Computer Science 2B

Practical Assignment 04

2017-08-22

Deadline: 2017-08-29 12h00 Marks: 100

This practical assignment must be uploaded to eve.uj.ac.za **before** 2017-08-29 12h00. Late or incorrect submissions **will not be accepted**, and will therefore not be marked. You are **not allowed to collaborate** with any other student.

The JDK has been installed on the laboratory computers along with the Eclipse IDE.

This practical will focus on UDP communication.

The University of Johannesburg wishes to track fitness information of connected students. Create a GUI Java client application which will send fitness data to a central server. Communication is not reliable as wireless access zones change according to the environment, therefore you are tasked with using UDP as the communication method. Messages sent to the server are in the following format: *STUDENT_NUMBER STEP_COUNT HEART_RATE*

Create a GUI Java server application which will process the fitness data. The server must store fitness information for each client which is identified by the *STUDENT_NUMBER*. This information is stored in a text file while the server is not running and loaded on startup. Positional data is displayed using a 3rd party library, JMathPlot. When the server is started the line plot displayed will show only the students which were saved to file. As the server receives more fitness data the line plot must be updated (update every 10 messages received). A refresh button will refresh the line plot immediately. Only the last 5 entries need to be stored on the server, for each student.

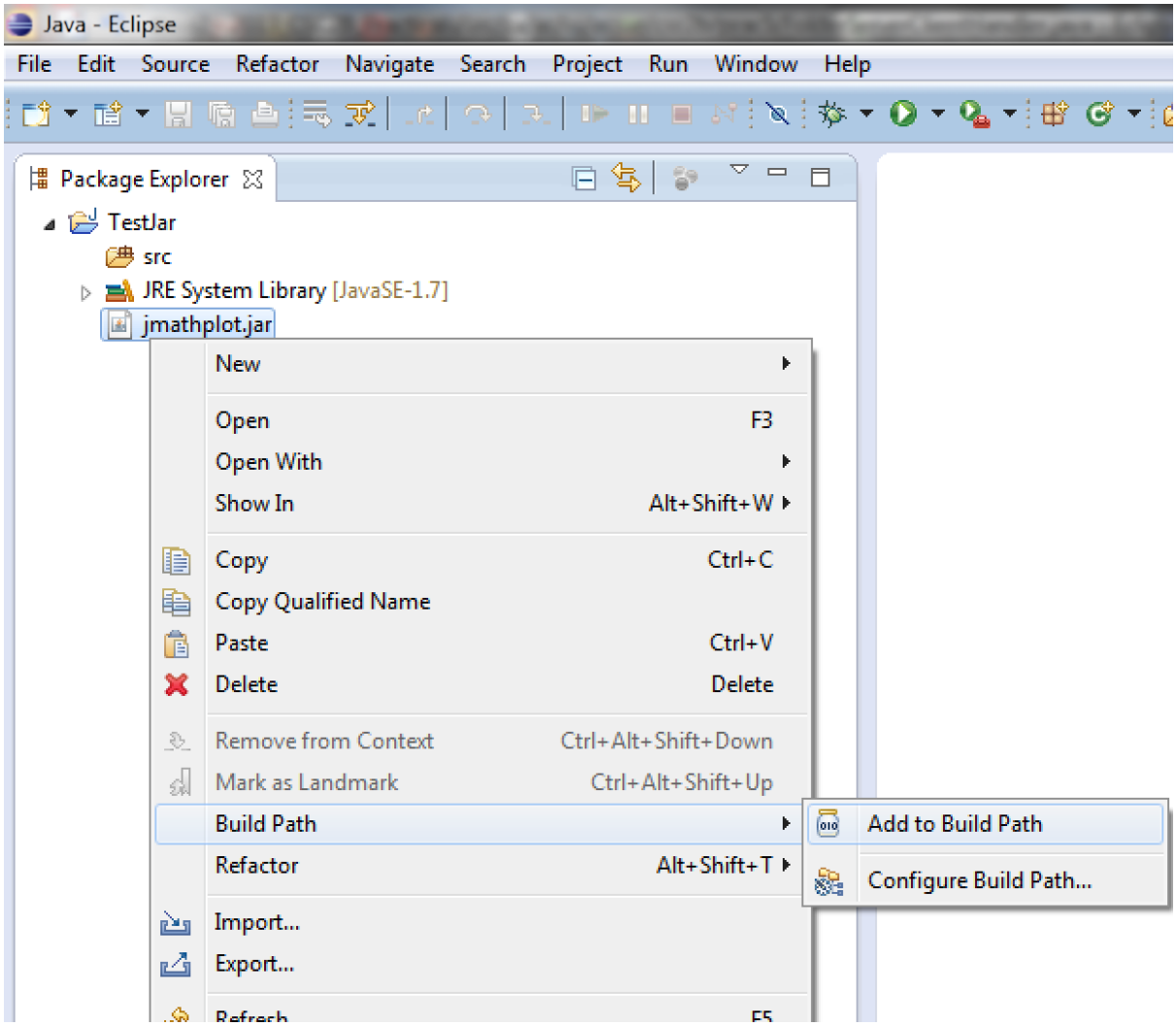

Figure 1: Adding a jar to the build path in Eclipse.

Using JMathPlot

In order to display the line plot you must add the jmathplot.jar file into your build path. Figure 1 shows how to add a jar to the build path of an eclipse project. If you are using the command line then you need to specify the jmathplot.jar file on the class path for both javac and java (the -cp option).

```
1 import java.awt.BorderLayout;
2 import java.awt.Color;
3 import java.awt.event.ActionEvent;
4 import java.awt.event.ActionListener;
5 import java.util.Random;
6 import javax.swing.JButton;
7 import javax.swing.JFrame;
8 import org.math.plot.Plot3DPanel;
9 public class TestPlot
10 {
11 private static Random r = new Random();
12 public static void main(String[] args)
13 {
14 final Plot2DPanel canvas = new Plot2DPanel("SOUTH");
15 canvas.setBackground(Color.BLACK);
16 JFrame frame = new JFrame();
17 frame.setLayout(new BorderLayout());
18 frame.add(canvas);
19 JButton btnRefesh = new JButton("Refresh");
20 frame.add(btnRefesh , BorderLayout.SOUTH);
21 btnRefesh.addActionListener(new ActionListener() {
22 private void addNewClientData(String name, Color colour)
23 {
24 double[] x = new double[100];
25 double[] y = new double[100];
26 x[0] = 0;
27 y[0] = r.nextDouble();
28 double change = 0;
29 for (int i = 1; i < 100; i++)
30 {
31 x[i] = i * 10;
32 change = r.nextBoolean() ? r.nextBoolean() ? r.nextDouble() :
33 -r.nextDouble() : 0;
34 y[i] = y[i - 1] + change;
35 }
36 canvas.addLinePlot(name, x, y);
37 }
38 @Override
39 public void actionPerformed(ActionEvent event)
40 {
41 canvas.removeAllPlots();
42 addNewClientData("John", Color.CYAN);
43 addNewClientData("Mary", Color.PINK);
44 addNewClientData("Thabo", Color.YELLOW);
45 addNewClientData("Tuli", Color.MAGENTA);
46 }
47 });
48 frame.setSize(800, 600);
49 frame.setLocationRelativeTo(null);
50 frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
51 frame.setVisible(true);
52 }
53 }
```

Marksheet

1. UDP Position Server: GUI [10]
2. UDP Position Server: Line plot for connected students [10]
3. UDP Position Server: Saving/Loading student fitness information [10]
4. UDP Position Server: Receiving UDP packets and processing information [10]
5. UDP Position Client: GUI [5]
6. UDP Position Client: Sending UDP packets and processing information. [10]
7. Coding convention (structure, layout) and commenting. [10]
8. Correct execution [35]