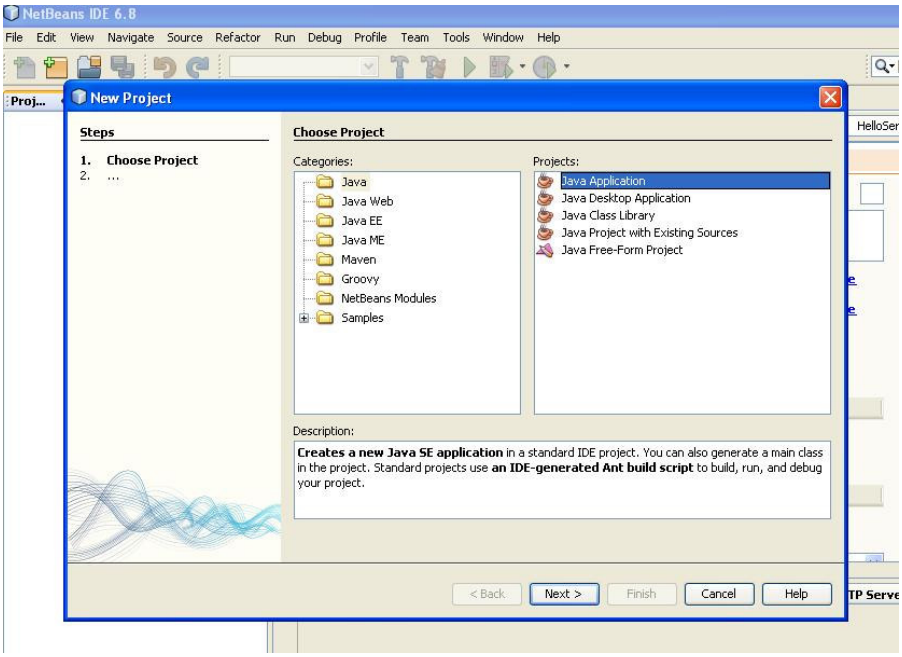
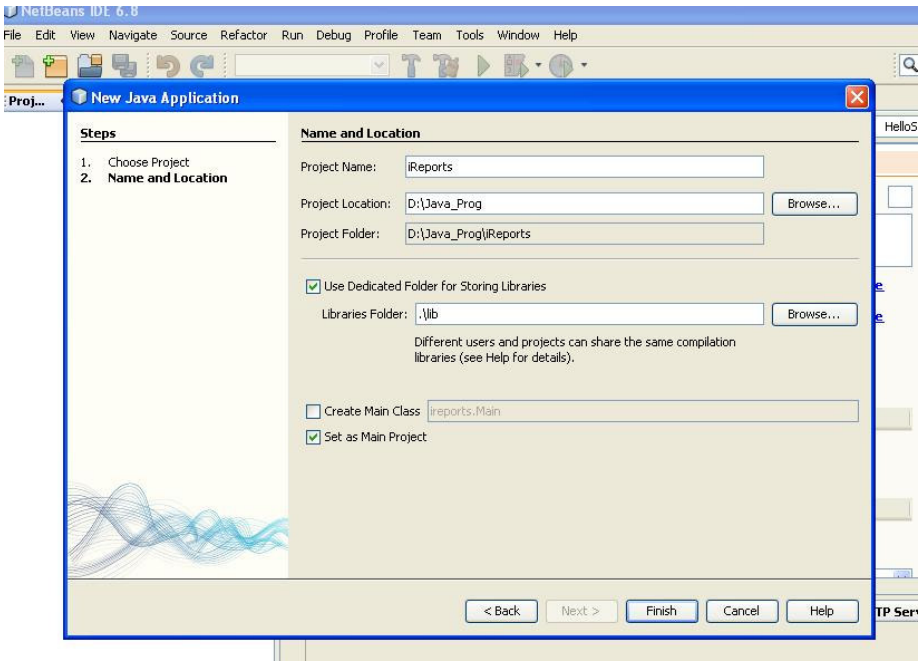
6 Call an iReport form a Java program.

This example describes how you display an iReport using a simple Java program.

6.1 Create a new Netbeans Project

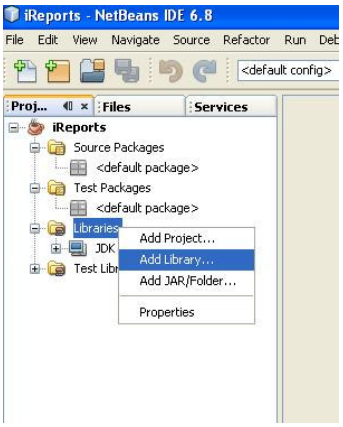




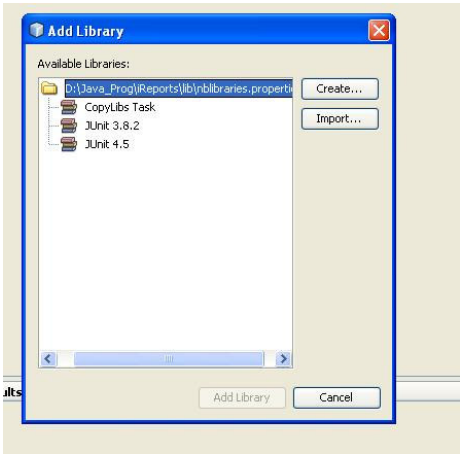
6.2 Import Required Libraries

6.2.1 Import MySQL Connector J

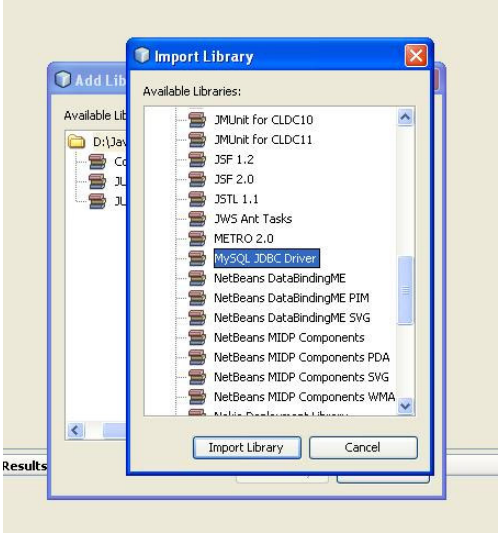
Select Libraries > Add Libraries



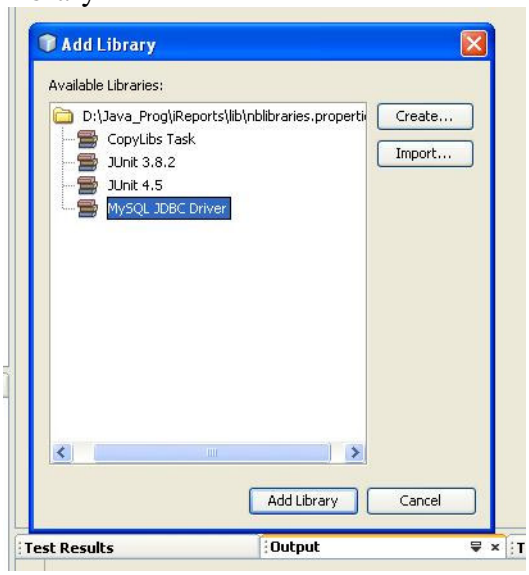
Select Import



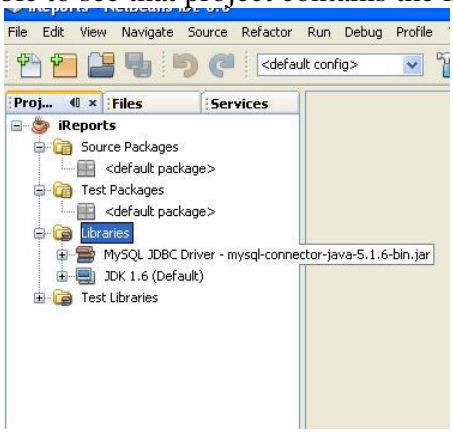
Select MySQL JDBC Driver and press Import Library button



Select Add Library

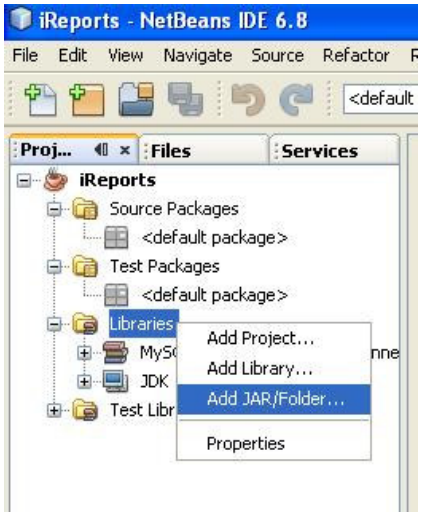


You must be able to see that project contains the library added



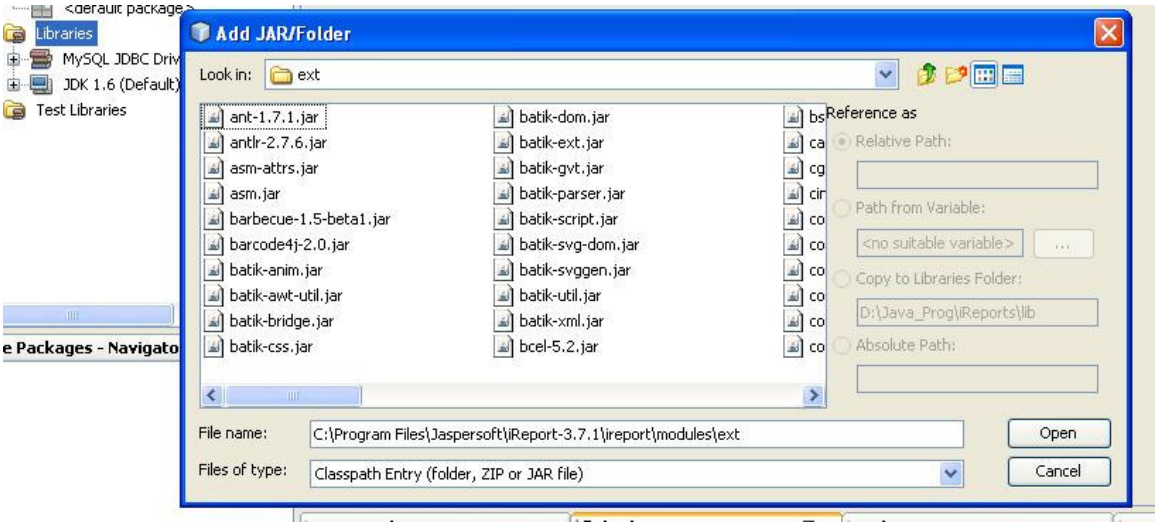
6.2.2 Import required Jaspersoft ".jar" archives.

Select Libraries > Add JAR/Folder



Browse the following folder
C:\Program Files\Jaspersoft\iReport-3.7.1\ireport\modules\ext

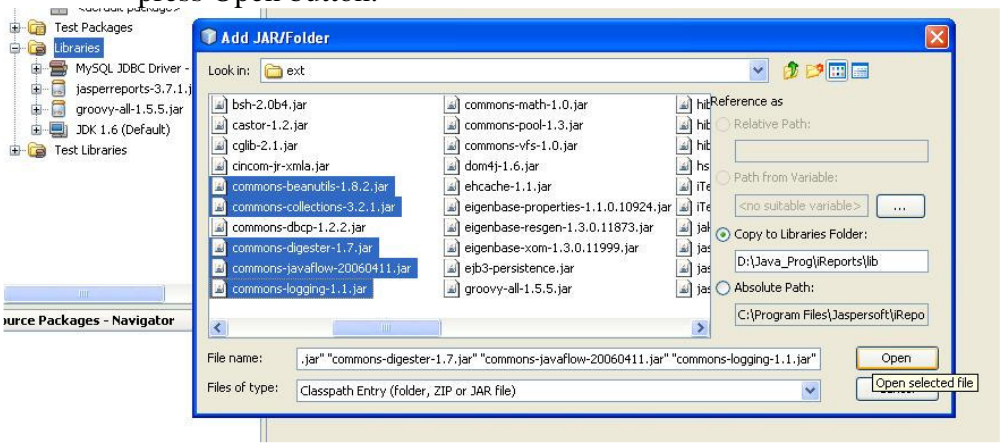
This folder contains all the libraries (.jar files) needed to generate iReprots. But we do not need all of them at the moment.



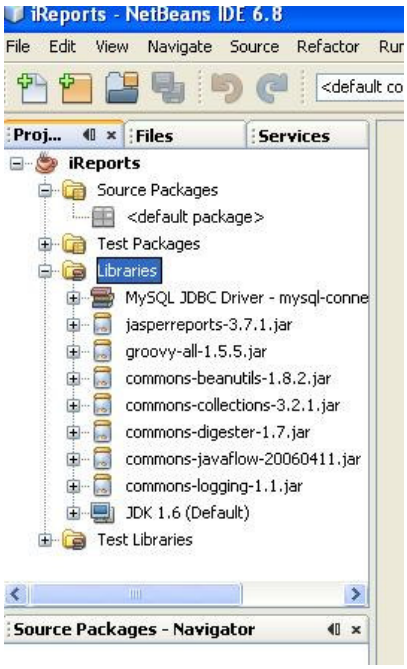
Import the following jar files

- jasperreports-3.7.1.jar
- groovy-all-1.5.5.jar
- commons-beanutils-1.8.2.jar
- commons-collections-3.2.1.jar
- commons-digester-1.7.jar
- commons-logging-1.1.jar
- commons-javaflow-20060411.jar

Select all of the above ".jar" files at once or select one at a time. Then press Open button.



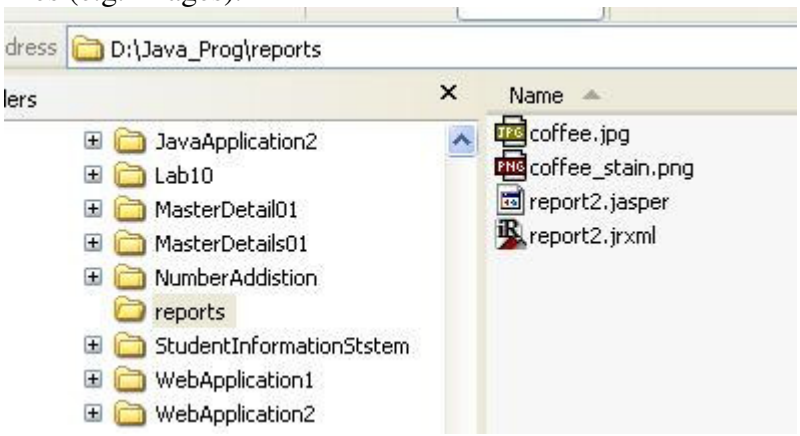
Once you finish importing, you must be able to see that your project contains all the ".jar" files added as Libraries



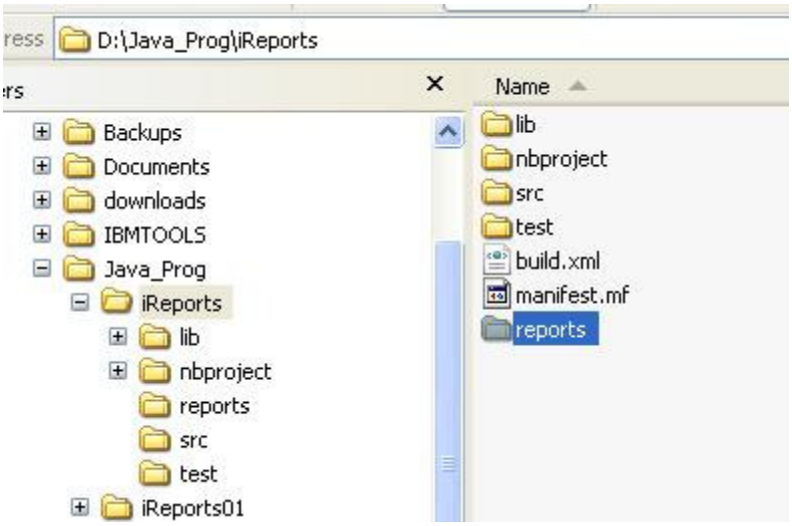
6.3 Copy the completed iReport to the NetBeans Project

Create a report with iReports and once you Pre-View the report with iReports, iReports will compile the report you created.

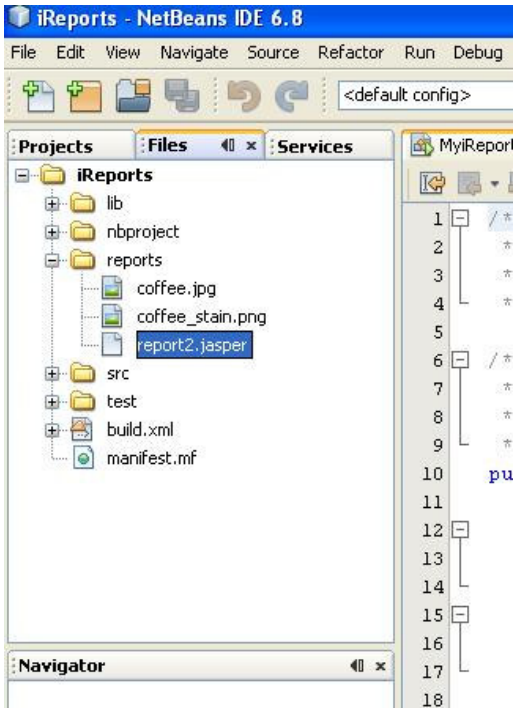
Once you browse the folder where you saved your reports you must be able to see a ".jrxml" file (the jasper report) , a ".jasper" file (compiled report) and other necessary files (e.g. images).



Now copy the complete report folder (in this example the folder "reports") in to your Netbeans project folder except the ".jrxml" file.

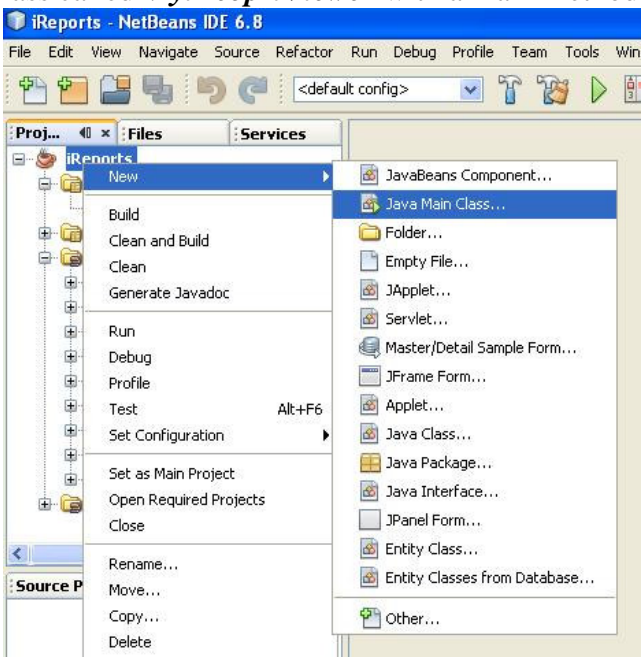


Now you must be able to see this reports folder and all the files in it once you browse files using ***Files*** tab in NetBeans IDE.

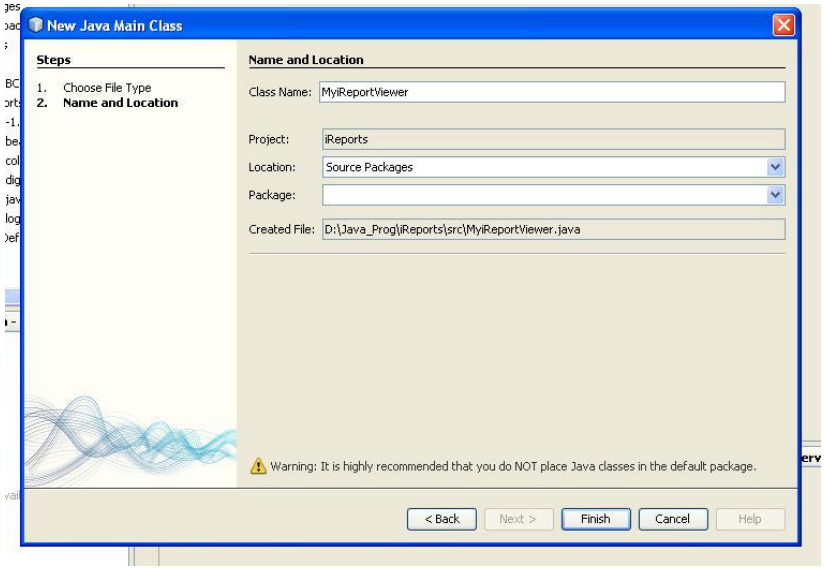


6.4 Add a new class to the project

Add a new class called ***MyiReoprtViewer*** with a main method to the project.



Select Finish



6.5 Complete the Java class.

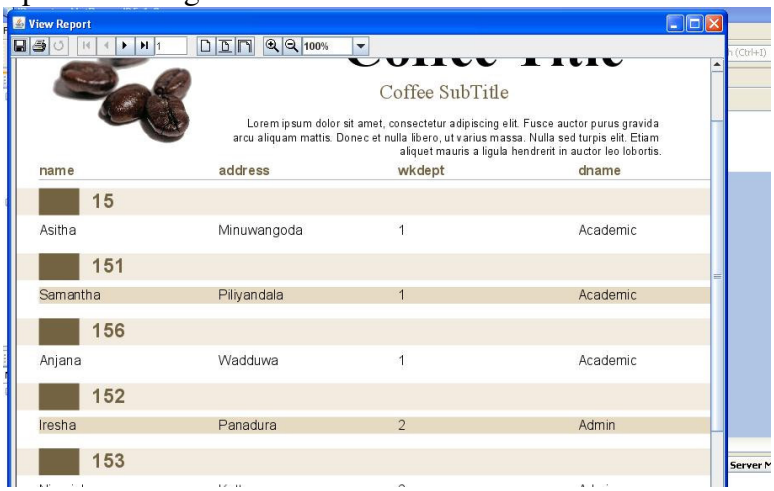
Complete the Java class as given below.

```
import javax.swing.*;
import java.awt.*;
import java.sql.*;
import java.util.*;
import java.io.*;
import net.sf.jasperreports.engine.*;
import net.sf.jasperreports.engine.design.*;
import net.sf.jasperreports.view.*;

public class MyiReportViewer extends JFrame{
   /*
    * You can use this constructor when you generate reports without
    * passing parametes
    */
   public MyiReportViewer(String fileName){
      this(fileName,null);
   }
     /*
    * You can use this constructor when you call an iReport by
    * passing parameters
    */
   public MyiReportViewer(String fileName,HashMap parameter) {
      //call a constructor in JFrame class
      super("View Report");

      //try block
      try  {
         // Instantiate MySQL Connector J driver
         Class.forName("com.mysql.jdbc.Driver").newInstance();
         // Open a connection to the database
         // use corect port, database instance, username and password
         Connection con =
               DriverManager.getConnection("jdbc:mysql://localhost:3306/test", "root", "test");

         //Way 1 to call iReposrts
         //JasperDesign jasperDesign = JasperManager.loadXmlDesign(fileName);
         //JasperReport jasperReport = JasperCompileManager.compileReport(jasperDesign);
         //JasperPrint print = JasperFillManager.fillReport(jasperReport, parameter, con);

         //Way 2 to call iReposrts
         // Here the filename must be a " .jrxml " file
         //JasperReport jasperReport = JasperCompileManager.compileReport(fileName);
         //JasperPrint print = JasperFillManager.fillReport(fileName, parameter, con);

          //Way 3 (Here the filename should be in .jasper extension i.e., the compiled report)
         JasperPrint print = JasperFillManager.fillReport(fileName, parameter, con);
```

```
            JRViewer viewer=new JRViewer(print);

          // get contaent pane
            Container c=getContentPane();
            // add the report to the frame
            c.add(viewer);
        }
        catch(ClassNotFoundException cnfe)       {
            cnfe.printStackTrace();
        }
        catch(SQLException sqle)  {
            sqle.printStackTrace();
        }
        catch(JRException jre) {
            jre.printStackTrace();
        }
        catch(Exception e){
            e.printStackTrace();
        }
        setBounds(10,10,600,500);
        setDefaultCloseOperation(DISPOSE_ON_CLOSE);
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
         // way 1 - call a report without passing passing parameters
        // Here you must make sure that you provide the correct path to .jsper file
        MyiReportViewer viewer=new MyiReportViewer("./reports/report2.jasper");

        // way 2 - call a report by passing passing parameters
        // you may pass any number of parameters. All the parameters must added
        // to the hash map
        //HashMap param=new HashMap();
        //param.put("reportParameterName",valueForTheParameter);
        //MyiReportViewer viewer=new MyiReportViewer("Report File Name With Extension",param);

        viewer.setVisible(true);
    }
}
```

6.6 Running the program

Once you run the program you must be able to see the report. Make sure that database is up and running.



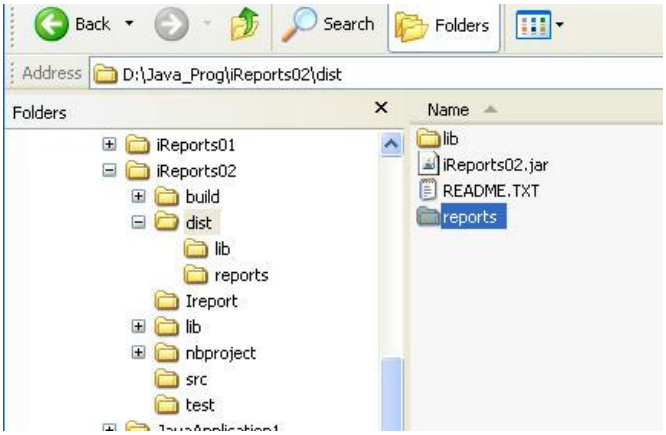
Note:

In this example we added a JRViewer to a JFram. You may add JRViewer to a JPannel or to a JformattedFrame .

6.7 Distribute the application.

Once you Build your application you will find the "iReports.jar" file inside ***"dist"*** folder. You may distribute this file. Inside the ***dist*** folder you will fine a sub folder called ***lib***. ***Lib*** folder contains all the libraries we imported.

You must manually copy the reports folder (which contains ".jasper files") to the dist folder.



You must distribute all the files in the dist folder in order to get your program executed properly.