**MODUL 14 PEMROGRAMAN BERORIENTASI OBJEK**

**NETWORKING BAGIAN 2**

1. Source Code

Github: <https://github.com/afrzl/Pemrograman-Berorientasi-Objek/tree/main/Week%2014/Praktikum/Latihan%20Server/src>

1. LatihanServer.java

import java.io.BufferedInputStream;  
import java.io.IOException;  
import java.io.ObjectInputStream;  
import java.io.PrintWriter;  
import java.net.ServerSocket;  
import java.net.Socket;  
import java.util.logging.Level;  
import java.util.logging.Logger;  
  
public class LatihanServer {  
 */\*\*  
 \* @param args the command line arguments  
 \*/* public static void main(String[] args) {  
 // *TODO code application logic here* int portNumber = 4444;  
 boolean listening = true;  
 try (  
 ServerSocket serverSocket = new ServerSocket(portNumber);  
 )  
 {  
 while(listening){  
 Socket clientSocket = serverSocket.accept();  
 new ServerThread(clientSocket).start();  
 }  
 } catch (IOException ex) {  
 Logger.*getLogger*(LatihanServer.class.getName()).log(Level.*SEVERE*, null, ex);  
 }  
 }  
}

1. LatihanClient.java

import java.io.BufferedOutputStream;  
import java.io.BufferedReader;  
import java.io.IOException;  
import java.io.InputStreamReader;  
import java.io.ObjectOutputStream;  
import java.io.PrintWriter;  
import java.net.Socket;  
import java.util.logging.Level;  
import java.util.logging.Logger;  
  
  
public class LatihanClient {  
 public static void main(String args[]){  
 String hostName = "localhost";  
 int portNumber = 4444;  
 try (  
 Socket echoSocket = new Socket(hostName, portNumber);  
 BufferedOutputStream bos = new BufferedOutputStream(echoSocket.getOutputStream());  
 ObjectOutputStream oos = new ObjectOutputStream(bos);  
 BufferedReader in = new BufferedReader(new InputStreamReader(echoSocket.getInputStream()));  
 BufferedReader stdIn = new BufferedReader(new InputStreamReader(System.*in*));  
 )  
 {  
 String msg;  
 while((msg = stdIn.readLine()) != null){  
 oos.writeObject(new Pesan("Budi", msg));  
 oos.flush();  
 System.*out*.println("Client receive: "+ in.readLine());  
 if(msg.equalsIgnoreCase("exit")) break;  
 }  
 } catch (IOException ex) {  
 Logger.*getLogger*(LatihanClient.class.getName()).log(Level.*SEVERE*, null, ex);  
 }  
 }  
}

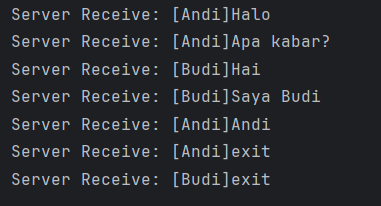
1. Pesan.java

import java.io.Serializable;  
  
public class Pesan implements Serializable {  
 private String nama;  
 private String pesan;  
  
 public Pesan(String nama, String pesan) {  
 this.nama = nama;  
 this.pesan = pesan;  
 }  
  
 @Override  
 public String toString() {  
 return "[" + nama + "]" + pesan;  
 }  
  
 public String getNama() {  
 return nama;  
 }  
  
 public void setNama(String nama) {  
 this.nama = nama;  
 }  
  
 public String getPesan() {  
 return pesan;  
 }  
  
 public void setPesan(String pesan) {  
 this.pesan = pesan;  
 }  
}

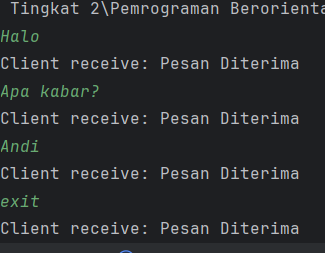
1. ServerThread.java

import java.io.BufferedInputStream;  
import java.io.IOException;  
import java.io.ObjectInputStream;  
import java.io.PrintWriter;  
import java.net.Socket;  
import java.util.logging.Level;  
import java.util.logging.Logger;  
  
public class ServerThread extends Thread {  
 private Socket clientSocket = null;  
 public ServerThread(Socket clientSocket) {  
 super();  
 this.clientSocket = clientSocket;  
 }  
  
 @Override  
 public void run(){  
 try (  
 PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);  
 BufferedInputStream bis = new BufferedInputStream(clientSocket.getInputStream());  
 ObjectInputStream ois = new ObjectInputStream(bis);){  
 Pesan pesan;  
 while((pesan = (Pesan) ois.readObject())!=null){  
 System.*out*.println("Server Receive: " + pesan.toString());  
 out.println("Pesan Diterima");  
 if(pesan.getPesan().equalsIgnoreCase("exit"))  
 break;  
 }  
 } catch (IOException ex) {  
 Logger.*getLogger*(ServerThread.class.getName()).log(Level.*SEVERE*, null, ex);  
 } catch (ClassNotFoundException ex) {  
 Logger.*getLogger*(ServerThread.class.getName()).log(Level.*SEVERE*, null, ex);  
 } finally {  
 if(clientSocket!=null){  
 try {  
 clientSocket.close();  
 } catch (IOException ex) {  
 Logger.*getLogger*(ServerThread.class.getName()).log(Level.*SEVERE*, null, ex);  
 }  
 }  
 }  
 }  
}

1. Output
2. Server



1. Client 1 (Andi)



1. Client 2 (Budi)

