Sarcina:

**7.** Se cere elaborarea unei perechi de programe client-server cu următoarea funcţionalitate:

-         serverul reţine într-o structură de directoare proprii (pentru fiecare client un director) imagini în diferite formate (gif/jpg/etc...);

-         clientul poate acorda drepturi rw pentru directorul propriu şi altor clienţi.

-         clientul poate cere adăugarea, ştergerea unei/unor imagini în/din directorul serverului (respectiv directoarele pentru care are drepturi corespunzatoare).

 OBS. Se cere scrierea unui server concurent folosind thread-uri. Accesul la lista de drepturi şi la directoarele clienţilor va fi sincronizat. Adăugarea/ştergerea imaginilor se va face folosind thread-uri.

Codul:

# Server side

package greetingserver;

import java.net.\*;

import java.io.\*;

import java.sql.SQLException;

import java.awt.image.BufferedImage;

import javax.imageio.ImageIO;

import javax.swing.ImageIcon;

public class GreetingServer extends Thread

{

private ServerSocket serverSocket;

Socket server;

public GreetingServer(int port) throws IOException, SQLException, ClassNotFoundException, Exception

{

serverSocket = new ServerSocket(port);

serverSocket.setSoTimeout(180000);

}

public void run()

{

while(true)

{

try

{

server = serverSocket.accept();

DataInputStream din=new DataInputStream(server.getInputStream());

DataOutputStream dout=new DataOutputStream(server.getOutputStream());

dout.writeUTF("server: -i am greeting server");

dout.writeUTF("server:- hi! hello client");

System.out.println(din.readUTF());

System.out.println(din.readUTF());

BufferedImage img=ImageIO.read(ImageIO.createImageInputStream(server.getInputStream()));

System.out.println("Image received!!!!");

System.out.println(img);

//lblimg.setIcon(img);

}

catch(SocketTimeoutException st)

{

System.out.println("Socket timed out!");

break;

}

catch(IOException e)

{

e.printStackTrace();

break;

}

catch(Exception ex)

{

System.out.println(ex);

}

}

}

public static void main(String [] args) throws IOException, SQLException, ClassNotFoundException, Exception

{

//int port = Integer.parseInt(args[0]);

Thread t = new GreetingServer(6066);

t.start();

}

}

# Client side

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package greetingclient;

/\*\*

\*

\* @author Cornel

\*/

import java.net.\*;

import java.io.\*;

import java.awt.\*;

import java.awt.image.BufferedImage;

import javax.imageio.\*;

import javax.swing.ImageIcon;

import java.lang.\*;

public class GreetingClient

{

Image newimg;

//BufferedImage bImage = null;

byte[] bytes;

public static void main(String [] args)

{

BufferedImage bImage = null;

String serverName = "localhost";

int port = 6066;

try

{

File initialImage = new File("D://start\_01.JPG");

System.out.println("Connecting to " + serverName + " on port " + port);

Socket client = new Socket(serverName, port);

System.out.println("Just connected to " + client.getRemoteSocketAddress());

DataInputStream in=new DataInputStream(client.getInputStream());

System.out.println(in.readUTF());

System.out.println(in.readUTF());

DataOutputStream out = new DataOutputStream(client.getOutputStream());

out.writeUTF("Hello from " + client.getLocalSocketAddress());

out.writeUTF("client: hello to server");

ImageIcon img1=new ImageIcon("Ashish.jpg");

Image img = img1.getImage();

Image newimg = img.getScaledInstance(100, 120, java.awt.Image.SCALE\_SMOOTH);

ImageIcon newIcon = new ImageIcon(newimg);

bImage = ImageIO.read(initialImage);

ImageIO.write(bImage,"JPG",client.getOutputStream());

System.out.println("Image sent!!!!");

client.close();

}catch(IOException e)

{

e.printStackTrace();

}

}

}

# Rezultate



