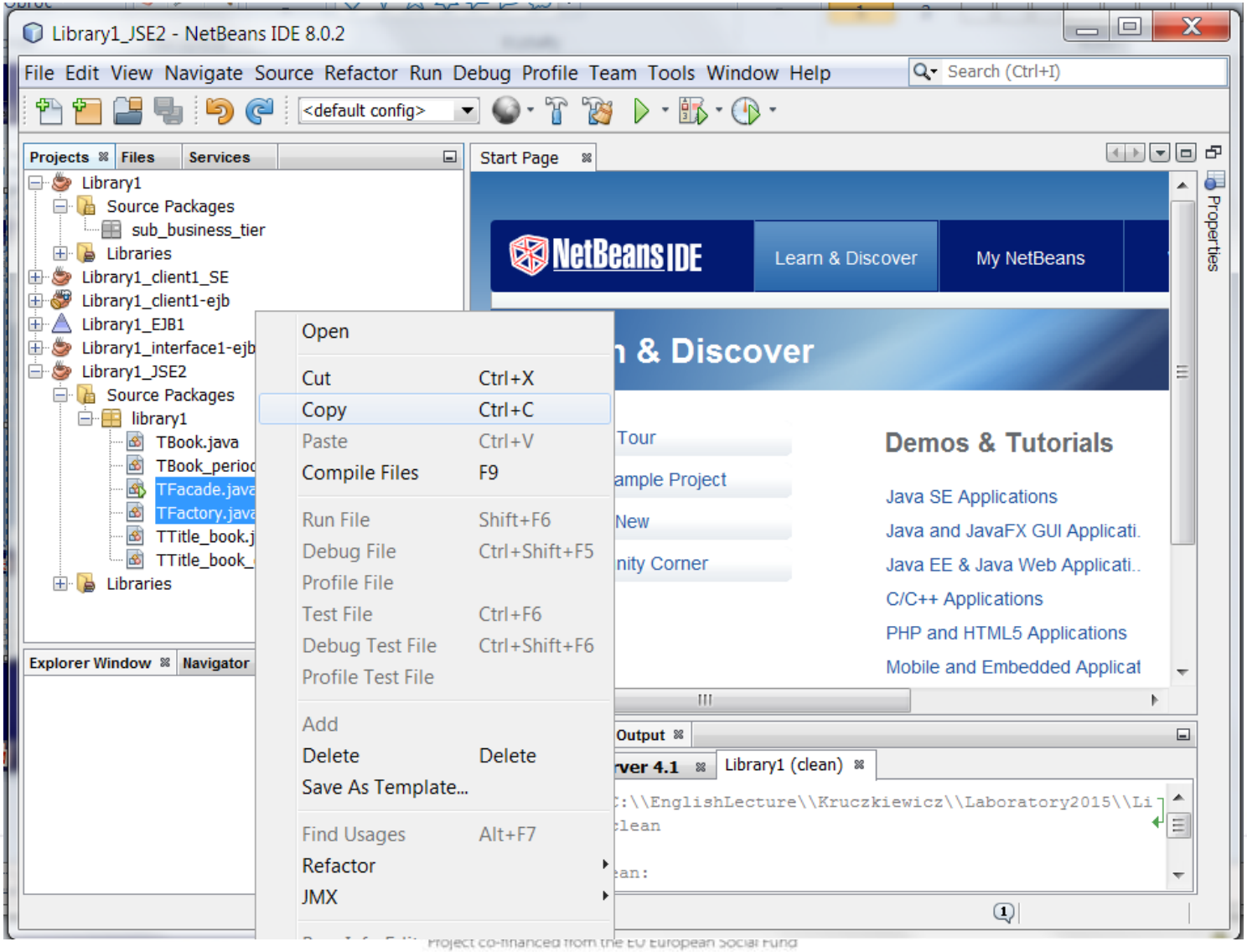


4.2.3. Creation the package for TFacade and TFactory classes – fill the name of package and select location . The result on the right.

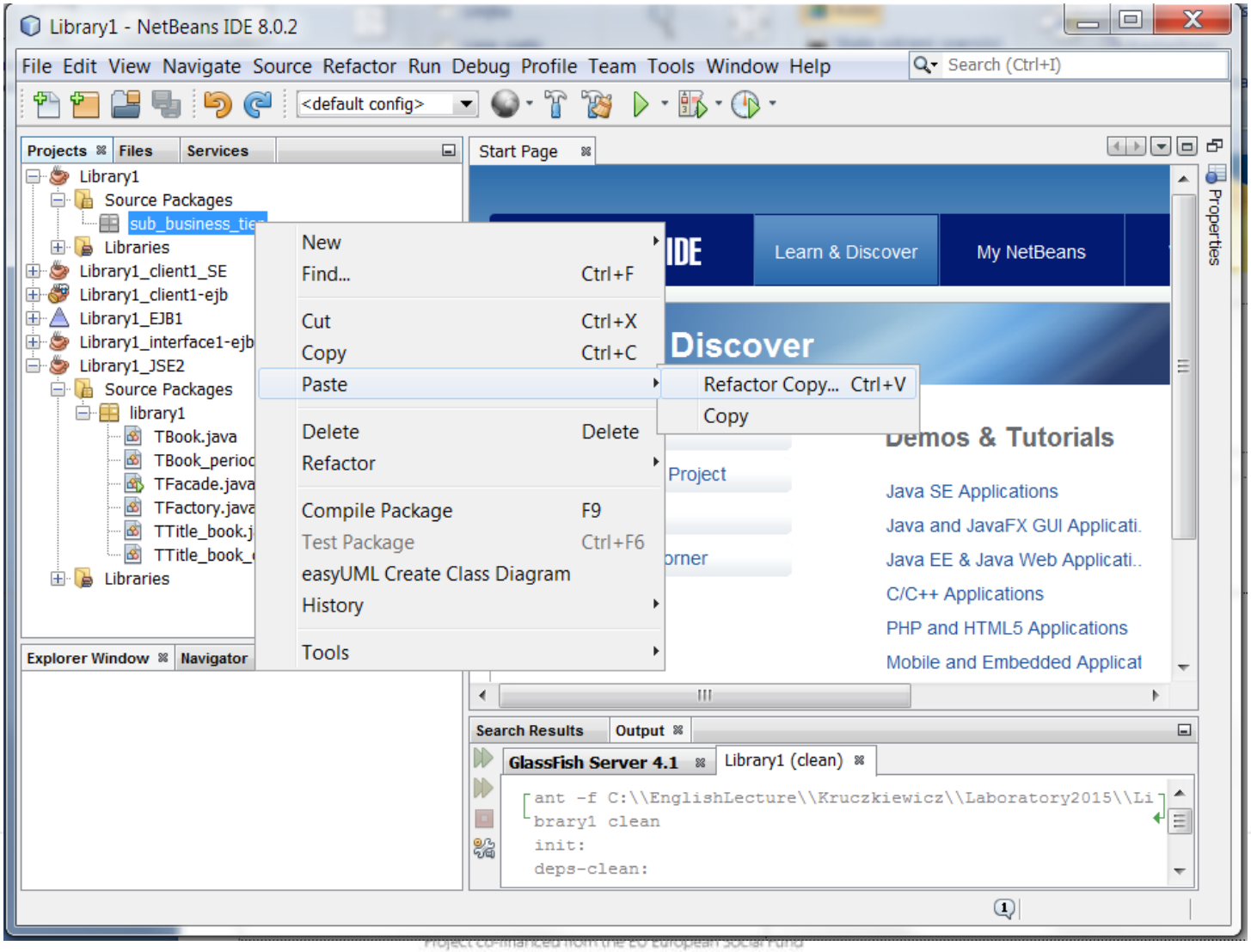




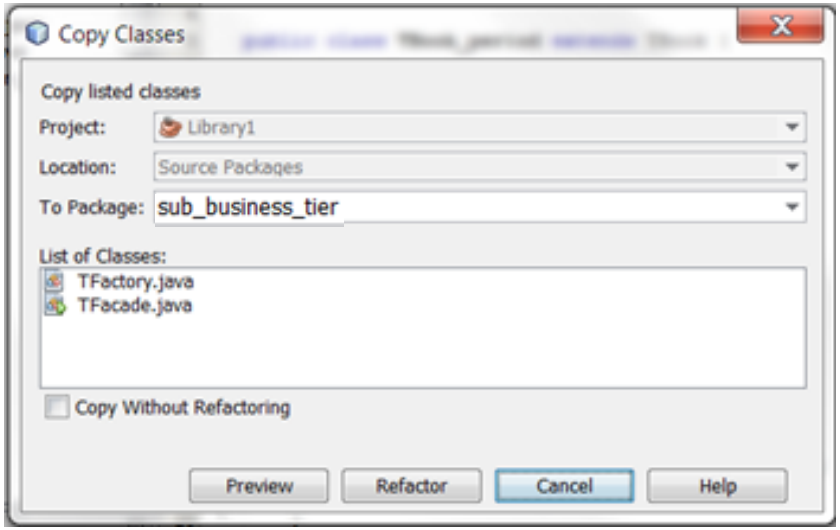
4.3. /4.3.1. Copying of TFacade and TFactory classes

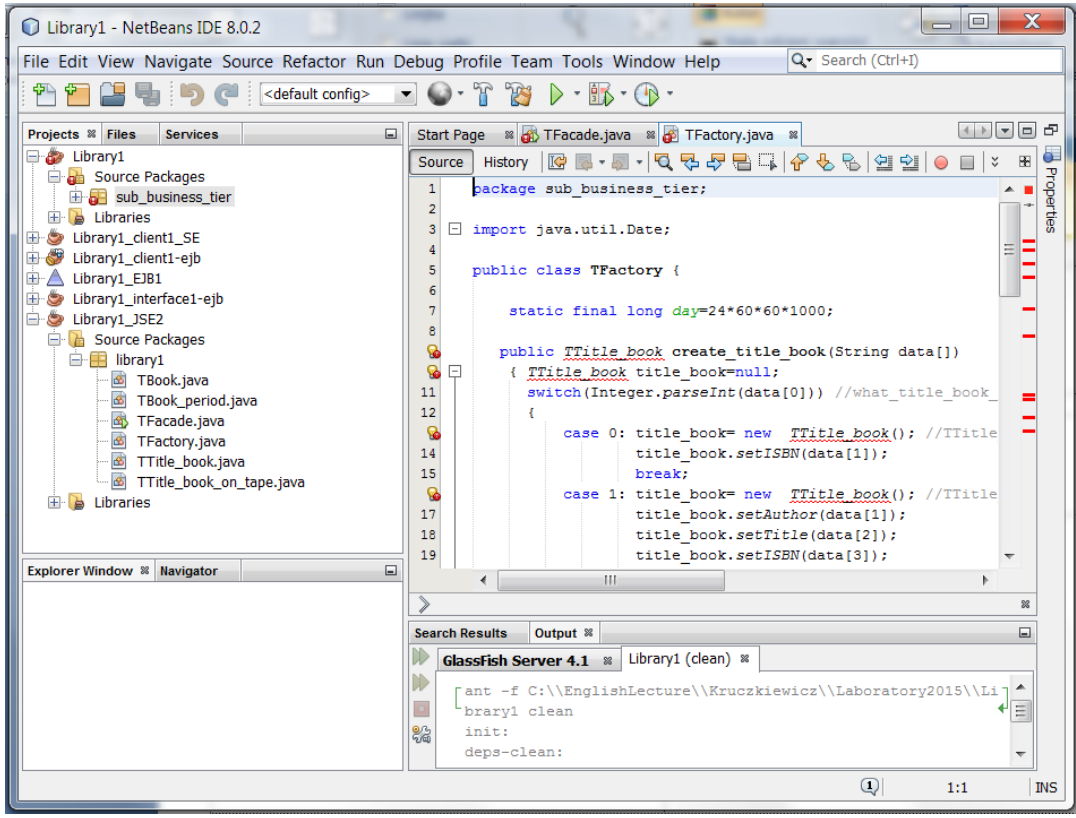


4.3.2. Paste the copies of the TFacade and TFactory classes from the lab1-2 project to the Library1 project (select the sub_business_tier package).

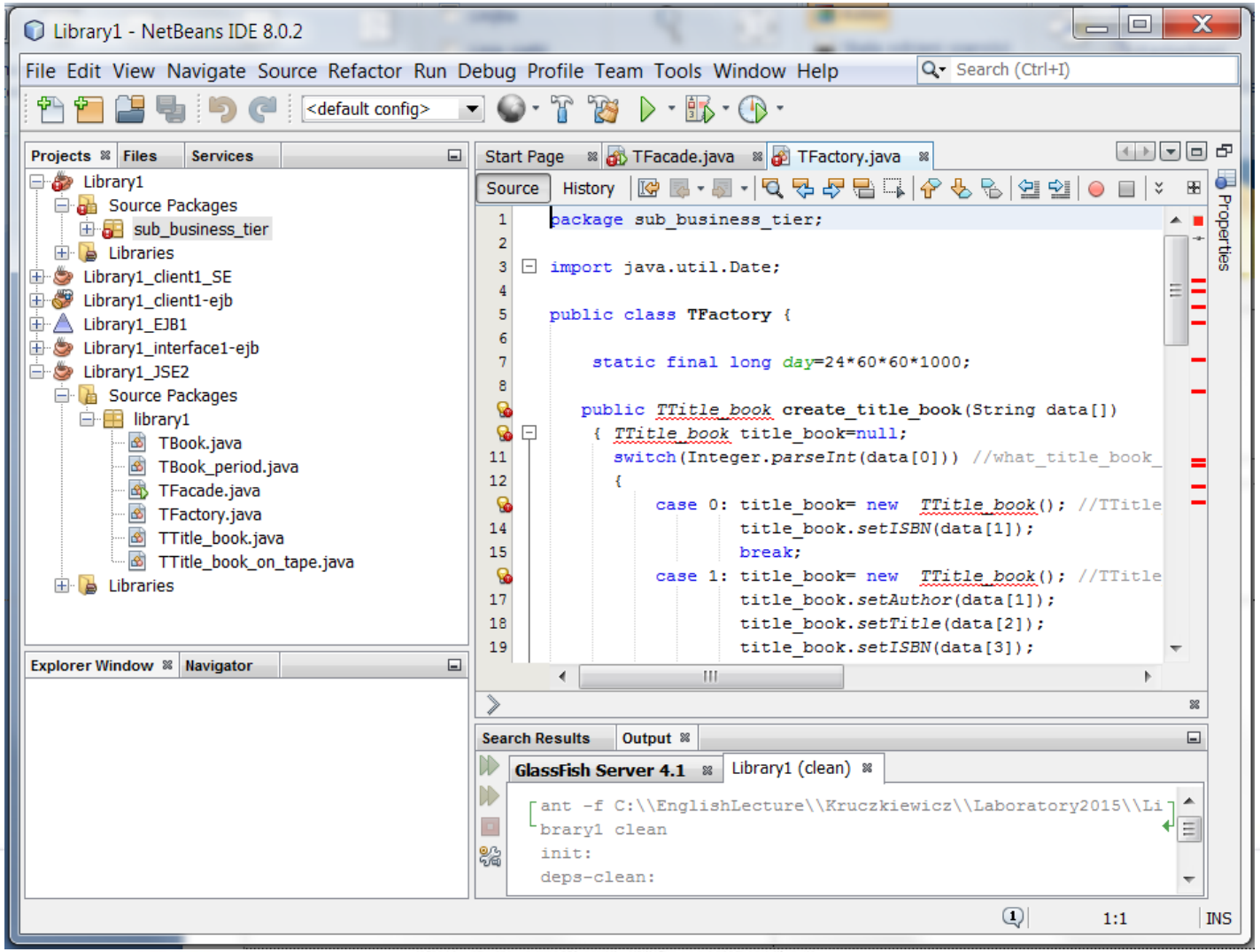


4.3.3. Paste the copies of the TFacade and TFactory classes from the lab1-2 project to the Library1 project – the refactoring process (click Refactor – on the left; the result on the right).

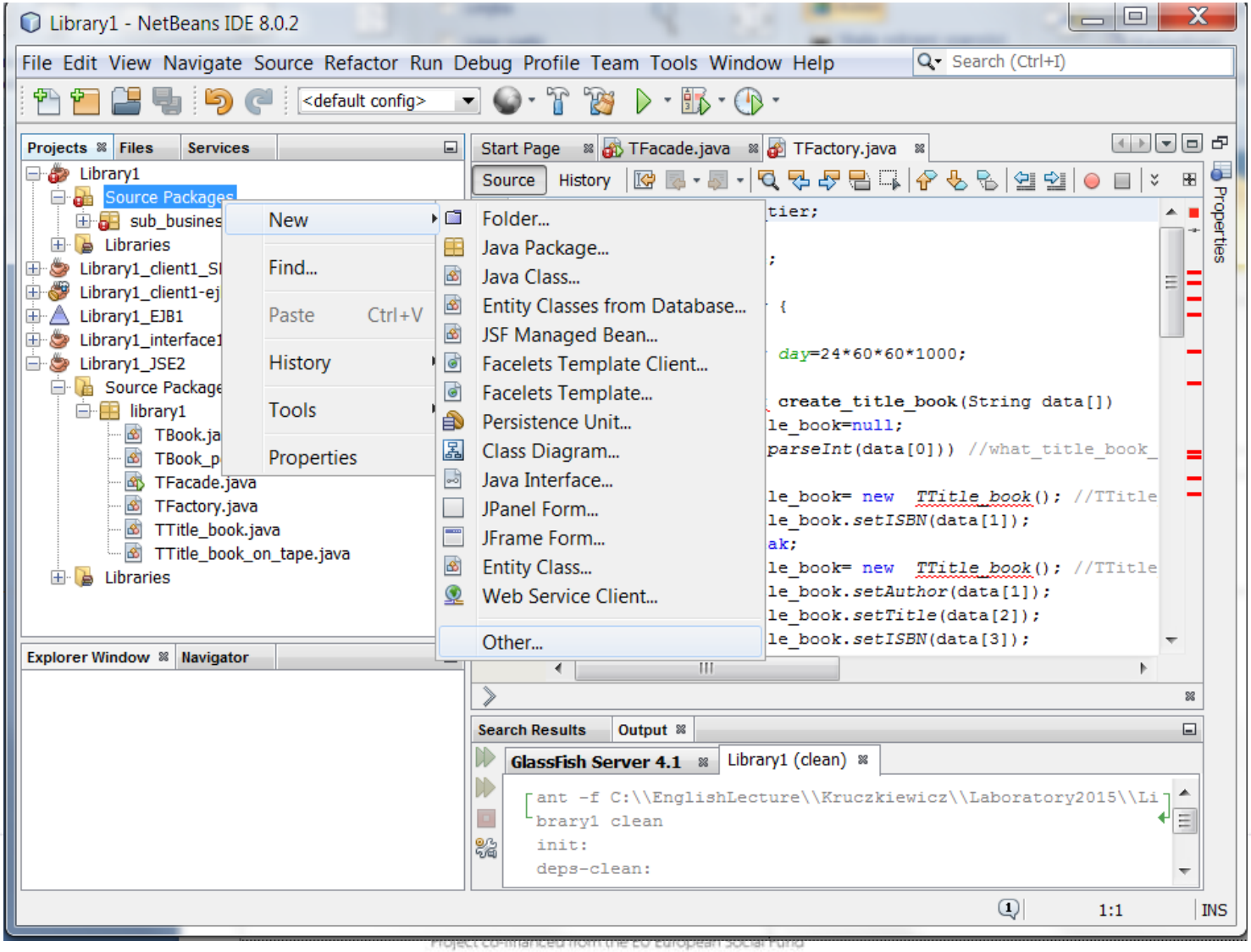




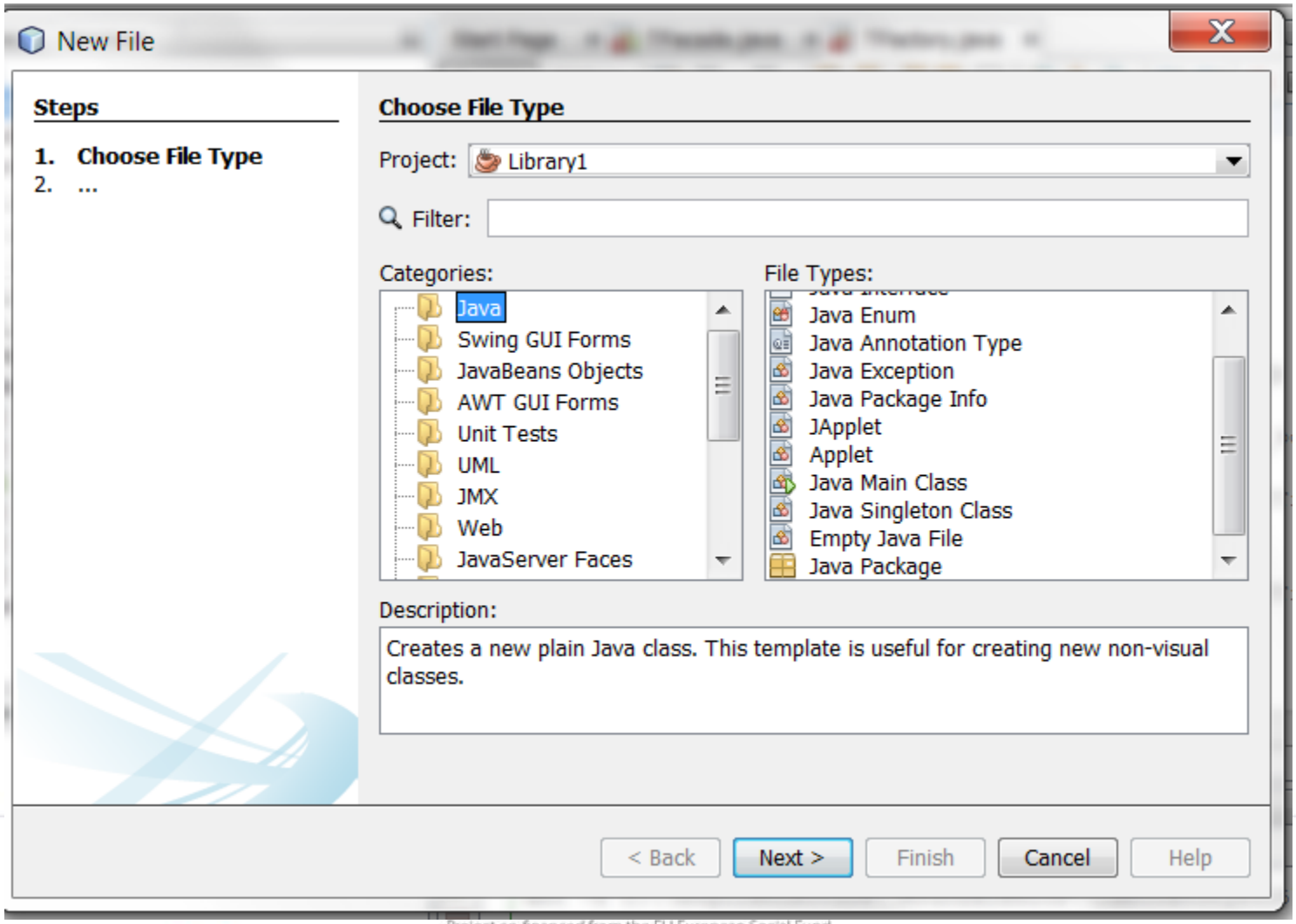
4.3.4. Paste the copies of the TFacade and TFactory classes from the lab1-2 project to the Library1 project – result



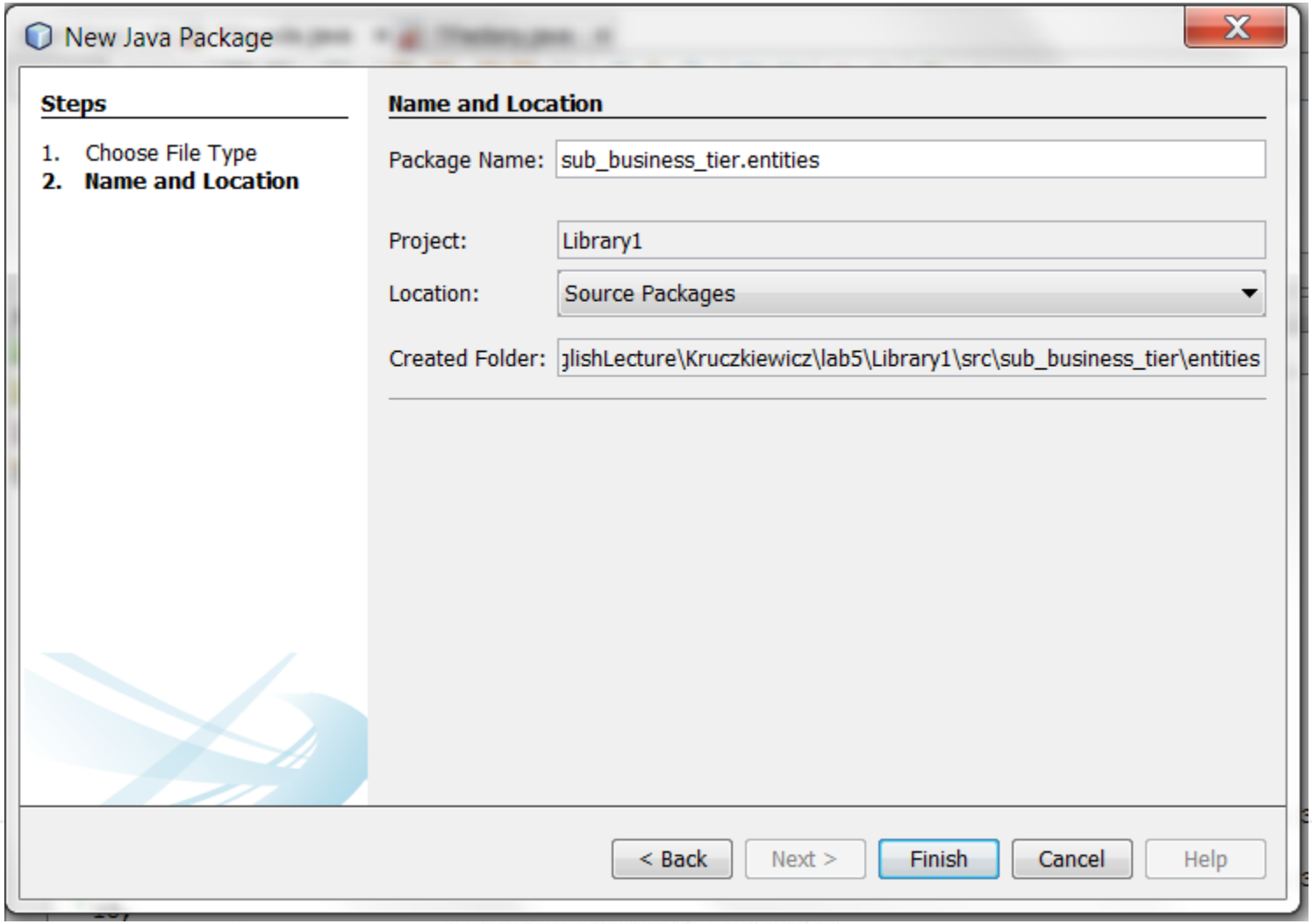
4.4. / 4.4.1. Creation of the new package for data classes from lab1-2



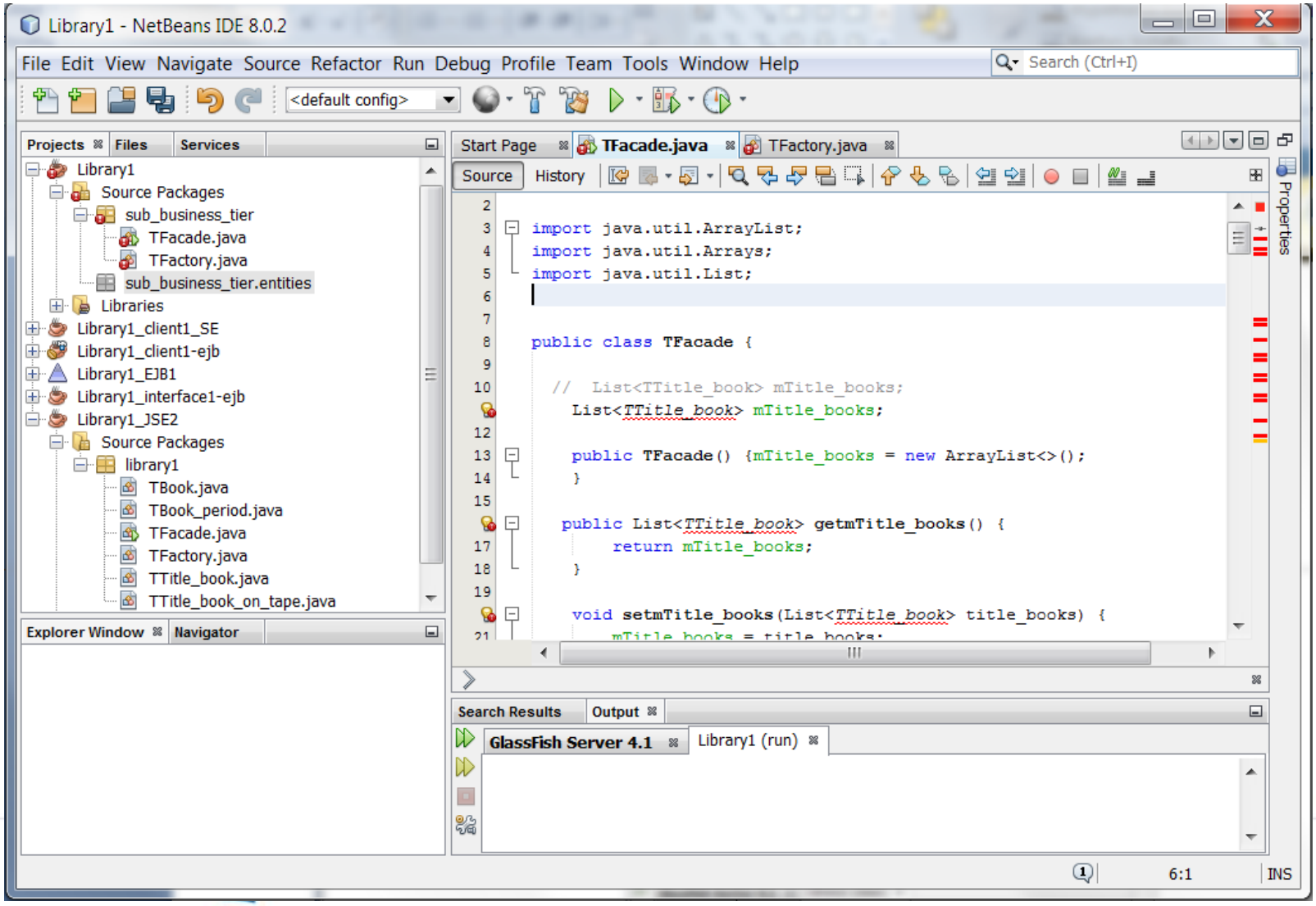
4.4.2. Creation of new package for data classes from lab1-2 – select Java/ Java Package



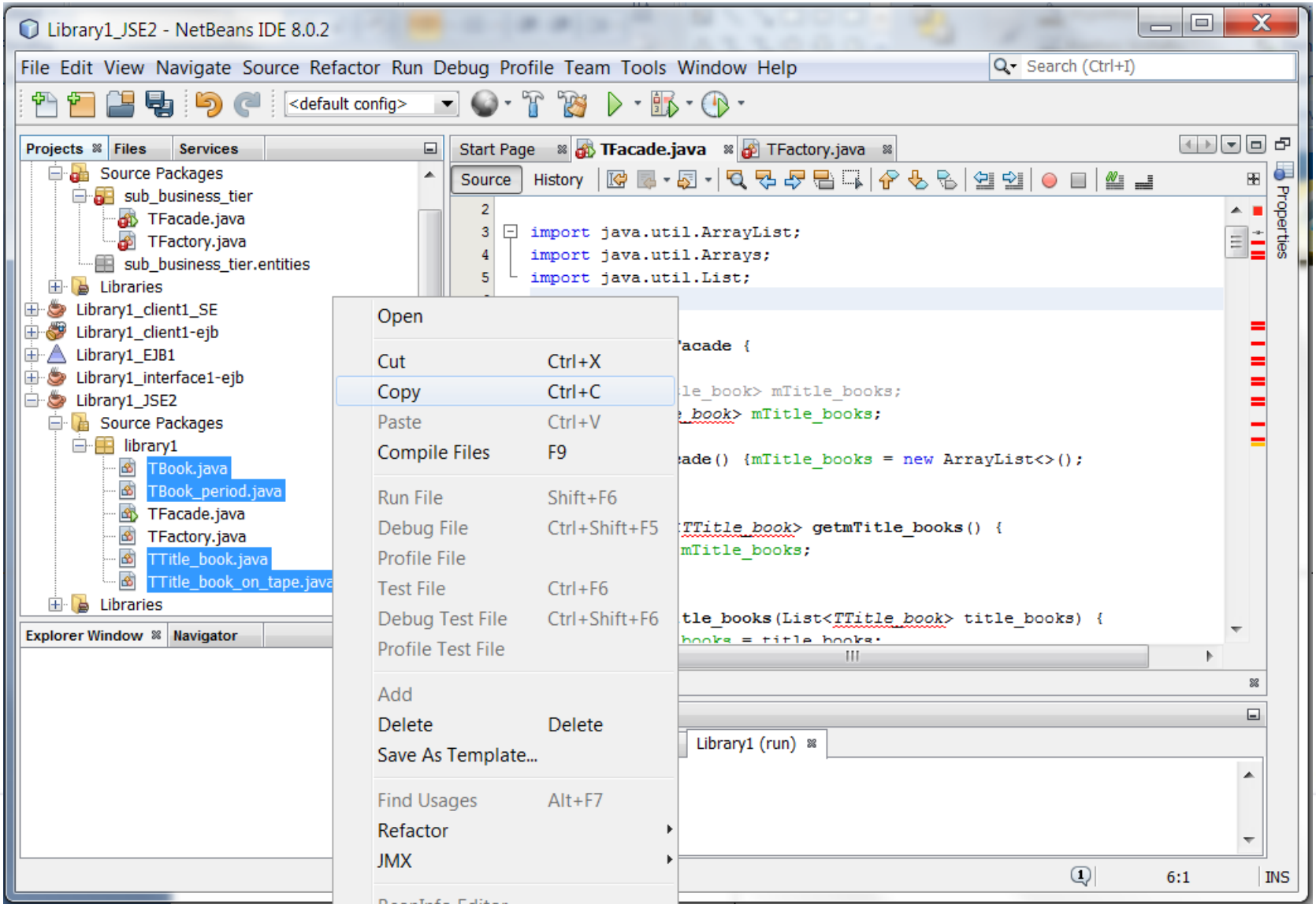
4.4.3. Creation of new package for data classes from lab1-2 – fill the name of package and select location



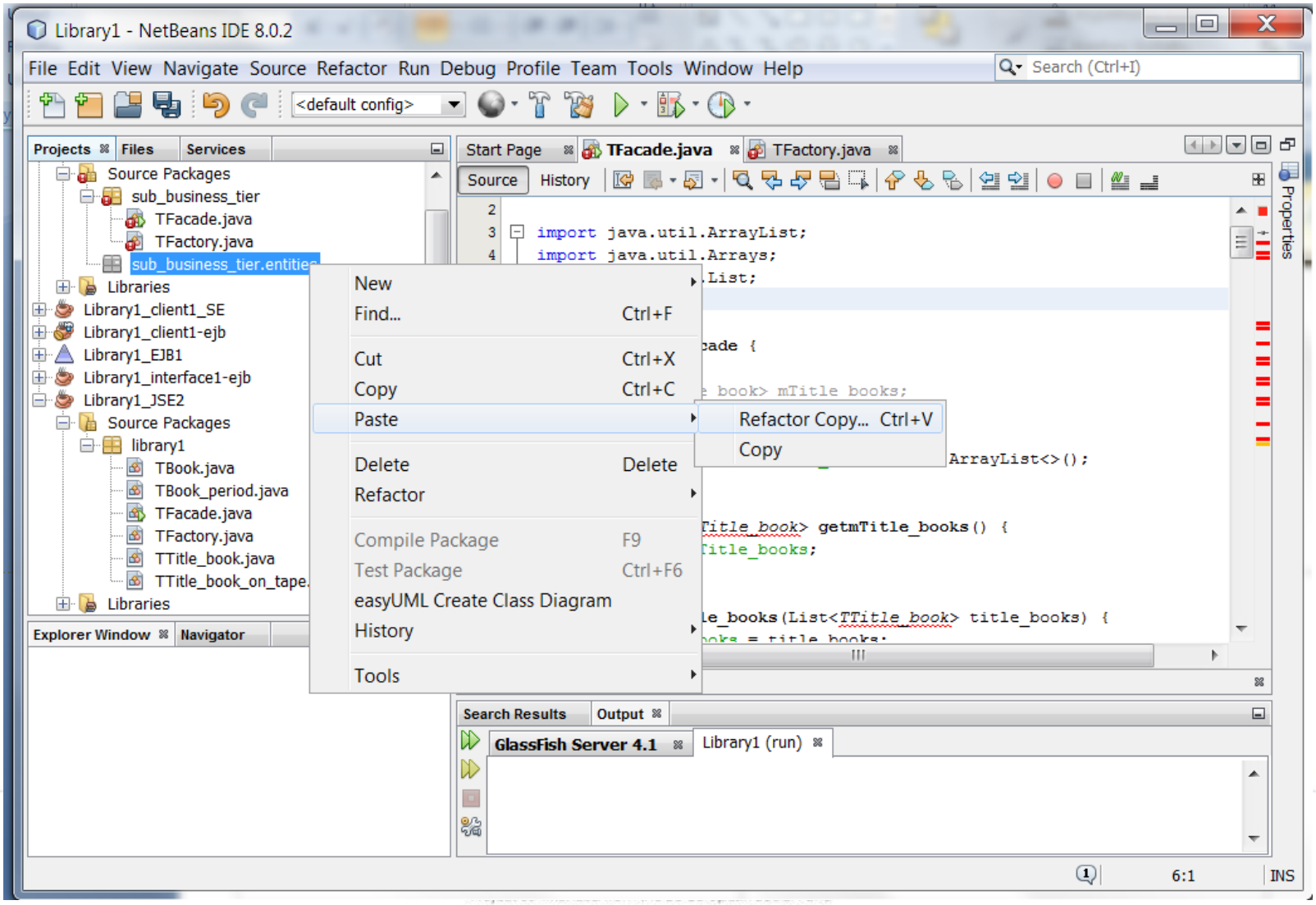
4.4. Result of creating of the sub_business_tier.entities packages



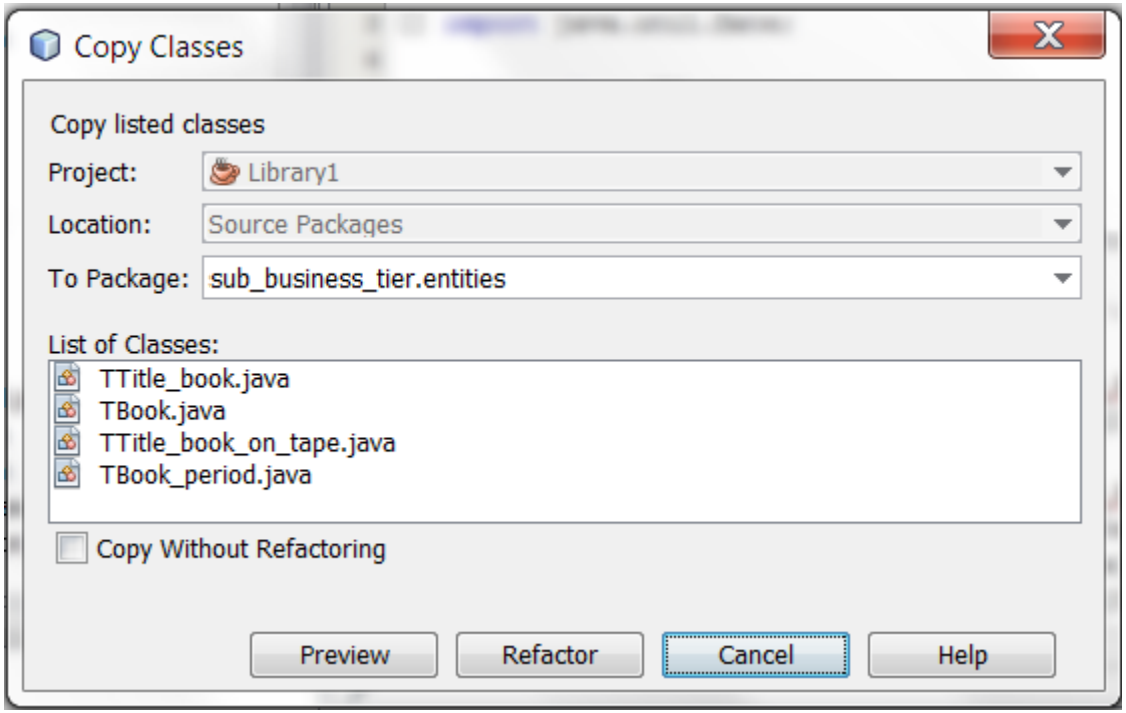
4.5. /4.5.1. Copying of the TTitle_book and other classes as data classes from lab1-2 project



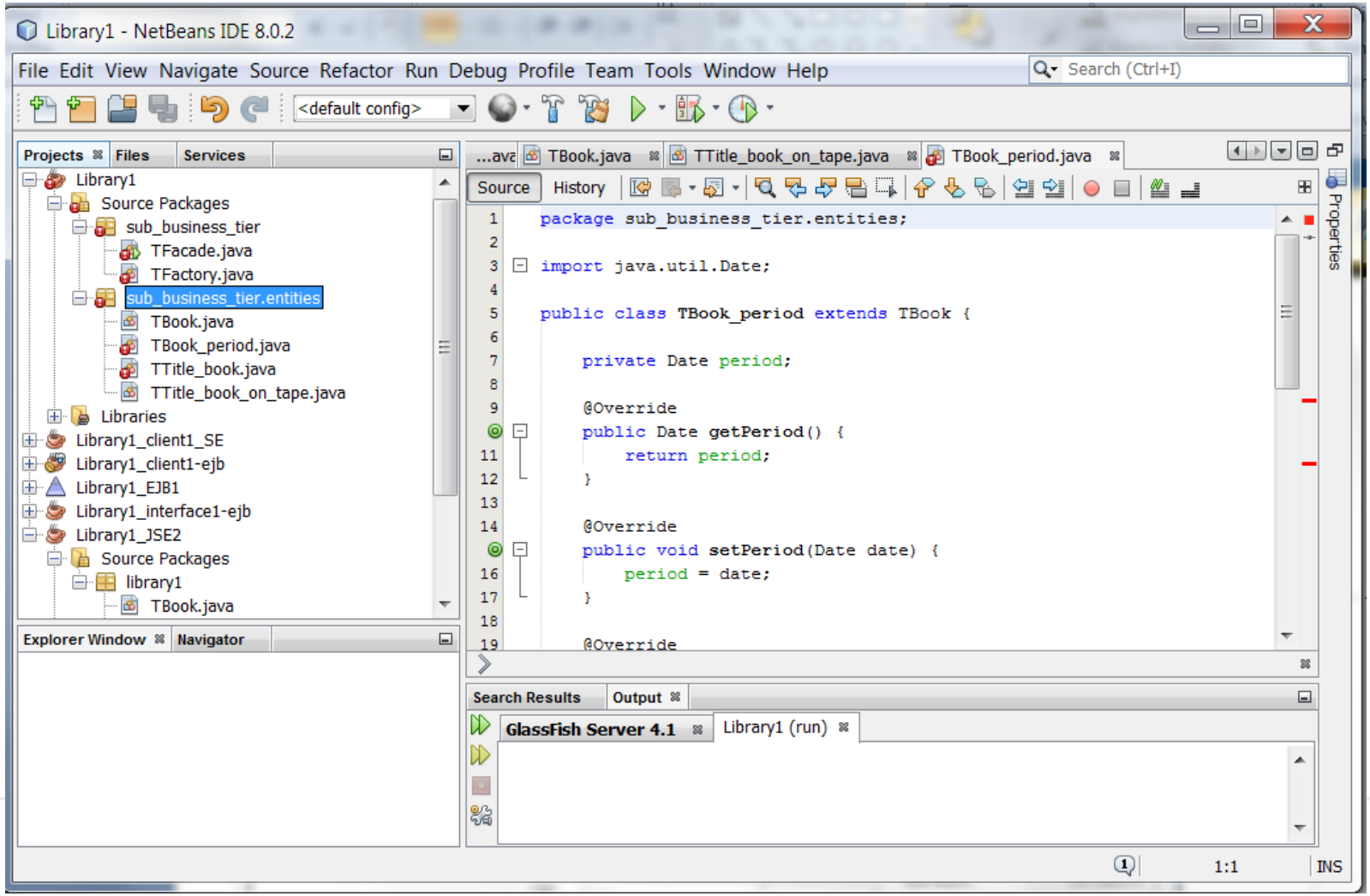
4.5.2 Paste the copies of the TTitle_book and other classes as data classes from the lab1-2 project to the Library1 project (select the sub_business_tier.entities package).



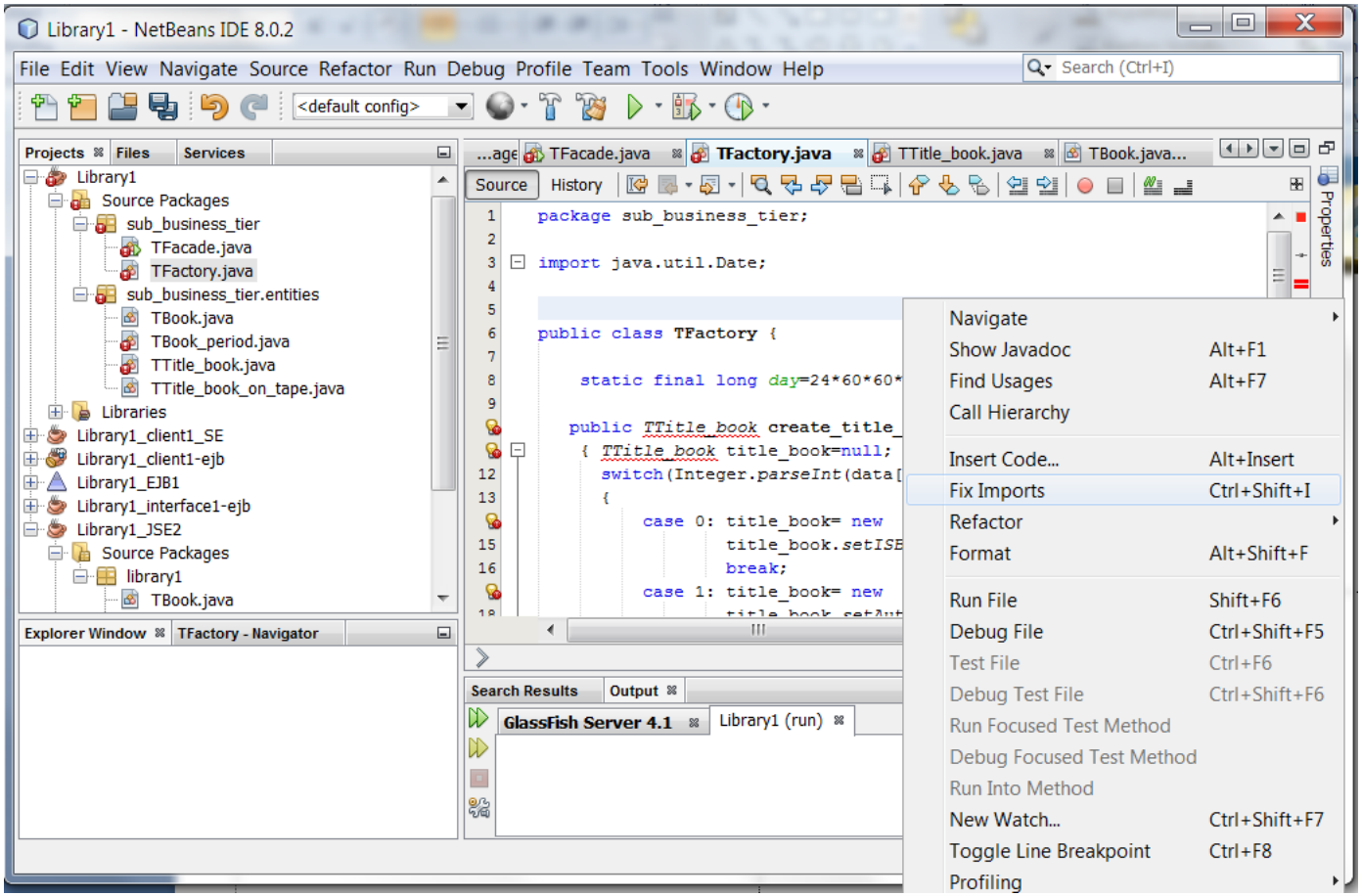
4.5.3. Paste the copies of the TTitle_book and other classes as data classes from the lab1-2 project to the Library1 project – the refactoring process (click the Refactor button).

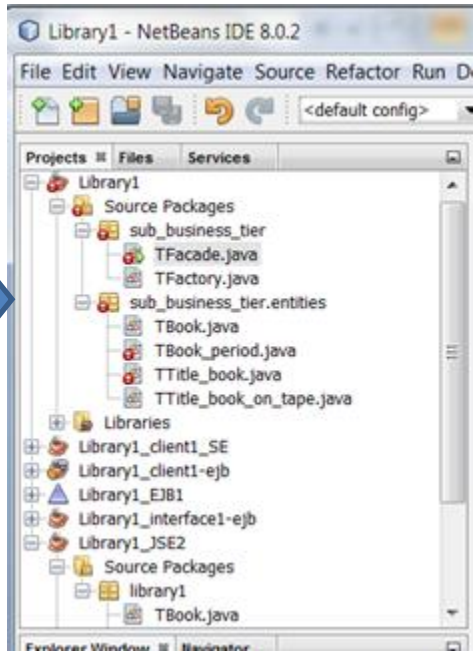


4.5.4. Paste the copies of the TTitle_book and other classes as data classes from the lab1-2 project to the Library1 project - result

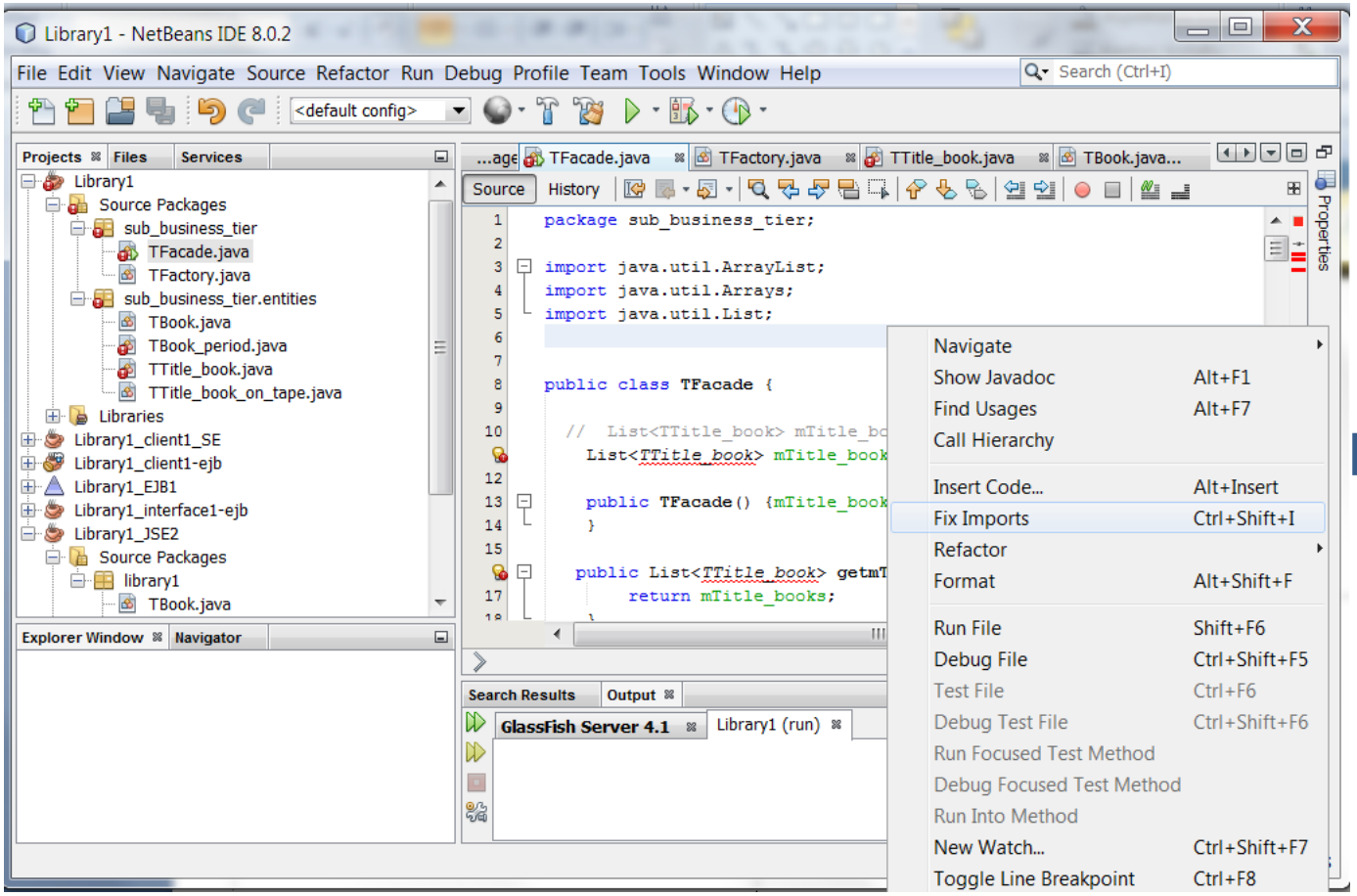


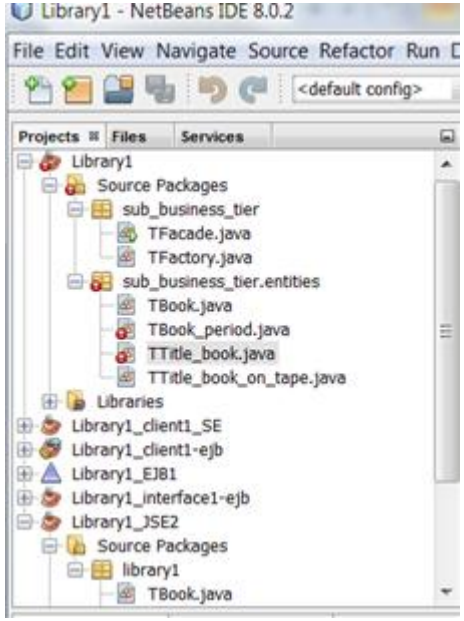
4.6. /4.6.1. „Fix Imports” process for adding the lacking imports of classes – the TFactory class



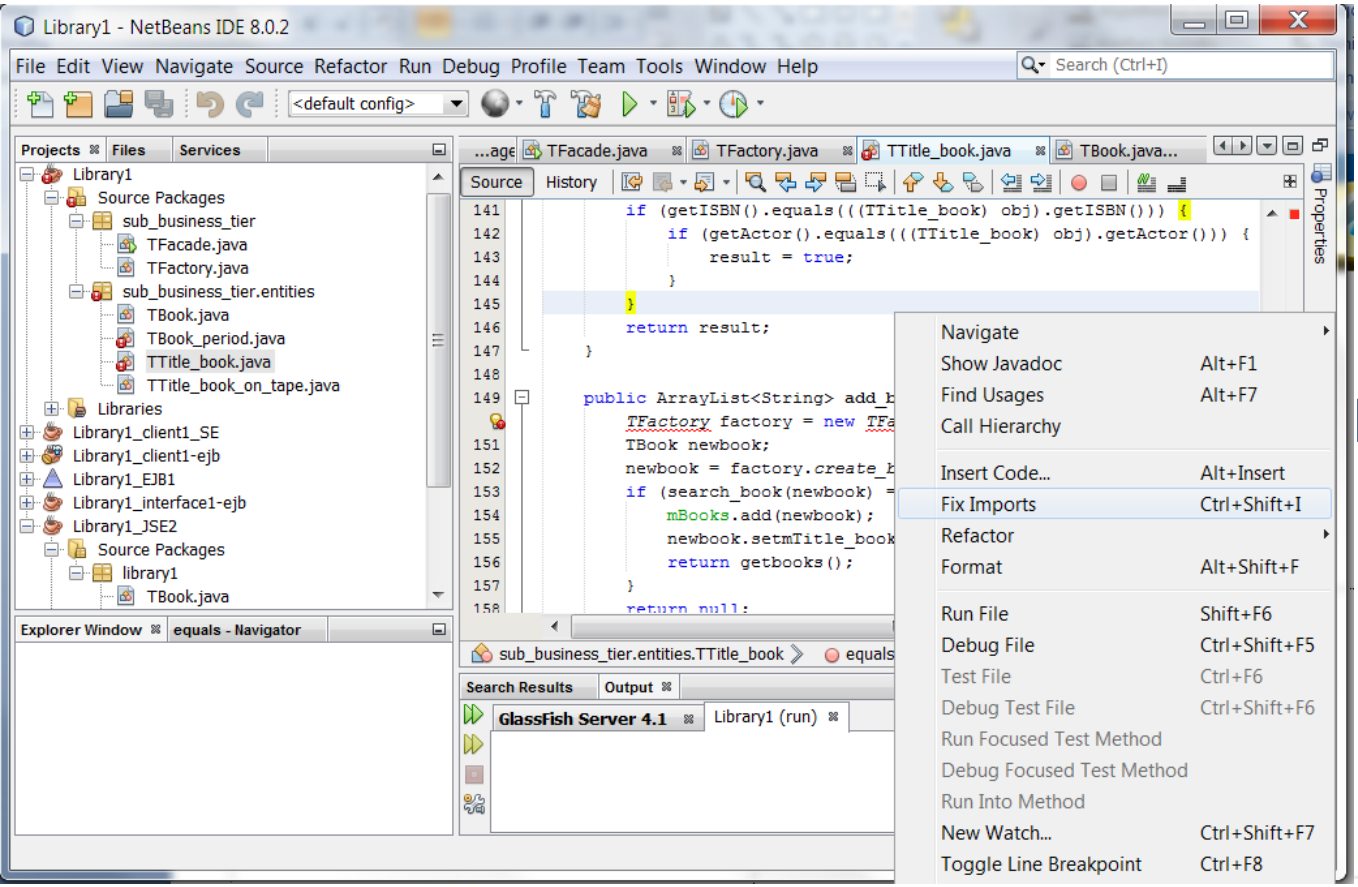


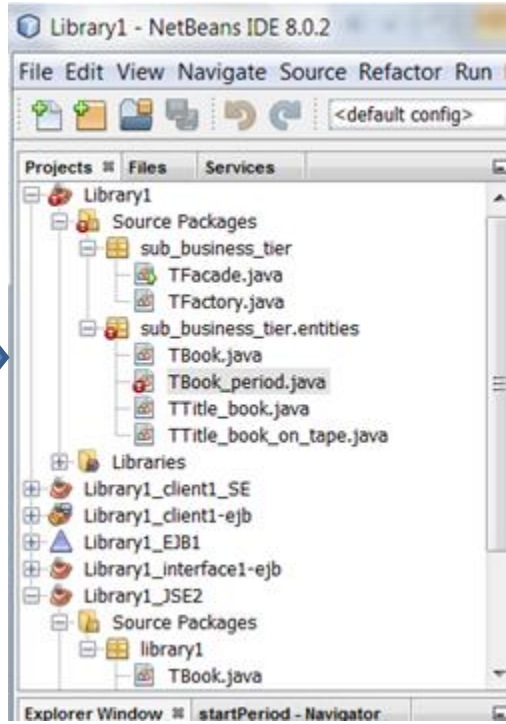
4.6.2. „Fix Imports” process for adding the lacking imports of classes – the TFacade class



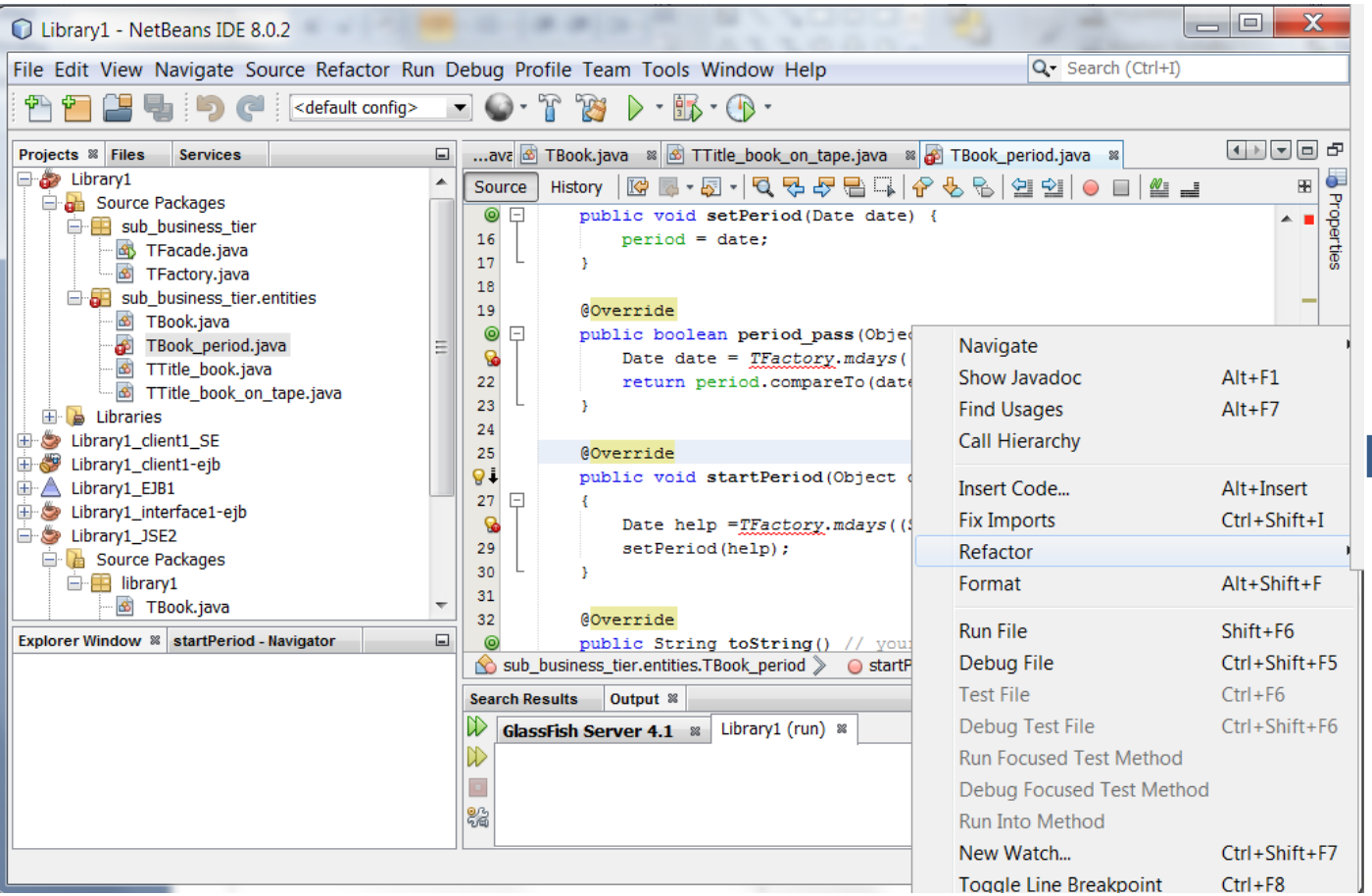


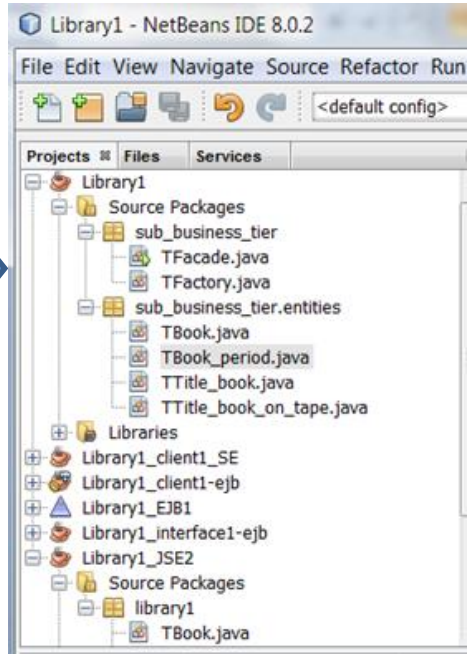
4.6.3. „Fix Imports” process for adding the lacking imports of classes – the TTitle_book class



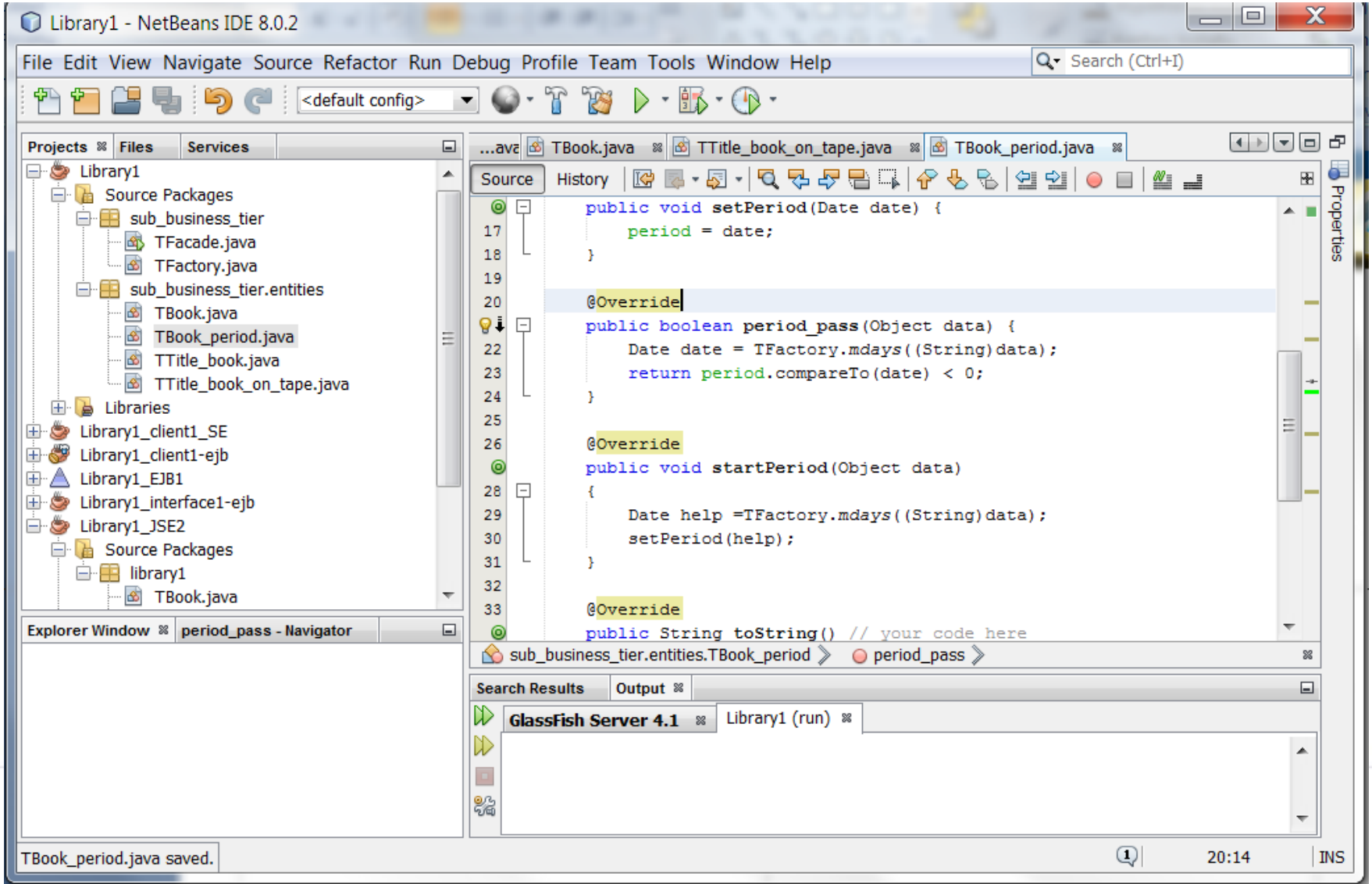


4.6.4. „Fix Imports” process for adding the lacking imports of classes – the TBook_period class

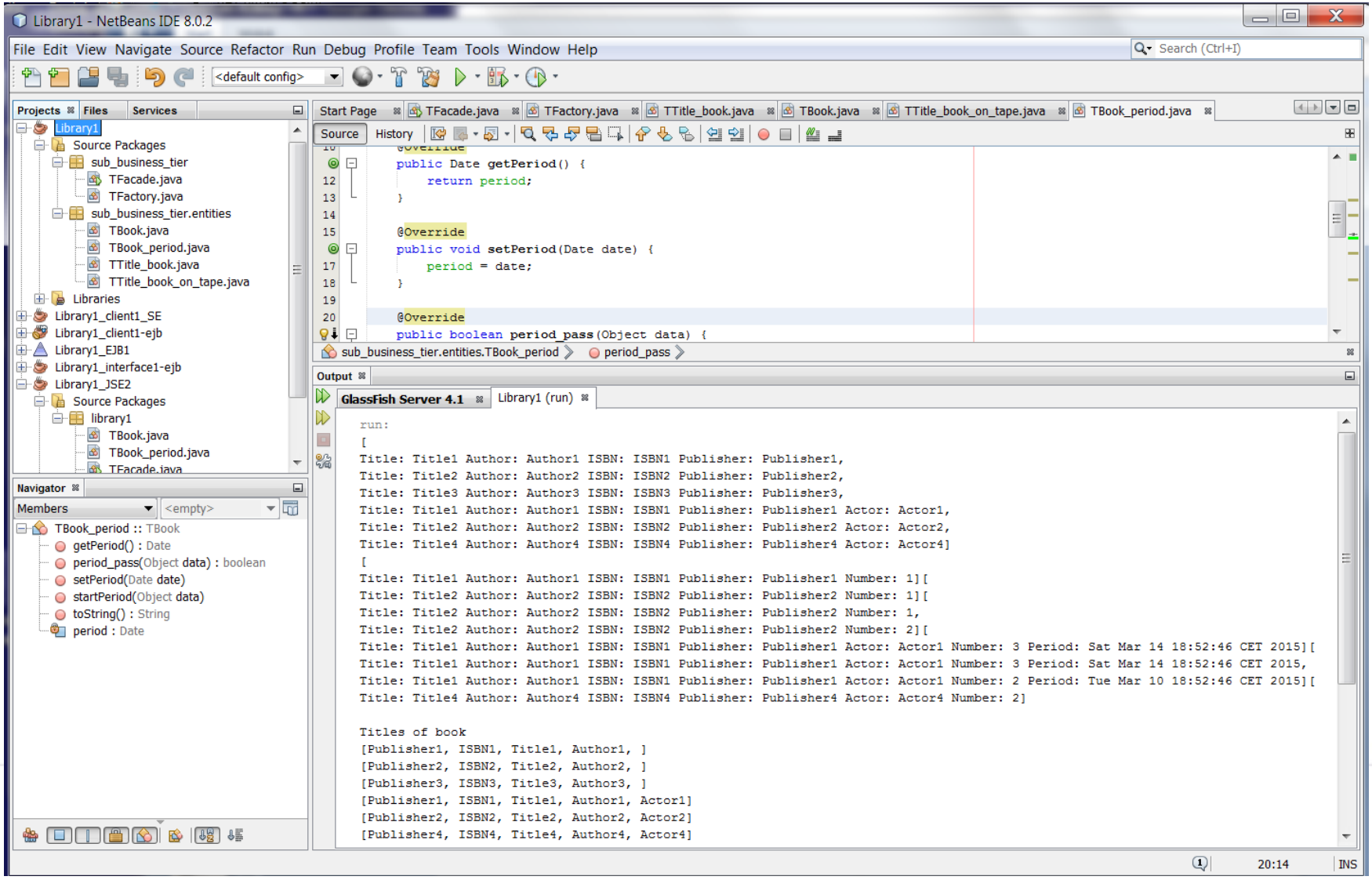




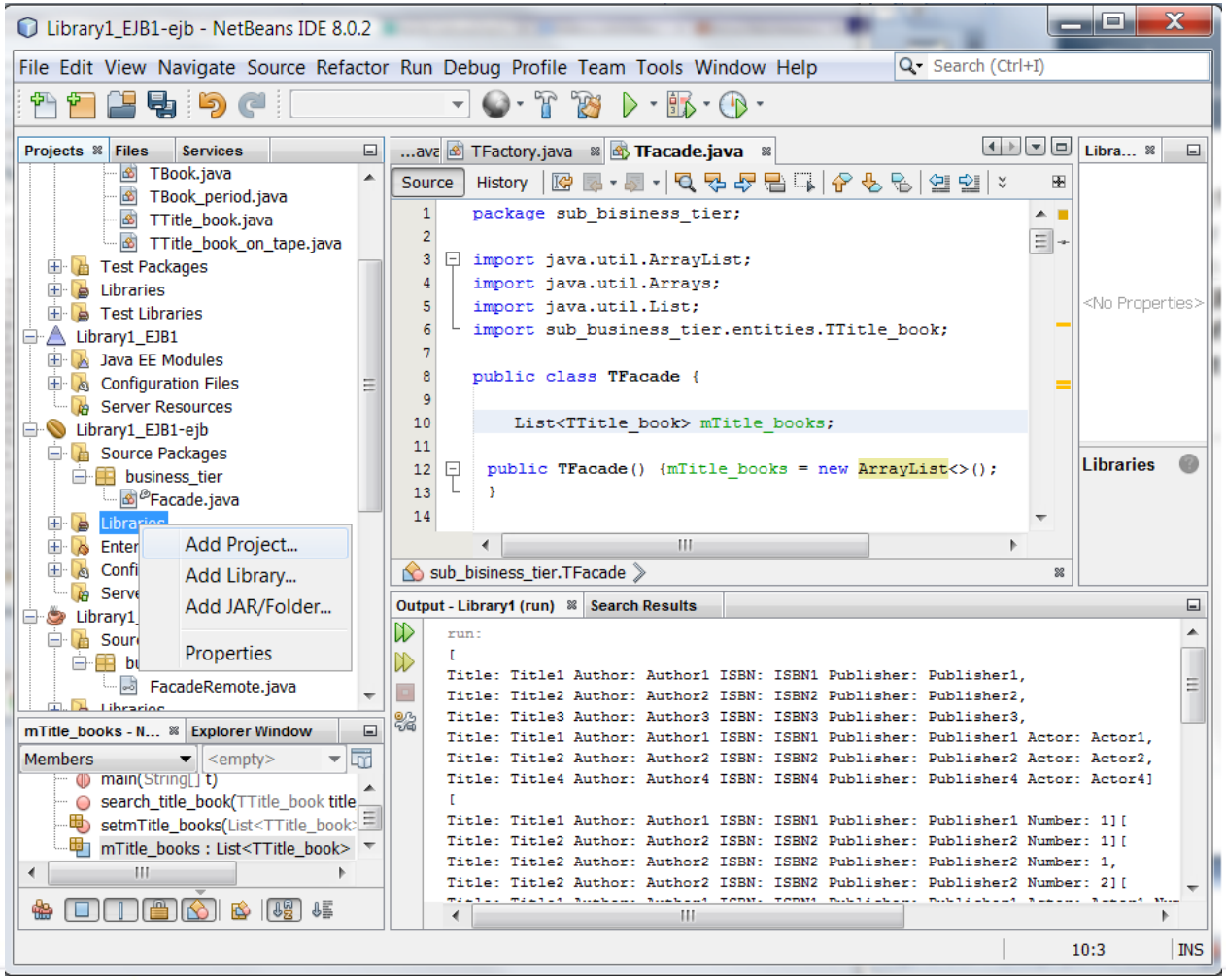
4.6.5. The result



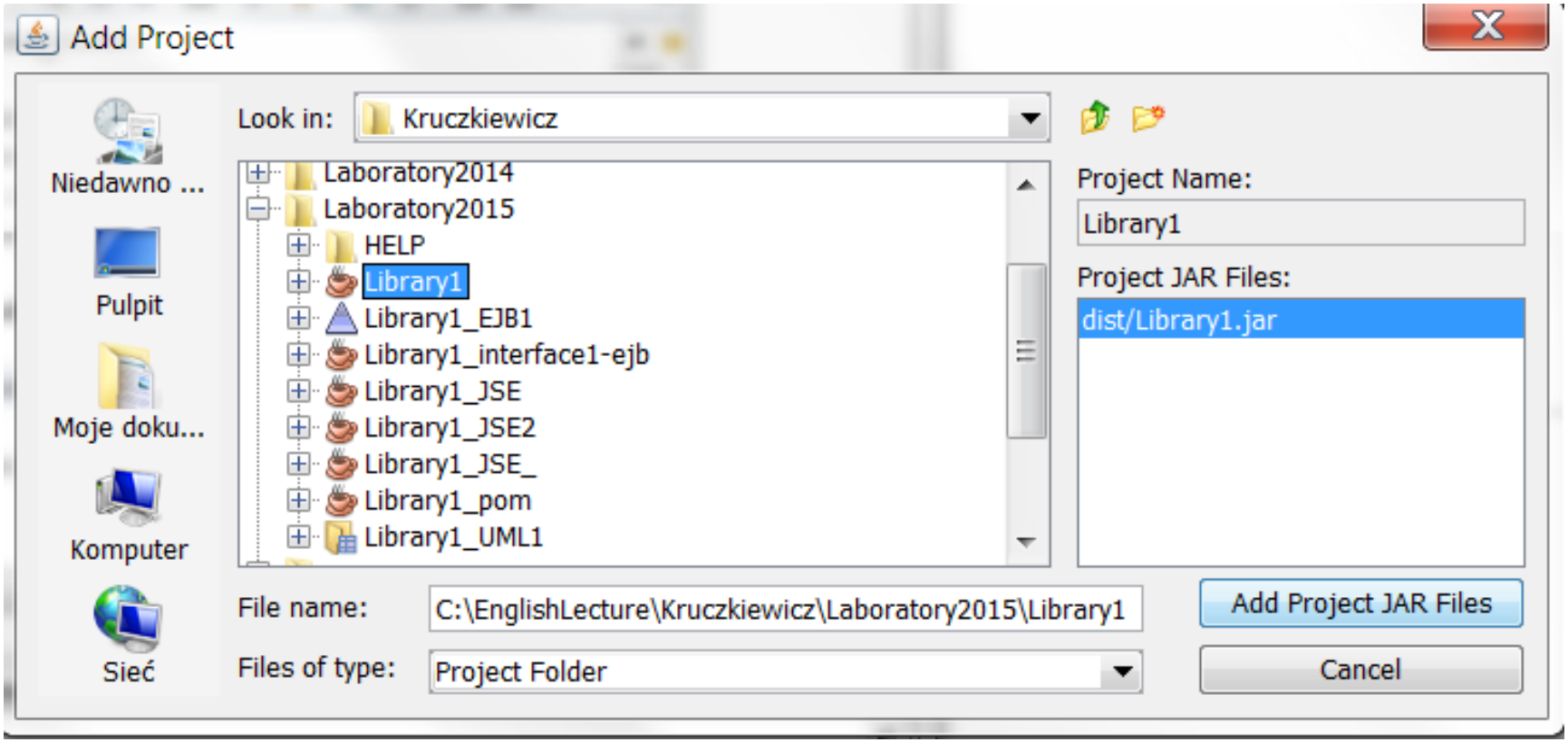
4.7. Build and Run the Library1 Project



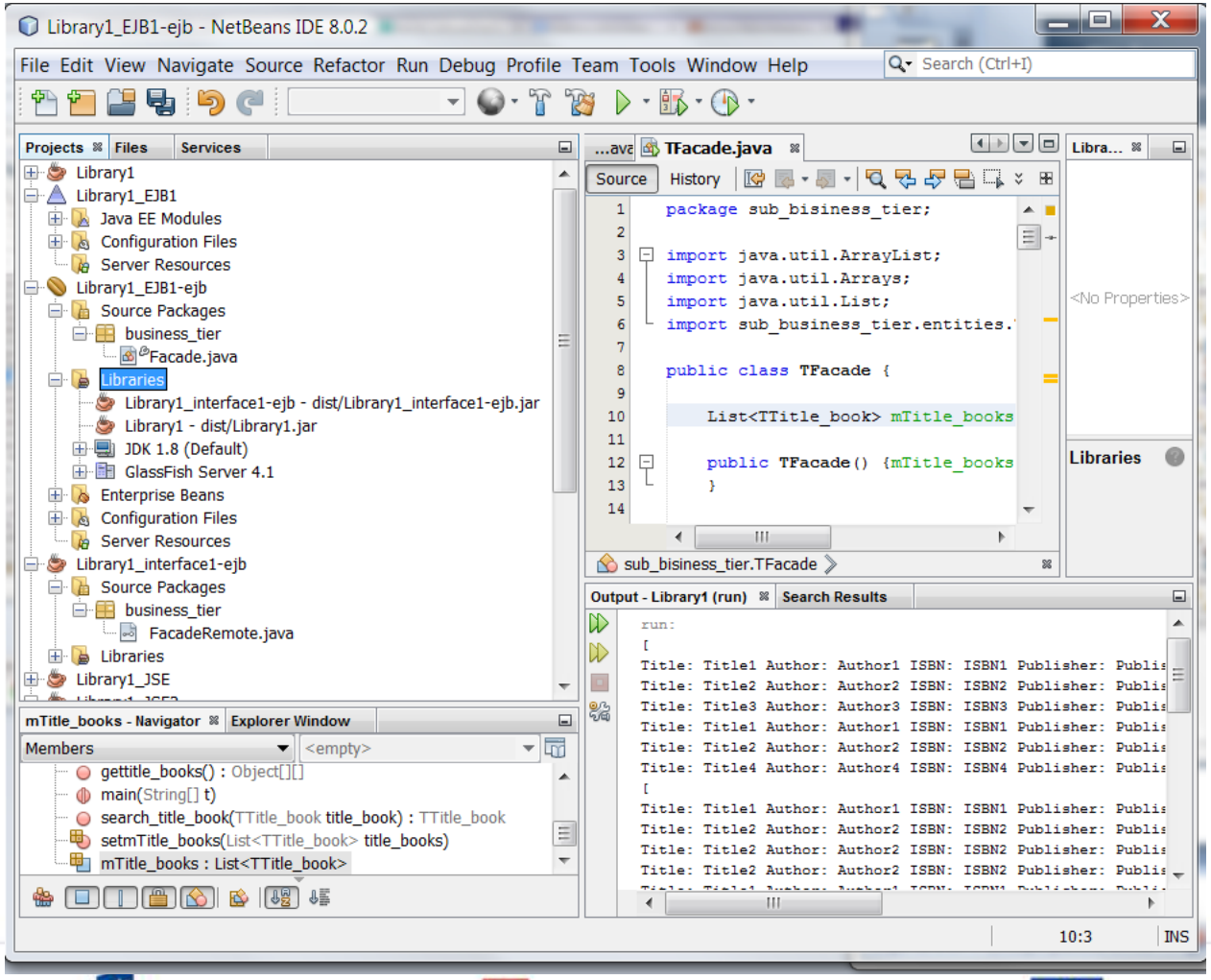
5./ 5.1. /5.1.1. Adding the library project (p.4) to the EJB module of EE project



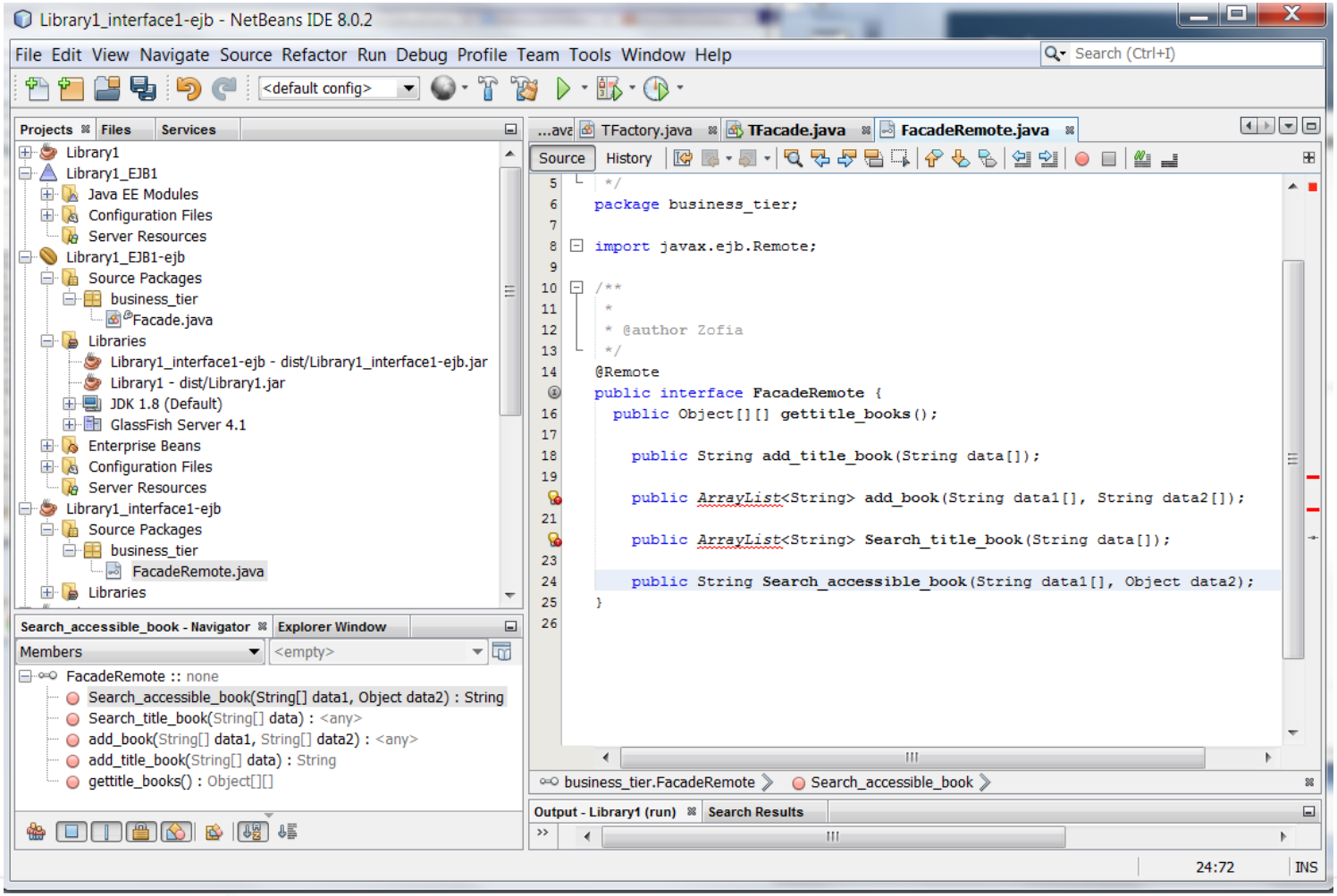
5.1.2. Adding the library project (p.4) to the EJB module of EE project - select the Library1 project and click the Add Project JAR Files



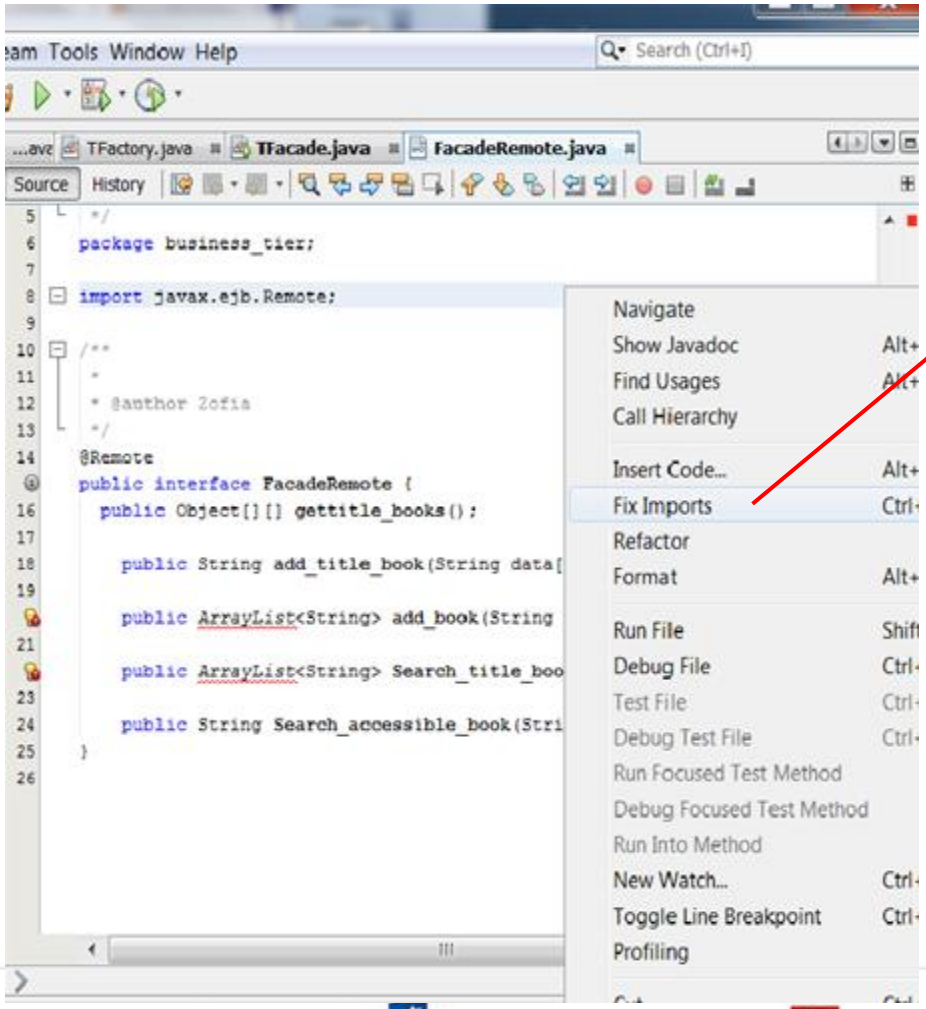
5.1.3. Adding the library project (p.4) to the EJB module of EE project p select the Library1 project - result



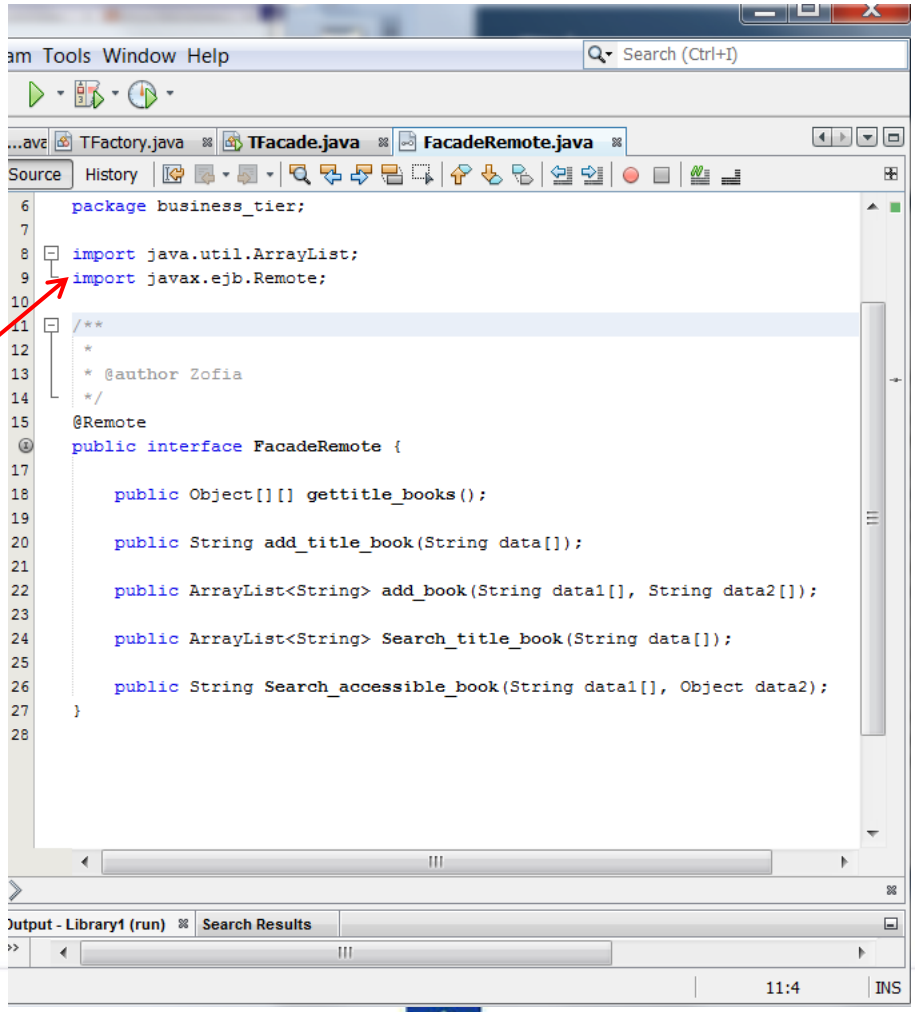
5.2. /5.2.1. Definition of interface of EJB Facade (as the adapter of methods of POJO TFacade)



5.2.2. „Fix Imports” process



5.2.3. „Fix Imports” process - result

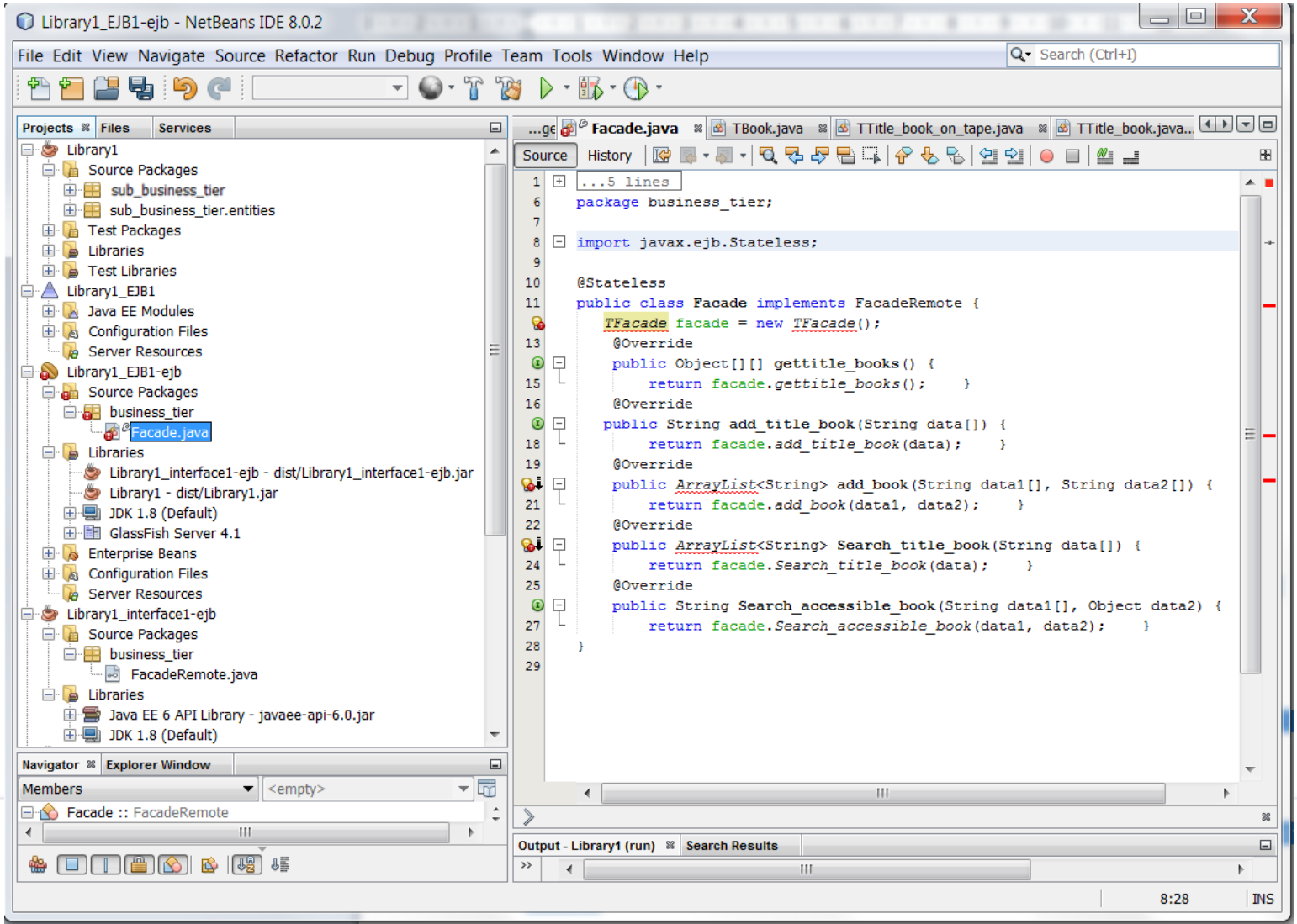



```
package business_tier;

import java.util.ArrayList;
import javax.ejb.Remote;

/**
 *
 * @author Zofia
 */
@Remote
public interface FacadeRemote {

    public Object[][] gettitle_books();

    public String add_title_book(String data[]);

    public ArrayList<String> add_book(String data1[], String data2[]);

    public ArrayList<String> Search_title_book(String data[]);

    public String Search_accessible_book(String data1[], Object data2);
}
```

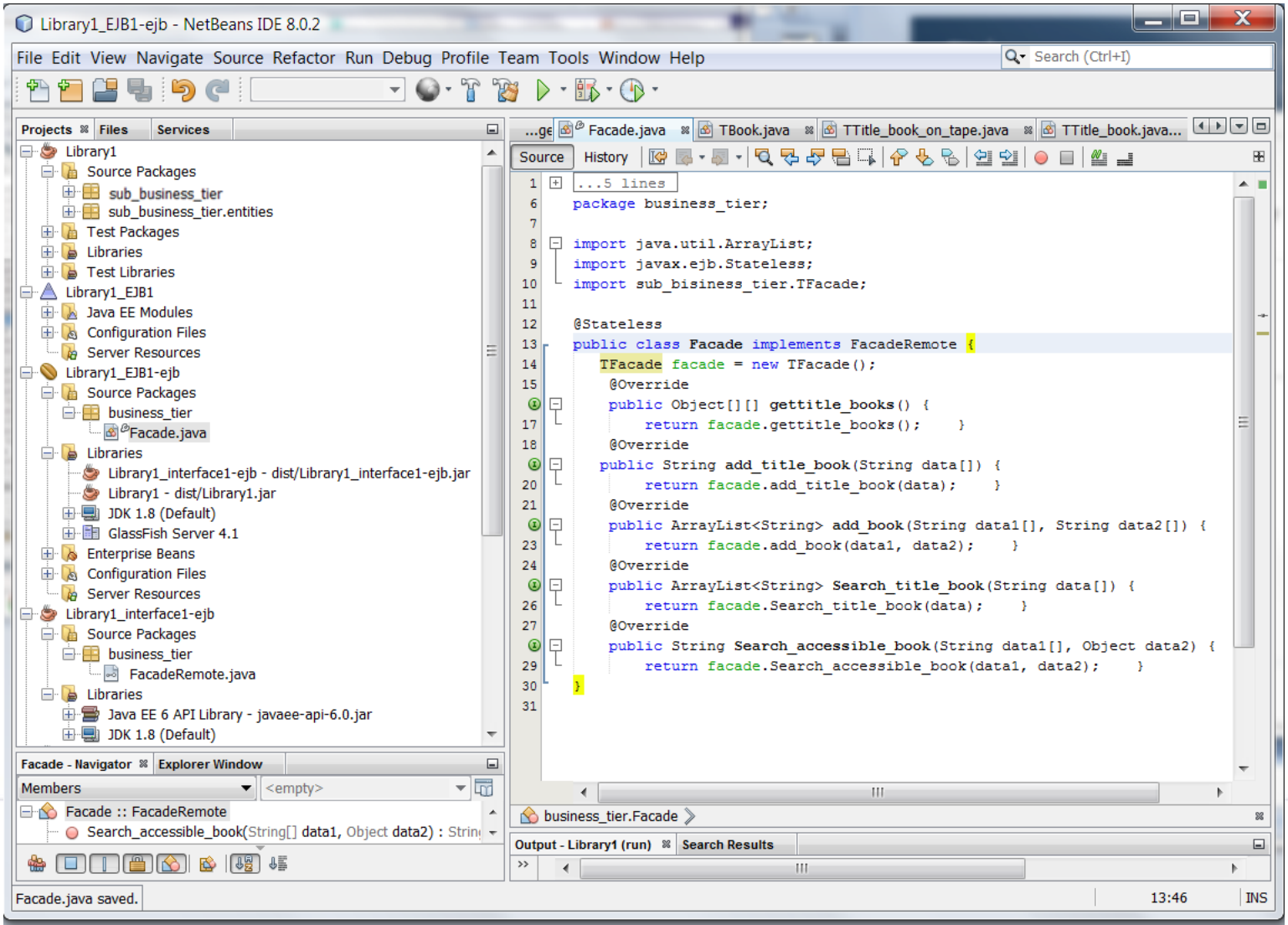
5.2.4 – Definition of the FacadeRemote interface - you must add the declaration of these methods, which you have added during the development of the lab1 during the lab2

```
package business_tier;

import java.util.ArrayList;
import javax.ejb.Remote;

@Remote
public interface FacadeRemote {

    public Object[][] gettitle_books();

    public String add_title_book(String data[]);

    public ArrayList<String> add_book(String data1[], String data2[]);

    public ArrayList<String> Search_title_book(String data[]);

    public String Search_accessible_book(String data1[], Object data2);
   // you must add the declaration of these methods, which you have added
   //during the development of the lab1 during the lab2
}
```

5.3. /5.3.1. Definition of EJB Facade as the Session Bean type attribute in the Librrary_EJB1 – ejb module (as the adapter of methods of the POJO TFacade class) – before Fix Imports



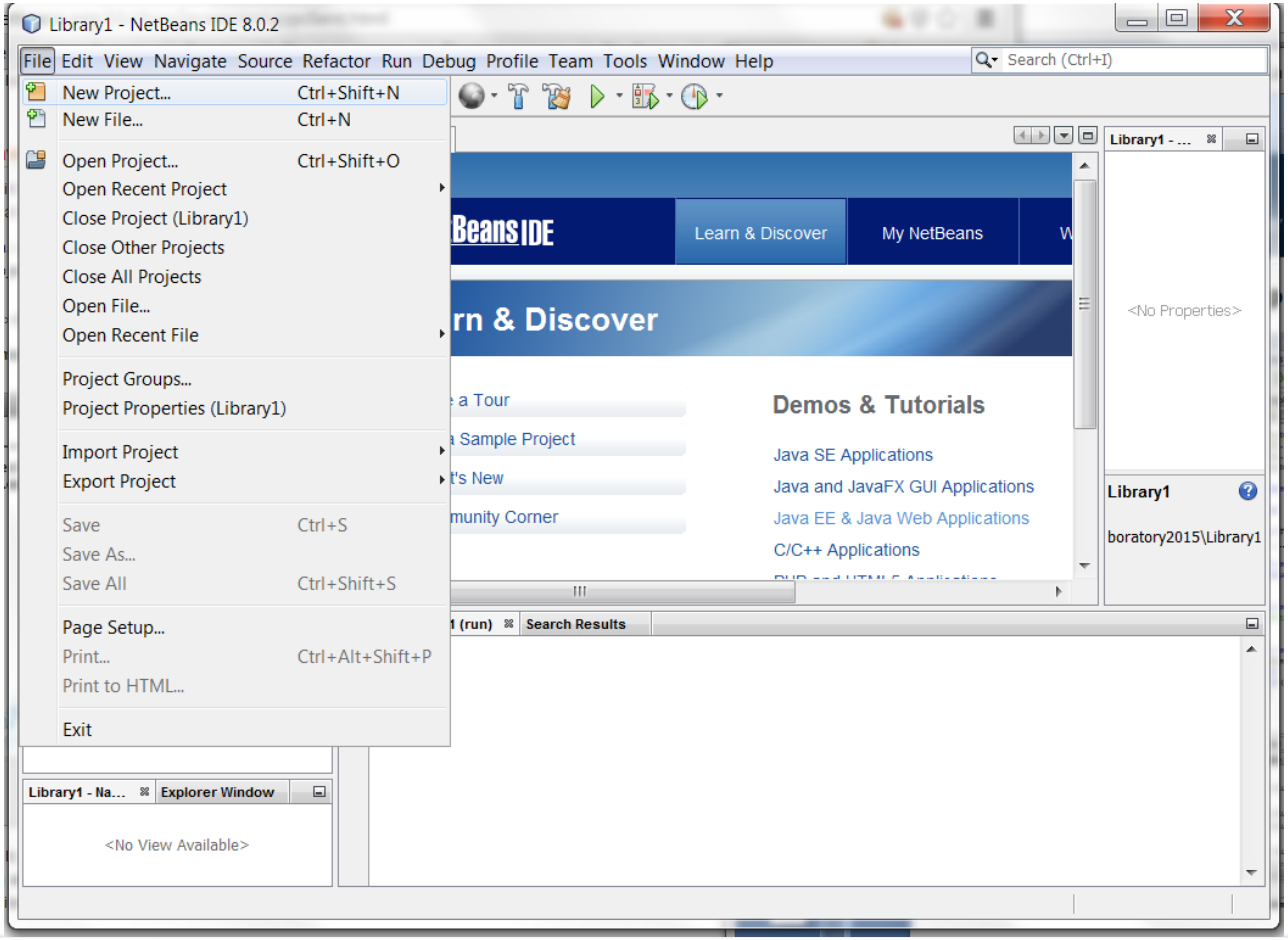
5.3.2. Definition of EJB Facade as the Session Bean type attribute in the Librrary_EJB1 – ejb module (as the adapter of methods of the POJO TFacade class) - after Fix Imports



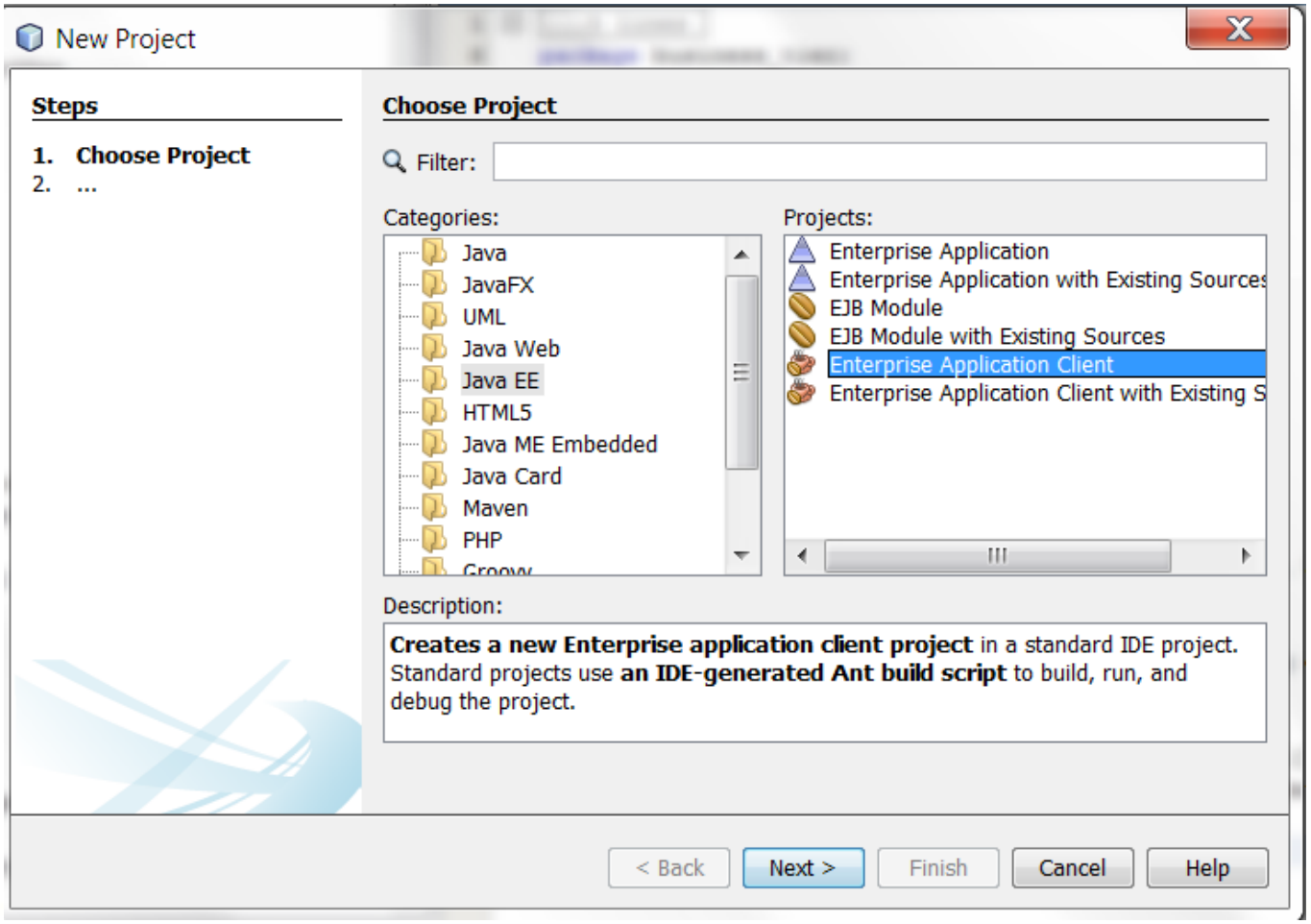
5.3.3. The implementation of FacadeRemote interface as the Facade SessionBean - // you must add the implementation of these methods, which you have added during the development of the lab1 during the lab2 and declared in the FacadeRemote interface.

```
package business_tier;

import java.util.ArrayList;
import javax.ejb.Stateless;
import sub_business_tier.TFacade;


@Stateless
public class Facade implements FacadeRemote {

    TFacade facade = new TFacade();

    @Override
    public Object[][] gettitle_books() {
        return facade.gettitle_books();    }

    @Override
   public String add_title_book(String data[]) {
        return facade.add_title_book(data);    }

    @Override
    public ArrayList<String> add_book(String data1[], String data2[]) {
        return facade.add_book(data1, data2);    }

    @Override
    public ArrayList<String> Search_title_book(String data[]) {
        return facade.Search_title_book(data);    }

    @Override
    public String Search_accessible_book(String data1[], Object data2) {
        return facade.Search_accessible_book(data1, data2);    }
}
```

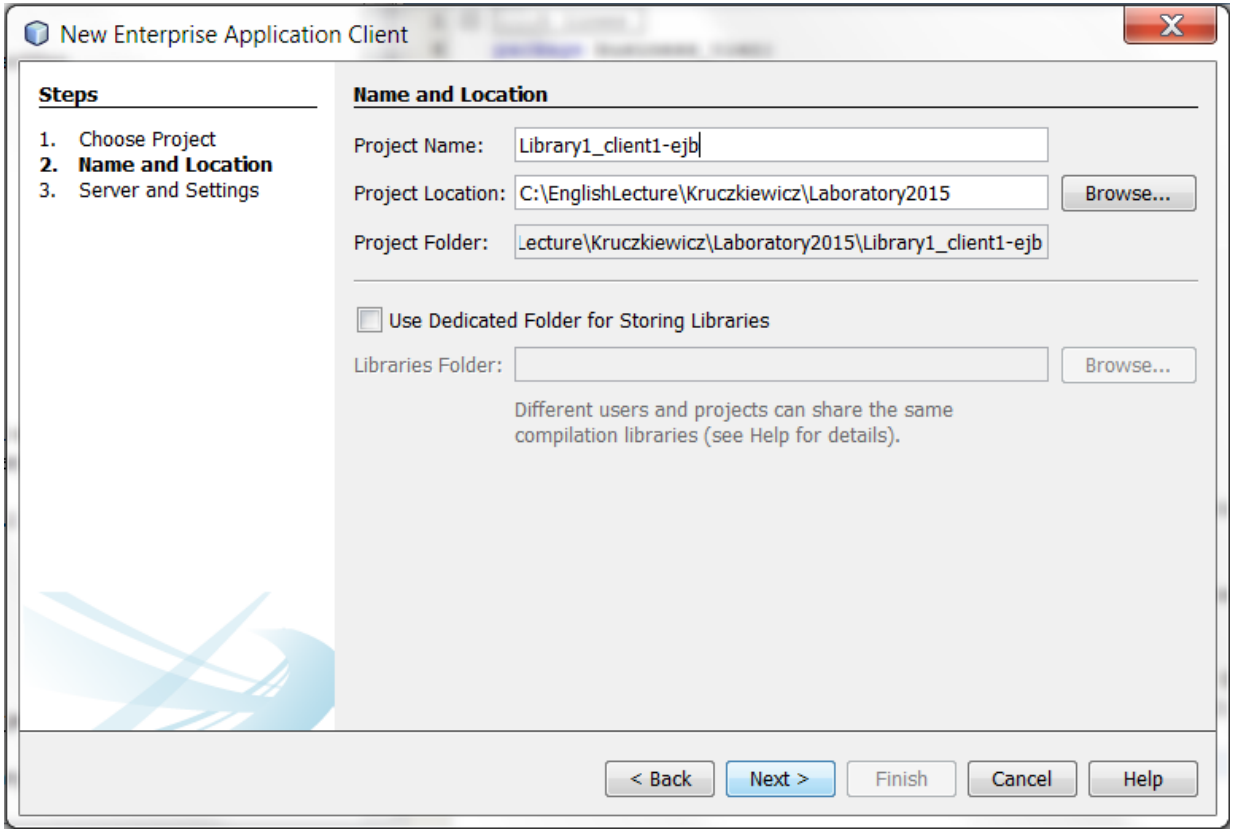
6. /6.1. /6.1.1. Creation of the EE client tier – click File/New Project…



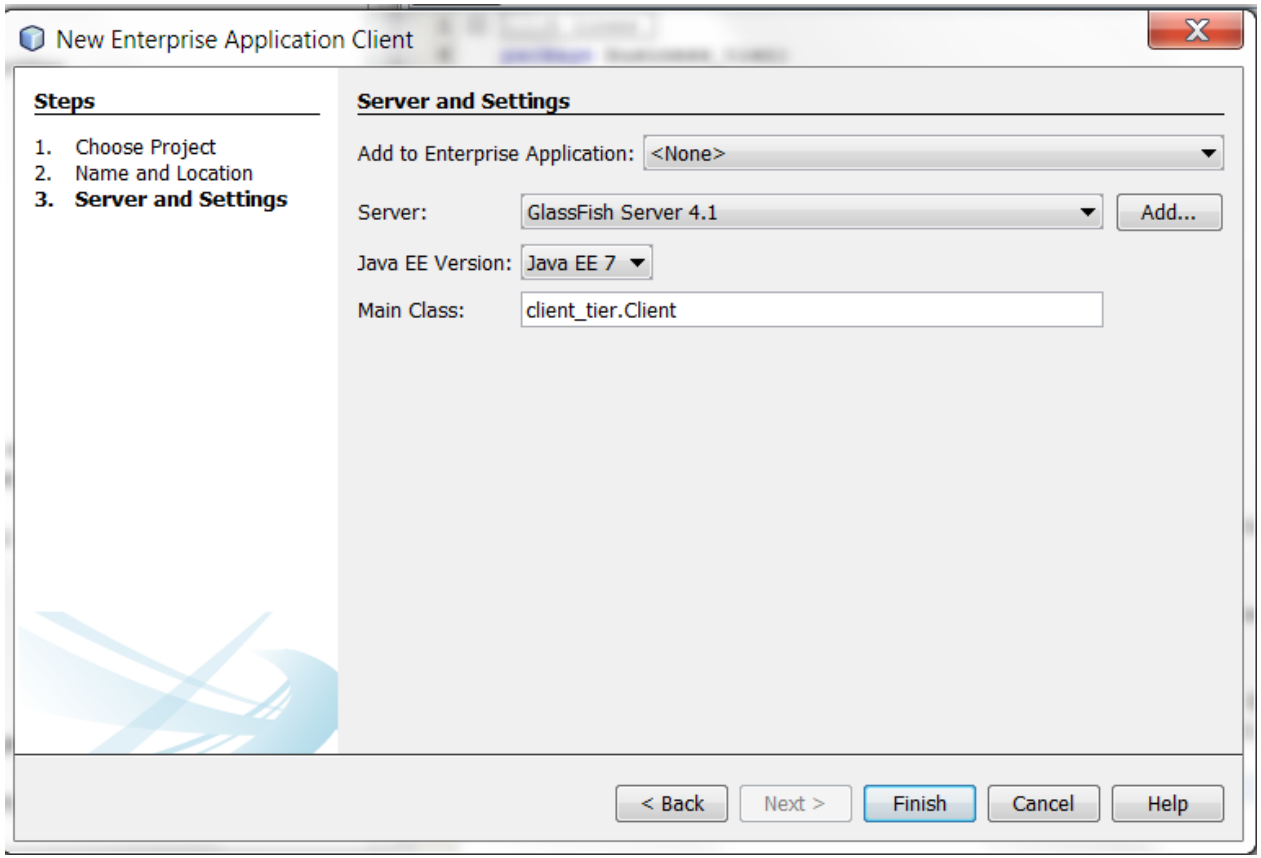
6.1.2. Creation of the EE client tier – select the Java EE/ Enterprise Application Client items. Click Next.



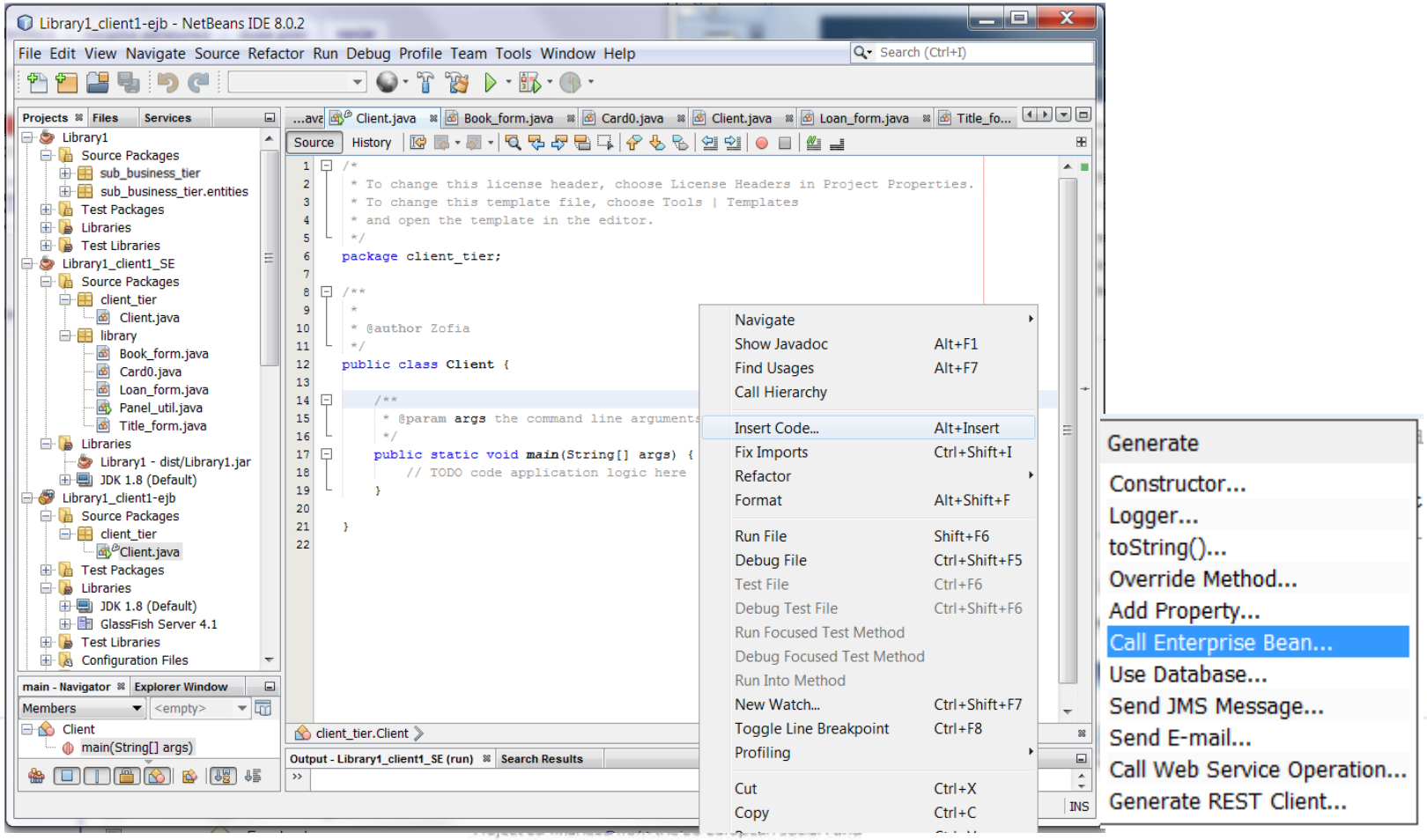
6.1.3. Creation of the EE client tier – fill the project name and select the project location. Click Next.



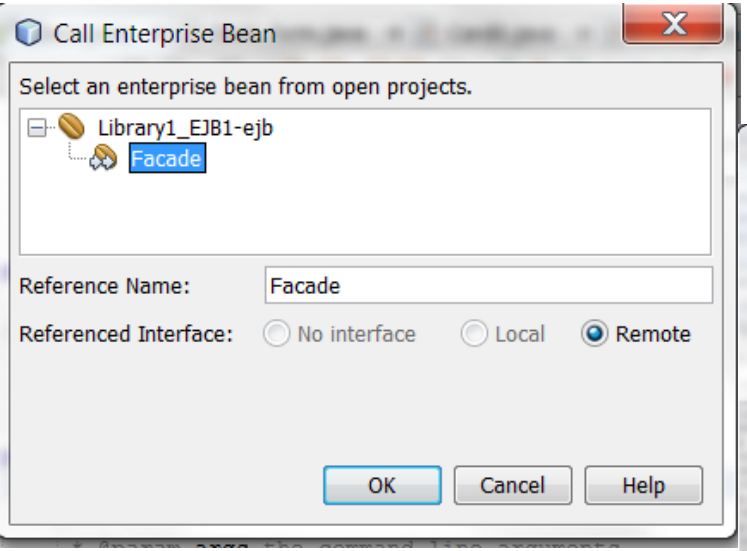
6.1.4. Creation of the EE client tier – select the server, Java EE version and fill the name of main class. Click Finish.



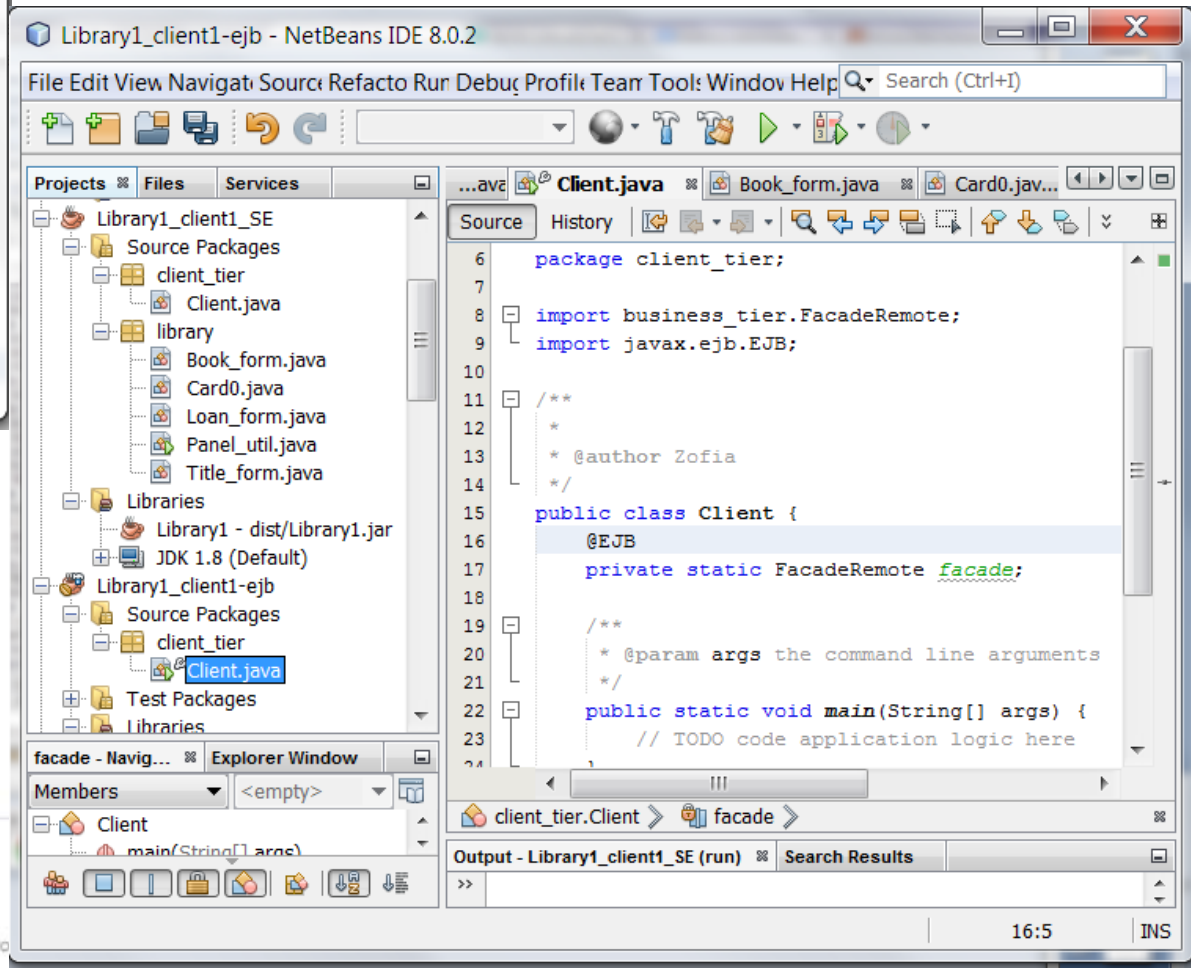
6.2. / 6.2.1. Definition of code of the EE client based on code of the SE client – only change of TFacade class from POJO SE TFacade class into the EJB Facade class.
Creation of connection from the Facade Session Bean of Library1_EJB1-ejb module to the Client class of the Library1_client1-ejb project – expand the Source Packages node of the Library1_client1-ejb project and open Client.java in the editor. Right-click in the source code and choose Insert Code… item and select Call Enterprise Bean to open the Call Enterprise Bean dialog.



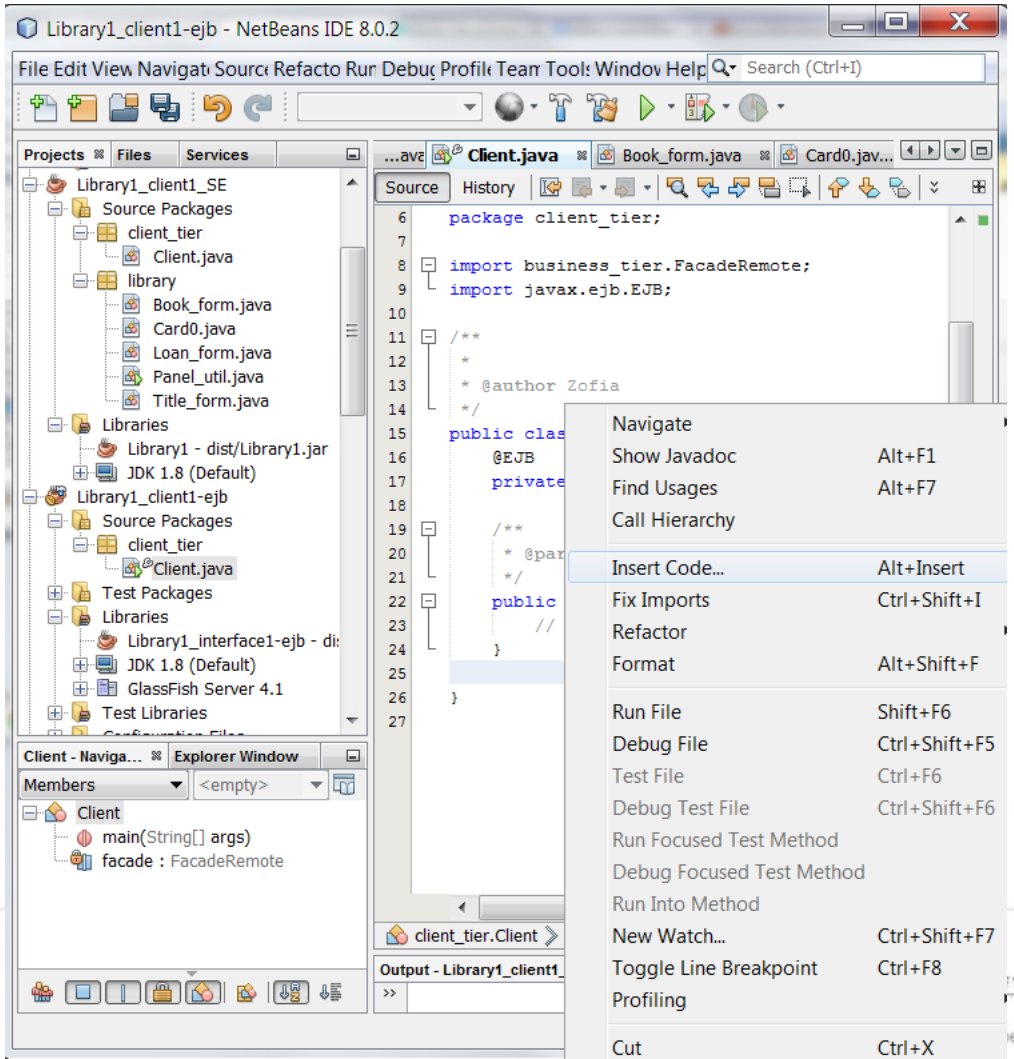
6.2.2. Creation of connection from the Facade Session Bean of Library1_EJB1-ejb module to the Client class of the Library_client-ejb project - select the Facade EJB from the Library1_EJB1-ejb module of Library1_EJB1 project

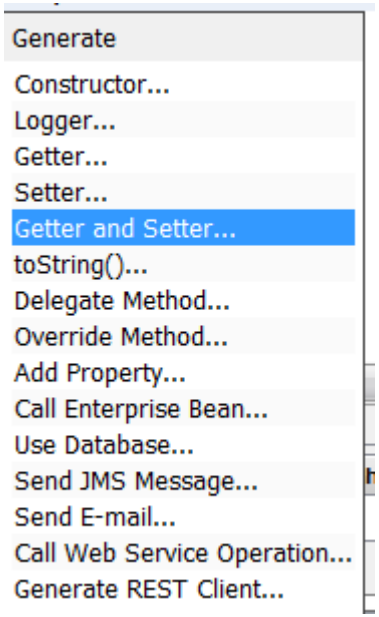


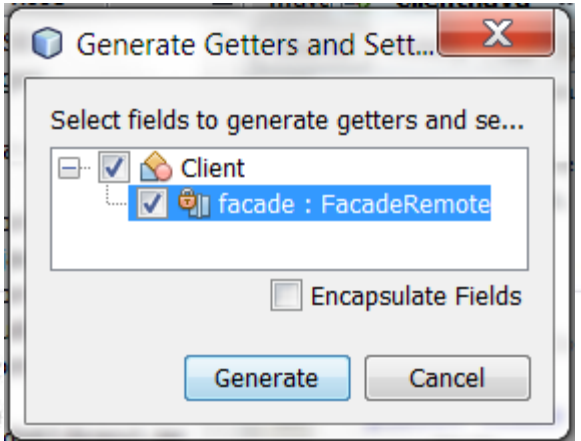
6.2.3. The result of creation the connection to the Facade SessionBean of the Library1_EJB1-ejb module.



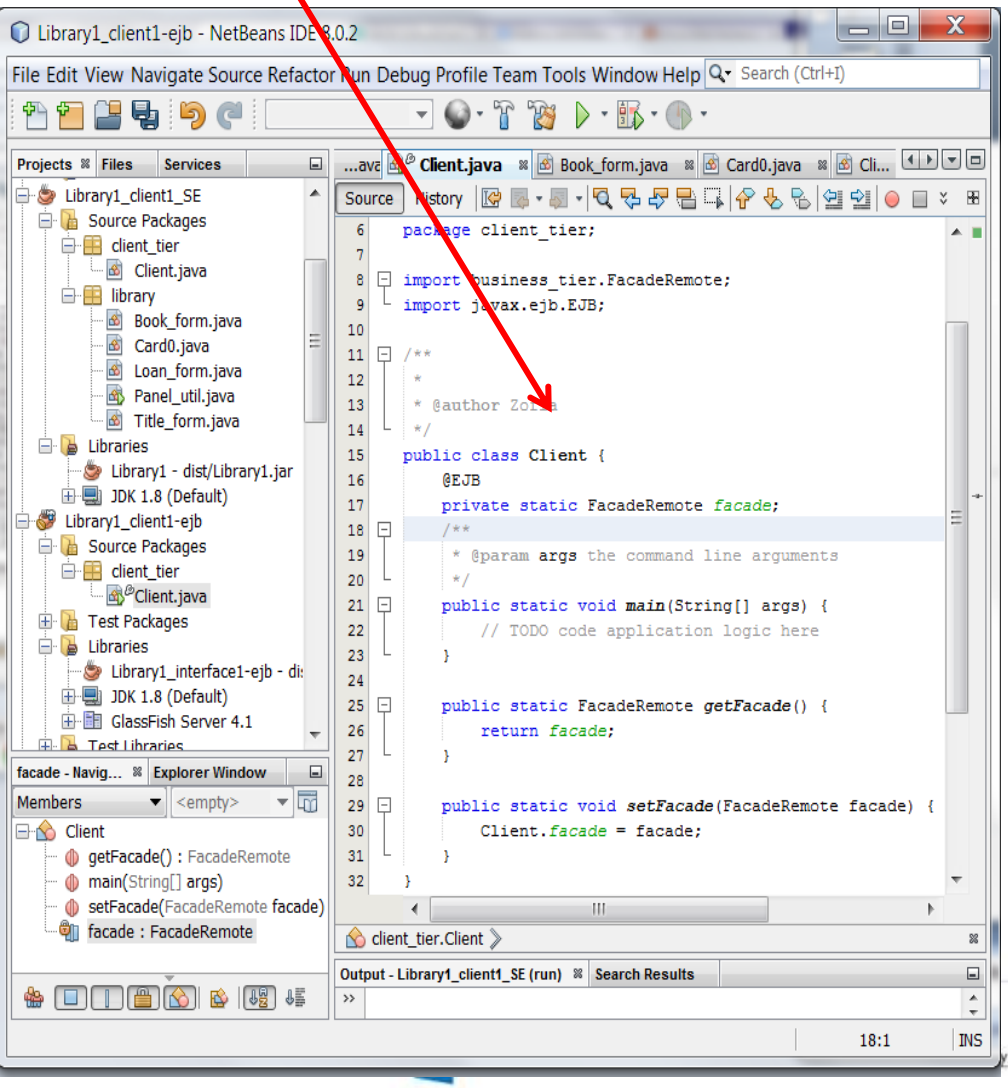
6.3. /6.3.1. Adding the getter and setter methods to the facade attribute as the reference to the Facade SessionBean by using the „Insert Code" process – select Insert Code/Getter and Setter… items to open the Generate Getters and Setters dialog and select the facade attribute







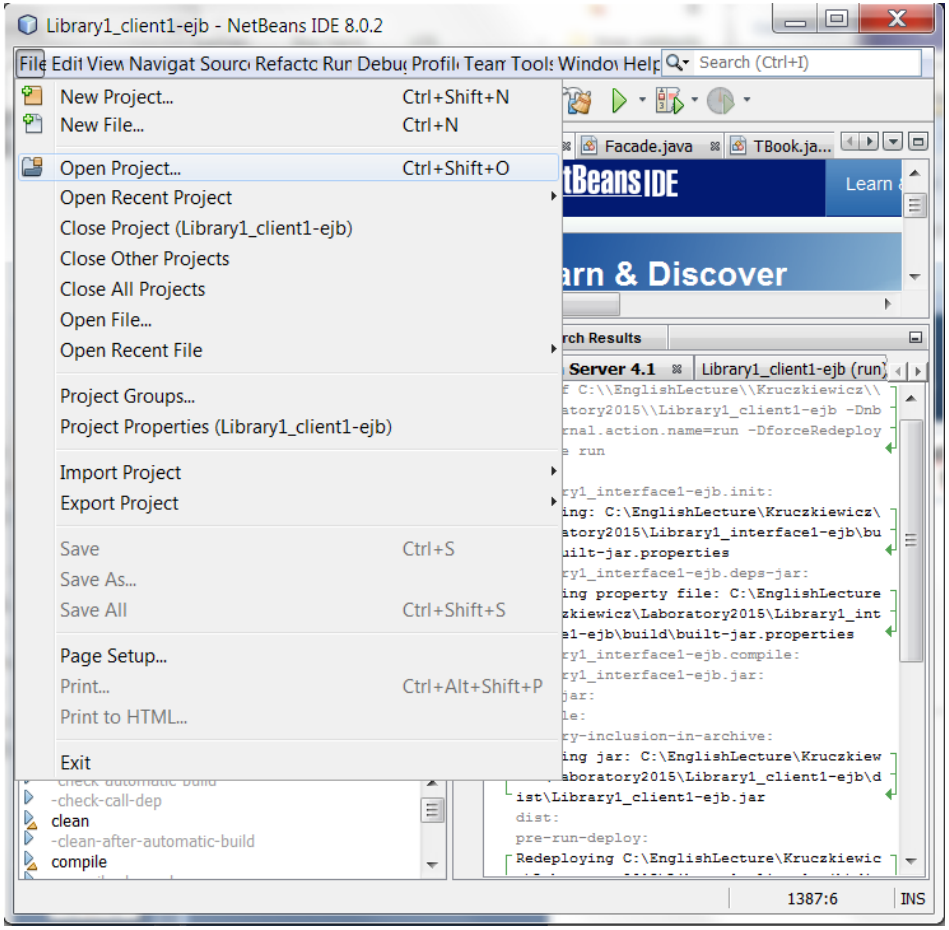
6.3.2. The result

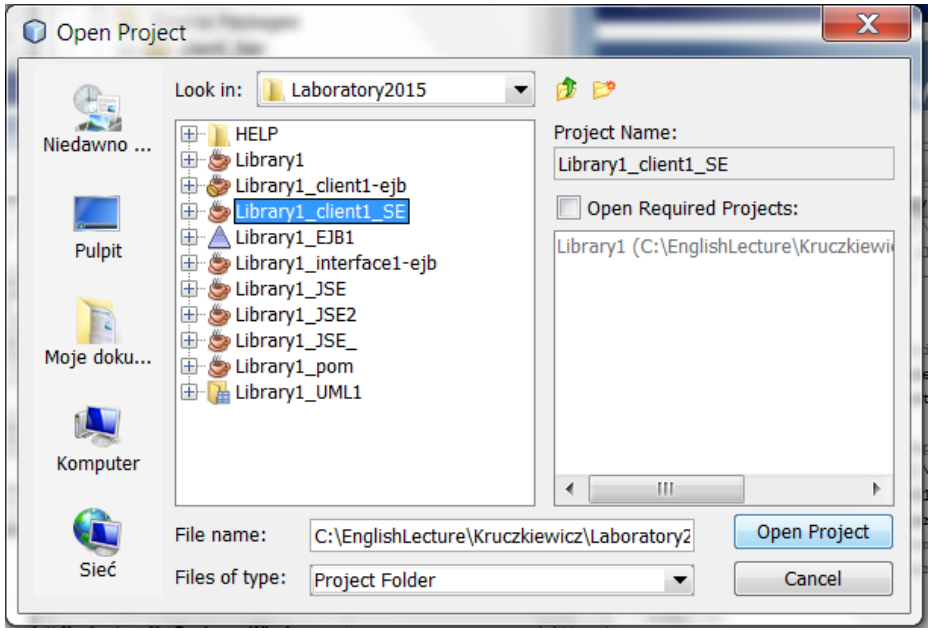


6.3.3. Define the body of main method based on code of the Client class main method from the Standard Edition client projecet named Library1_client1_SE

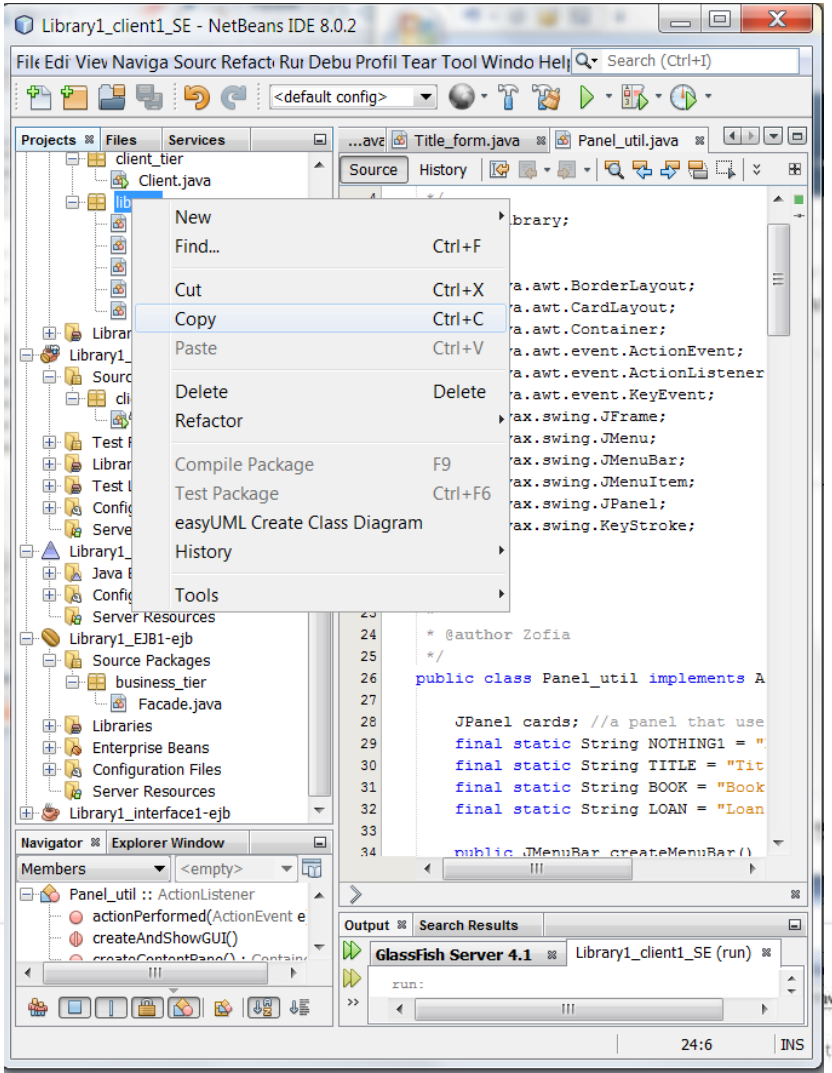
```
public static void main(String[] args) {
  java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            Panel_util.createAndShowGUI();
        }
    });
}
```

6.4. /6.4.1. Coping the library package from the Standard Edition Application Client project as the Library1_client1_SE one (attachment to the lab3)

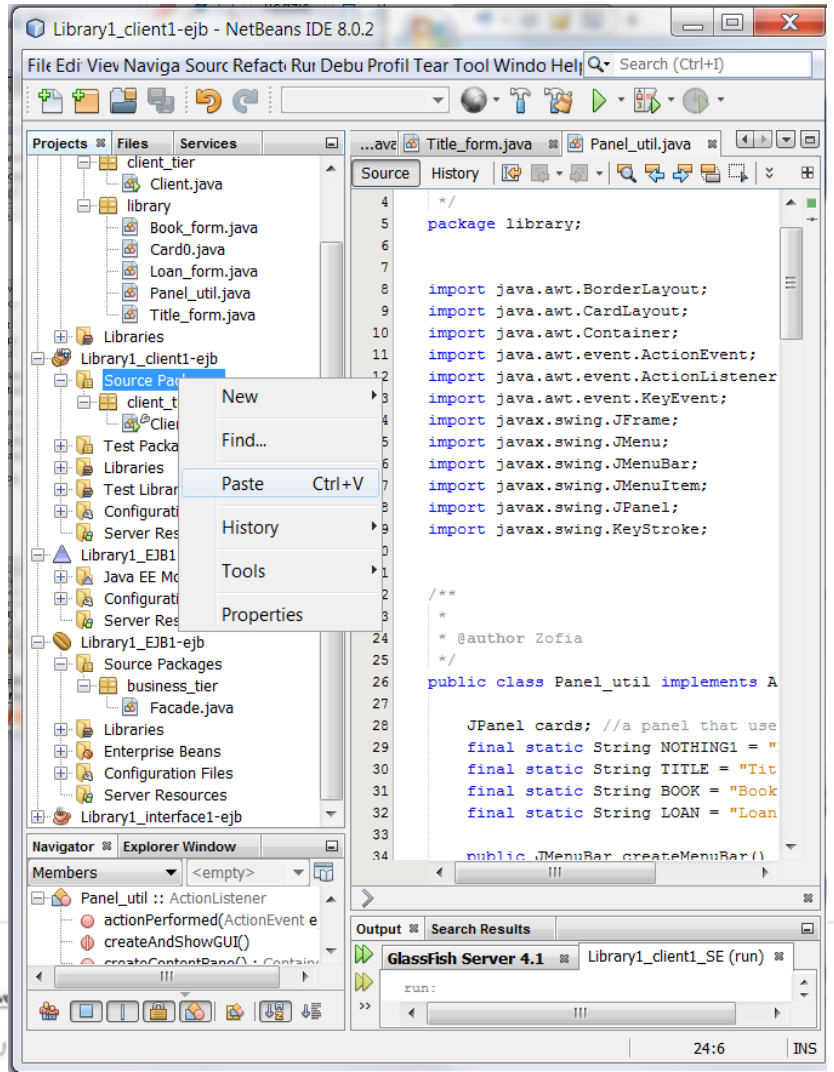




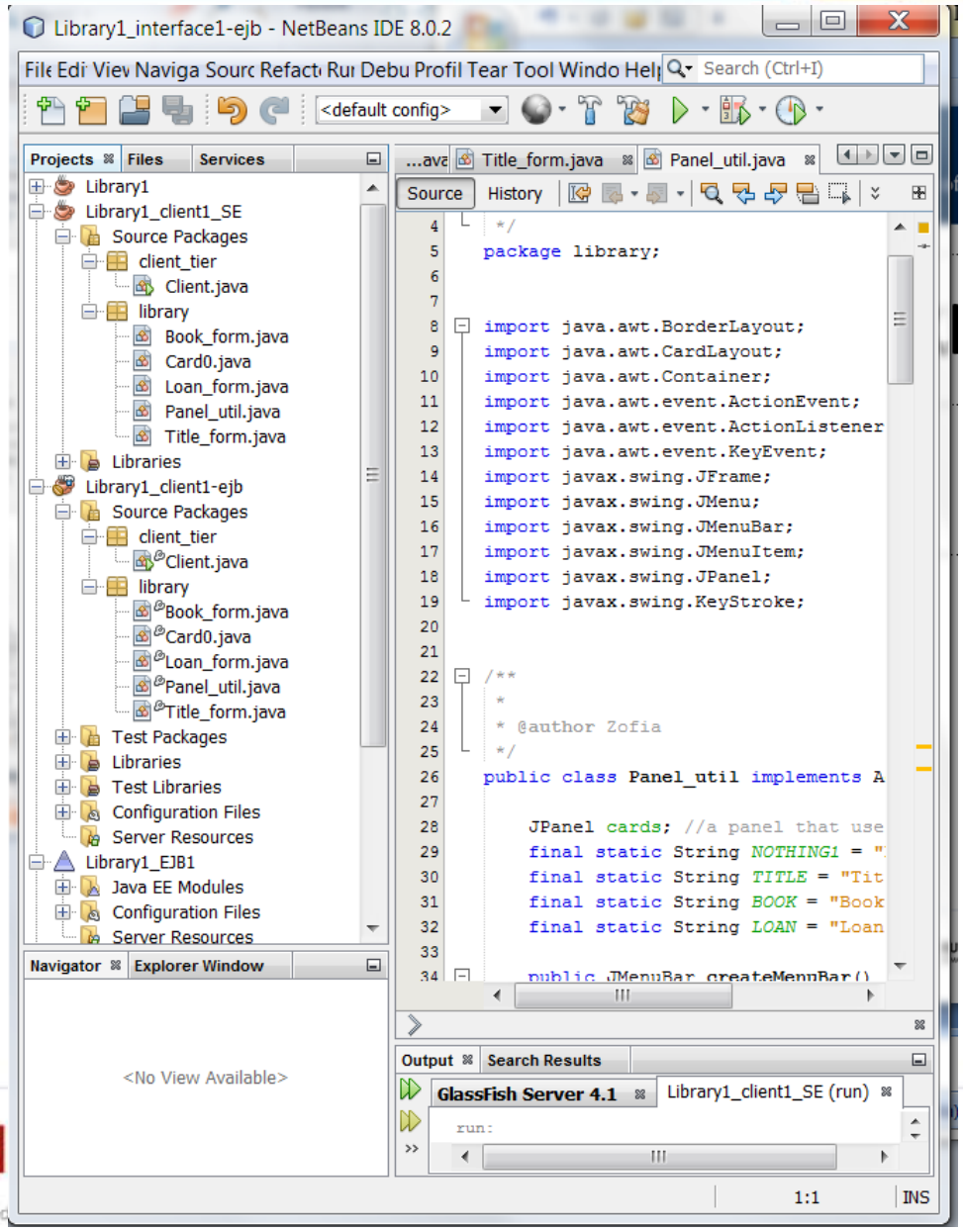
6.4.2. Coping the library package from the Standard Edition Application Client project as the Library1_client1_SE one



6.4.3. Paste the library package from the Library1_client1_SE to the Library1_client1-ejb type Enterprise Application Client project



6.4.4. The result – creation the Library1_client1-ejb type the Enterprise Application Client project based on the code of the Standard Edition version of the client project



6.5. The change of getter and setter methods of the Facade class as the Session Bean type attribute

```
package client_tier;

import library.Panel_util;
import sub_business_tier.TFacade;


public class Client {

   static TFacade facade = new TFacade();

    static public TFacade getFacade() {
       return facade;
     }

    static public void setFacade(TFacade facade) {
       Client.facade = facade;
     }

    public static void main(String[] args) {
       java.awt.EventQueue.invokeLater(new Runnable() {
          public void run() {
               Panel_util.createAndShowGUI(); }
       });
    }
}
```

```
package client_tier;

import business_tier.FacadeRemote;
import javax.ejb.EJB;
import library.Panel_util;

public class Client {
    @EJB
    private static FacadeRemote facade;

    public static FacadeRemote getFacade() {
       return façade;
    }

    public static void setFacade(FacadeRemote facade) {
       Client.facade = façade;
    }

    public static void main(String[] args) {
       java.awt.EventQueue.invokeLater(new Runnable() {
          public void run() {
             Panel_util.createAndShowGUI();  }
       });
    }
}
```

6.6. The same code – different type of return value from client.getFacade()

```
public void actionPerformed(ActionEvent evt) {
      String[] data = form_title();
      if (data == null) {
         return;
      }
      Client.getFacade().add_title_book(data);
}
```

```
public void actionPerformed(ActionEvent evt) {
      String[] data = form_title();
      if (data == null) {
         return;
      }
      Client.getFacade().add_title_book(data);
}
```

Object POJO in SE Client tier

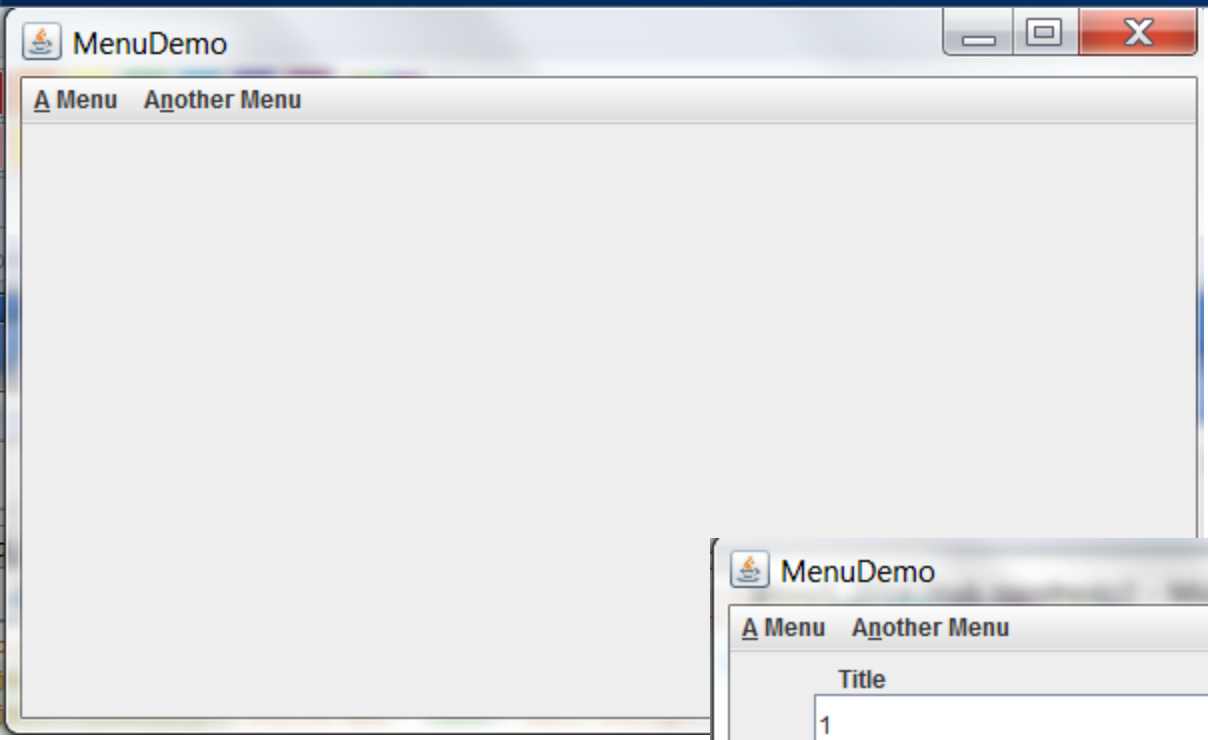
EJB in EE Client tier

6.7. You must add the new panels with forms for supported the new functions (the reservation or the loan of books)

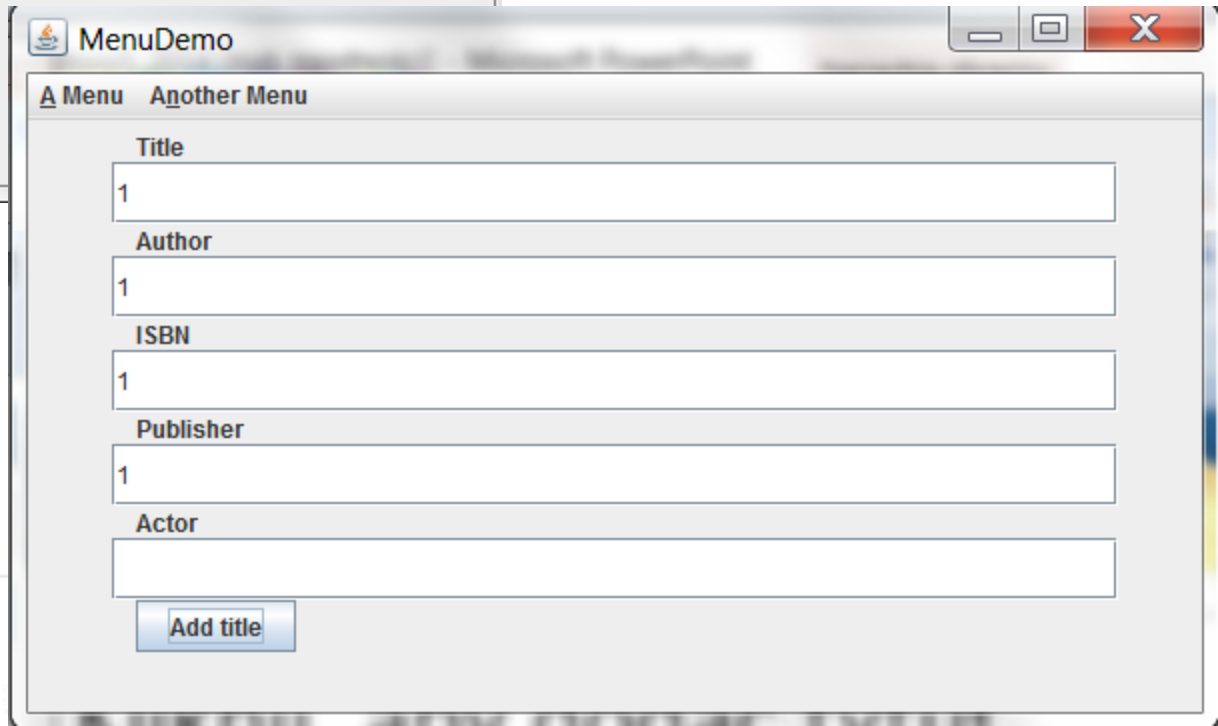
1. **At first, you must add the new panels with forms for supported the new functions (the reservation or the loan of books) in the Java Application projects (as Library1_client1_SE).**
2. **Secondly, you must transform this project into the Enterprise Application Client project, accordingly the instruction of the lab3 (50-62 slides)**

7.0. Running the program

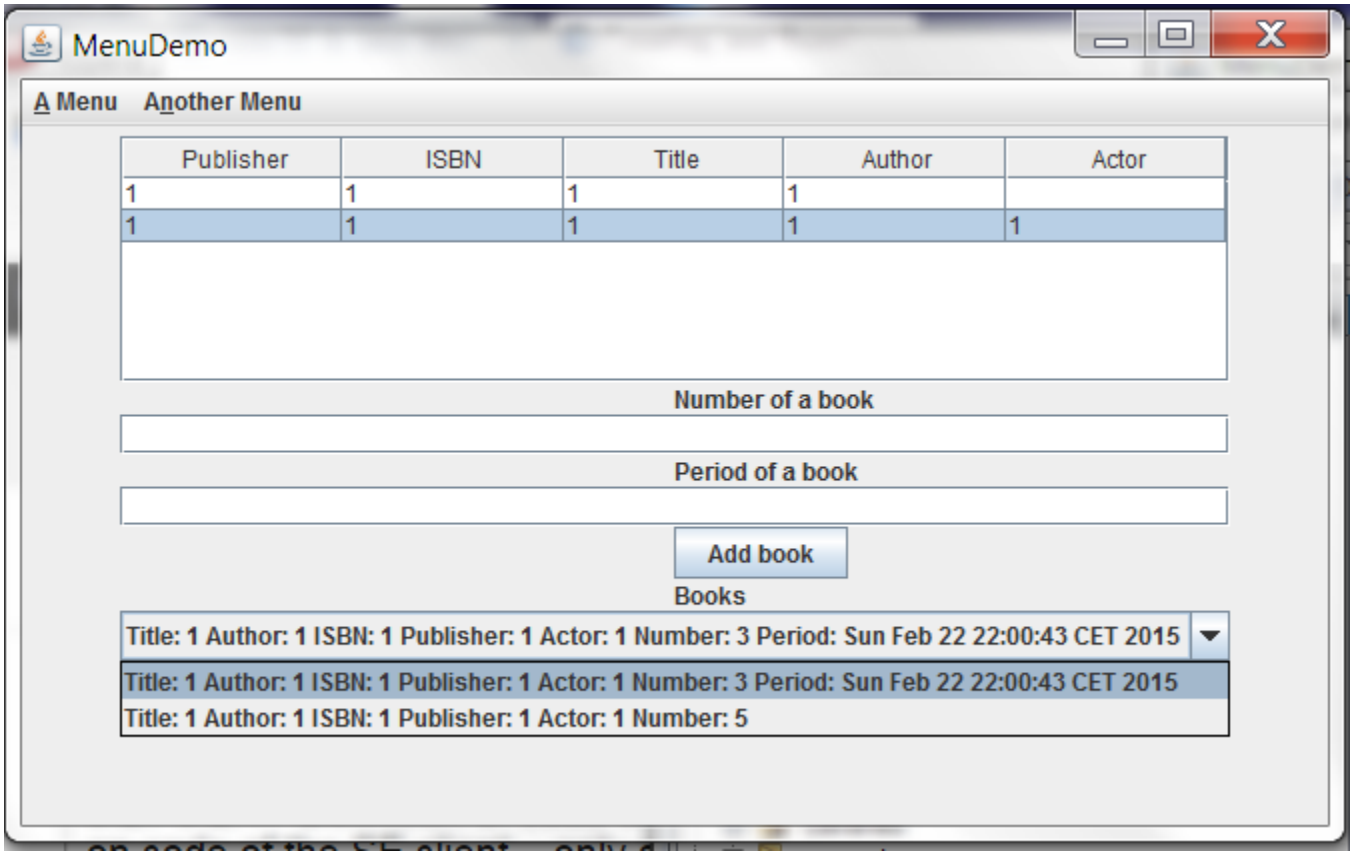
1. After this development, your program will be executed properly, if you clean and build the following programs:
    1. Library1
    2. Library_interface-ejb
    3. Library_EJB1-ejb
    4. Library_client-ejb
2. Then you must deploy the Library_EJB1 program.
3. Finally, you may run a few instances of Library_client-ejb programs - these programs share the common data as titles and books.
4. In the Service Tab you may see, if your EE program deploy properly (Server item). The other useful information you may get from the Glassfish output window tab.



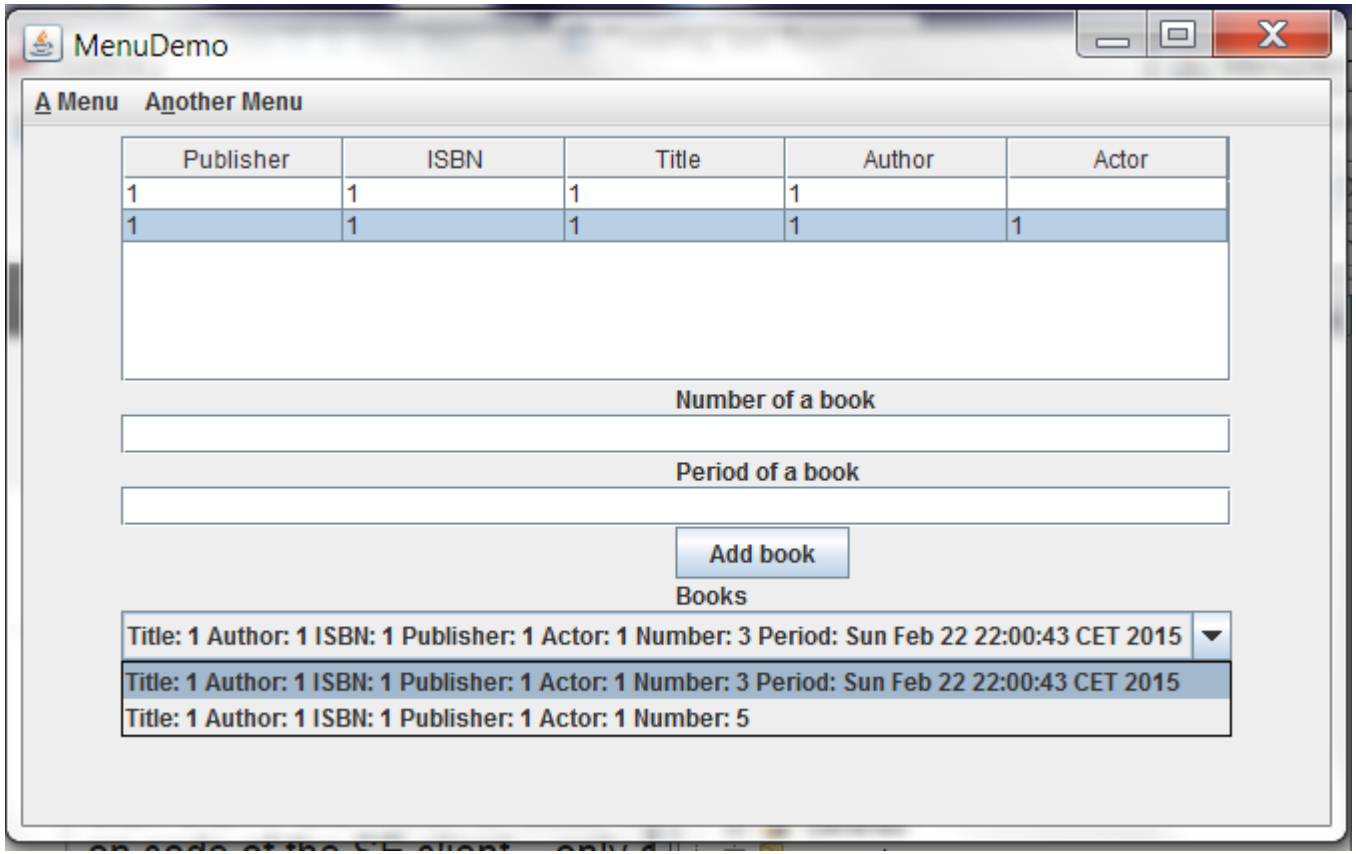
7.1. The view of the Enterprise Application Client (**Library_client-ejb**) for processing of application data - with the same responsibilities as of the version of third laboratory.



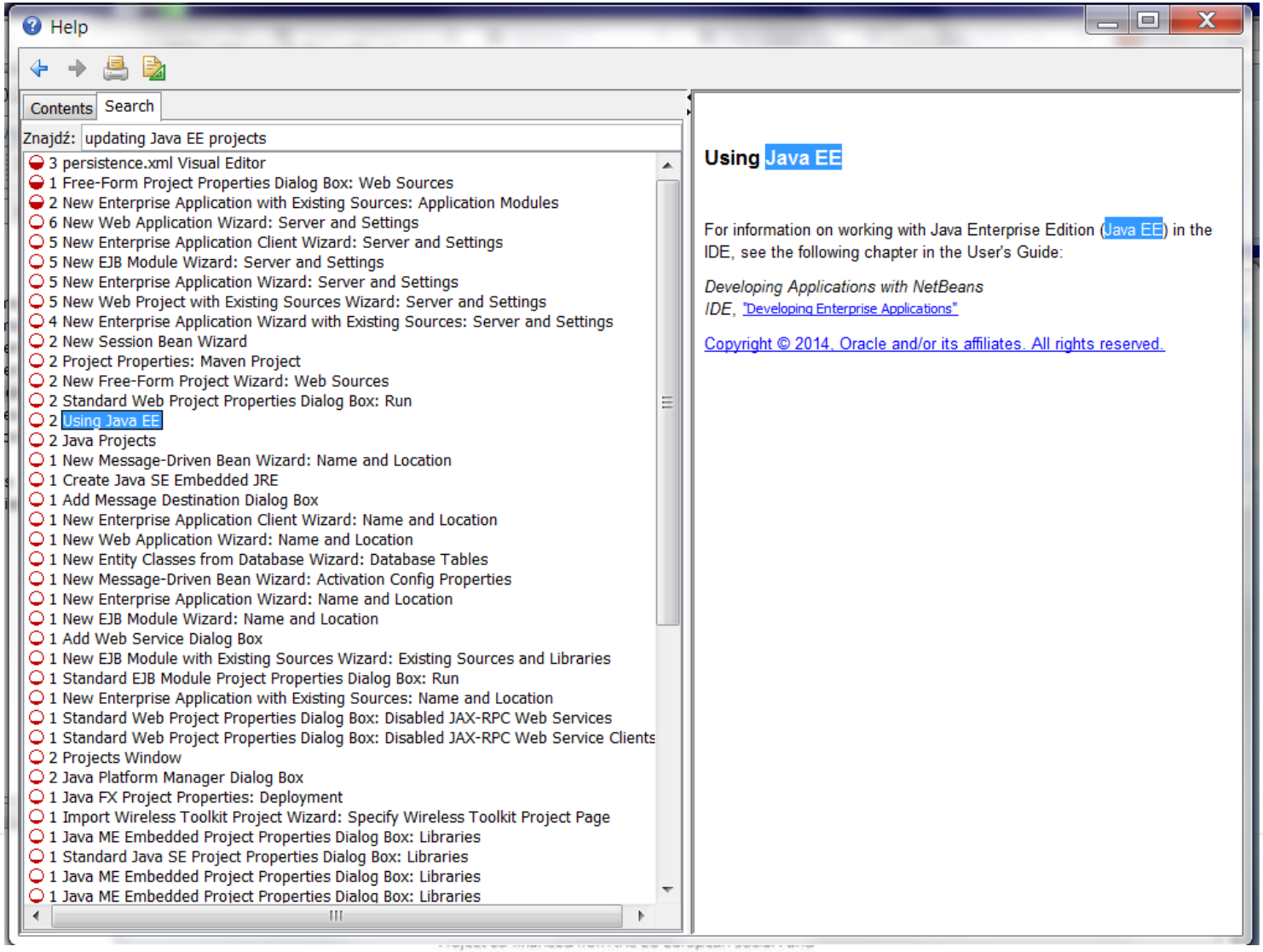
7.2. The **Library_client-ejb** form to adding the new books of the selected title, as the application data.



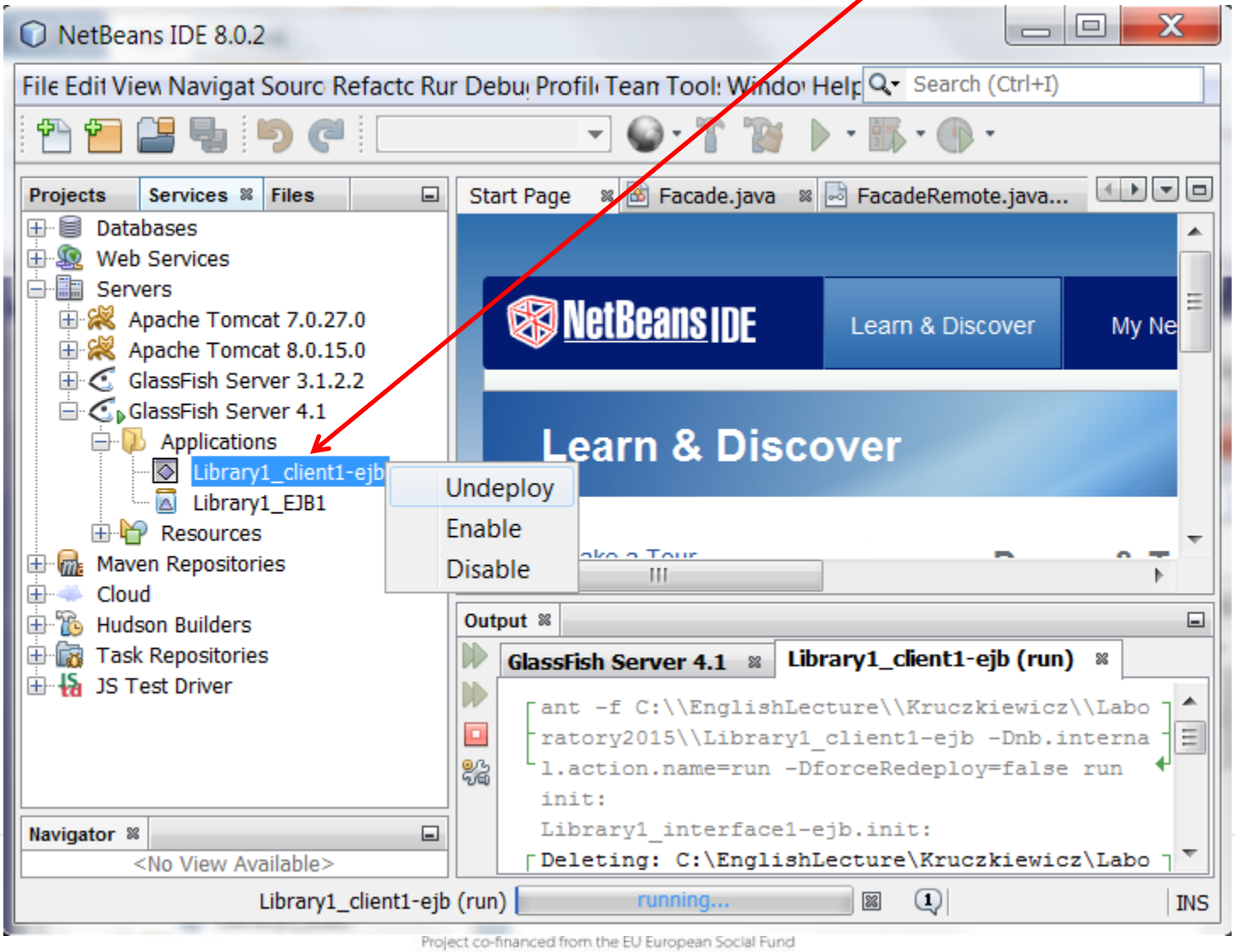
7.3. Restored data from database, after again opening EE application with the one kind of client:
Library_client-ejb as the Enterprise Application client (below)



8. /8.1. The help information about development of Java EE projects



8.2. The closing or before the update of Enterprise application – after undeploy process of of EE project components



8.3. The result after undeploying process of EE project components