**LAPORAN PRAKTIKUM**

**PEMROGRAMAN JARINGAN KOMPUTER**

****

**DISUSUN OLEH :**

Nama : Nasywa Deby Azanna

Nim : 2022903430048

Kelas : TRKJ 2B

Jurusan : Teknologi Informasi dan Komputer

Program Studi : Teknologi Rekayasa Komputer Jaringan

Dosen Pembimbing : Umri Erdiansyah, S.Kom., M.Kom

**JURUSAN TEKNOLOGI INFORMASI DAN KOMUNIKASI**

**PRODI TEKNOLOGI REKAYASA KOMPUTER DAN JARINGAN**

**POLITEKNIK NEGERI LHOKSEUMAWE**

**TAHUN 2022/2023**

**LEMBARAN PENGESAHAN**

Laporan Yang Berjudul : Pemrograman Jaringan Komputer

Disusun Oleh : Nasywa Deby Azanna

NIM : 2022903430048

Jurusan : Teknologi Informasi & Komputer

Program Studi : Teknologi Rekayasa Komputer Jaringan

Mata Kuliah : Network Programing

Tabel Penilaian : :

Mengetahui,

Dosen Pembimbing, Penyusun,

Umri Erdiansyah, S.Kom., M.Kom. Nasywa Deby Azanna

NIP 199210132022031003 NIM. 2022903430048

**I. Teori**

Java socket programming menyediakan kelas-kelas untuk berbagi data

diantara perangkat yang berbeda. Sebuah socket merupakan endpoint diantara dua

buah perangkat dalam berkomunikasi. Dalam aplikasi send file ini ada beberapa

kelas java yang dipakai yaitu File, FileInputStream, FileOutputStream,

InputStream, dan OutputStream.

1. Kelas File java mewakili dari sekumpulan file dan alamat direktori secara abstrak. Kelas ini digunakan untuk membuat file, direktori, pencarian file, penghapusan file dan lain-lain.
2. Kelas FileInputStream Berisi inputan byte dari sebuah file dalam sebuah file

system.

1. Kelas FileOutputStream Merupakan sebuah outpu stream untuk menuliskan

data ke dalam sebuah file atau sebuah filedeskriptor.

1. Kelas InputStream merupakan sebuah super kelas yang berisi beberapa kelas

yang mewakili dari sebuah input stream dari suatu byte.

1. Kelas OutputStream merupakan sebuah super kelas yang berisi beberapa kelas

yang mewakili dari sebuah output stream dari suatu byte.

**II. Alat/Bahan**

Alat dan bahan yang diperlukan dalam praktikum ini adalah perangkat hardware dan sofware yaitu:

1. Komputer/Labtop
2. NetBeans IDE (Integrated Development Environment
3. Java SE Development Kit 13

**III. Prosedur Praktikum**

Prosedur praktikum untuk kirim pesan dengan menggunakan protokol UDP adalah sebagai berikut.

1. Siapkan komputer/labtop

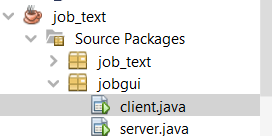
2. Siapkann Sistem operasi linux mint

3. Bukalah aplikasi netbeans

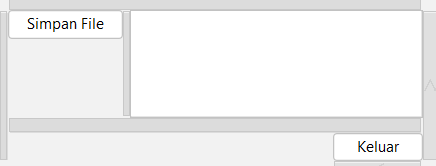
1. Buatlah package baru dengan nama Jobgui di dalam job\_text. Dalam

Package Jobgui buatlah dua buah jframe form dengan nama client.java dan

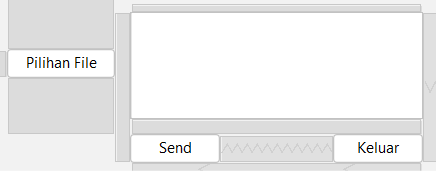
server .



1. Buatlah desain pada server jframe form seperti ini



Dan pada client seperti di bawah ini:



1. Buatlah kodingan untuk server seperti di bawah ini.

import javax.swing.JFileChooser;

import javax.swing.JOptionPane;

import java.io.File;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

import java.net.ServerSocket;

import java.net.Socket;

import java.util.logging.Level;

import java.util.logging.Logger;

public class server extends javax.swing.JFrame {

File pilihanFile;

ServerSocket serv;

Socket sock;

public server() {

initComponents();

}

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jButton1 = new javax.swing.JButton();

jScrollPane1 = new javax.swing.JScrollPane();

jTextArea1 = new javax.swing.JTextArea();

jButton2 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

jButton1.setText("Simpan File");

jButton1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton1ActionPerformed(evt);

}

});

jTextArea1.setColumns(20);

jTextArea1.setRows(5);

jScrollPane1.setViewportView(jTextArea1);

jButton2.setText("Keluar");

jButton2.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton2ActionPerformed(evt);

}

});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(jButton2)

.addGroup(layout.createSequentialGroup()

.addComponent(jButton1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addContainerGap(60, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(82, 82, 82)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(jButton1))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(jButton2)

.addContainerGap(92, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

JFileChooser pilihSimpan = new JFileChooser();

pilihSimpan.setCurrentDirectory(new File("C:\\Users\\dikis\\Documents\\SEMUA CODING"));

int returnVal = pilihSimpan.showSaveDialog(null);

if (returnVal == JFileChooser.APPROVE\_OPTION) {

pilihanFile = pilihSimpan.getSelectedFile();

jTextArea1.setText(pilihanFile.getAbsolutePath());

String masukanport = JOptionPane.showInputDialog("Masukkan port: ");

int port = Integer.parseInt(masukanport);

try (ServerSocket serv = new ServerSocket(port);

Socket sock = serv.accept();

InputStream in = sock.getInputStream();

OutputStream out = new FileOutputStream(pilihanFile.getAbsolutePath())) {

byte[] bytes = new byte[16 \* 1024];

int loop;

while ((loop = in.read(bytes)) > 0) {

out.write(bytes, 0, loop);

}

JOptionPane.showMessageDialog(null, "Pesan File Telah Disimpan");

} catch (IOException ex) {

Logger.getLogger(server.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try {

serv.close();

sock.close();

System.exit(0);

} catch (IOException ex) {

Logger.getLogger(server.class.getName()).log(Level.SEVERE, null, ex);

}

}

public static void main(String args[]) {

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new server().setVisible(true);

}

});

}

1. Buatkan kodingan pada client seperti di bawah.

import java.io.File;

import java.net.Socket;

import javax.swing.JFileChooser;

import javax.swing.JOptionPane;

import java.io.File;

import java.io.FileInputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

import java.net.Socket;

import java.util.logging.Level;

import java.util.logging.Logger;

public class client extends javax.swing.JFrame {

File pilihanFile;

Socket sock;

public client() {

initComponents();

}

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

jButton1 = new javax.swing.JButton();

jButton2 = new javax.swing.JButton();

jButton3 = new javax.swing.JButton();

jScrollPane1 = new javax.swing.JScrollPane();

jTextArea1 = new javax.swing.JTextArea();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

jButton1.setText("Pilihan File");

jButton1.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton1ActionPerformed(evt);

}

});

jButton2.setText("Send");

jButton2.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton2ActionPerformed(evt);

}

});

jButton3.setText("Keluar");

jButton3.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

jButton3ActionPerformed(evt);

}

});

jTextArea1.setColumns(20);

jTextArea1.setRows(5);

jScrollPane1.setViewportView(jTextArea1);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(46, 46, 46)

.addComponent(jButton1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addGroup(layout.createSequentialGroup()

.addComponent(jButton2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(jButton3))

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addContainerGap(36, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(119, 119, 119)

.addComponent(jButton1)

.addGap(45, 45, 45))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addContainerGap()

.addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jButton2)

.addComponent(jButton3))

.addContainerGap(90, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

JFileChooser pilihAlamatFile = new JFileChooser();

int hasilPilihan = pilihAlamatFile.showOpenDialog(null);

if (hasilPilihan == JFileChooser.APPROVE\_OPTION) {

pilihanFile = pilihAlamatFile.getSelectedFile();

jTextArea1.setText(pilihanFile.getAbsolutePath());

}

}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

JFileChooser pilihAlamatFile = new JFileChooser();

pilihAlamatFile.showOpenDialog(null);

pilihanFile = pilihAlamatFile.getSelectedFile();

jTextArea1.setText(pilihanFile.getAbsolutePath());

String alamatIP = JOptionPane.showInputDialog("Masukkan Alamat IP Komputer Server");

String masukanPort = JOptionPane.showInputDialog("Masukkan Port Server");

int port = Integer.parseInt(masukanPort);

try {

Socket sock = new Socket(alamatIP, port);

File file = new File(pilihanFile.getAbsolutePath());

long panjangFile = file.length();

byte[] bytes = new byte[16 \* 1024];

InputStream in = new FileInputStream(file);

OutputStream out = sock.getOutputStream();

int count;

while ((count = in.read(bytes)) > 0) {

out.write(bytes, 0, count);

}

JOptionPane.showMessageDialog(null, "Pesan File Telah Terkirim");

sock.close();

} catch (IOException ex) {

Logger.getLogger(client.class.getName()).log(Level.SEVERE, null, ex);

}

}

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try {

sock.close();

System.exit(0);

} catch (IOException ex) {

Logger.getLogger(client.class.getName()).log(Level.SEVERE,

null, ex);

}

}

public static void main(String args[]) {

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

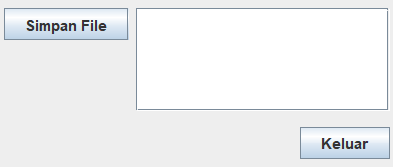
new client().setVisible(true);

}

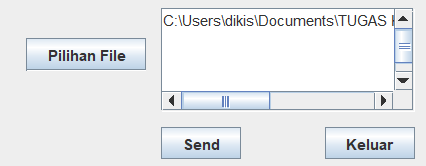
});

}

1. Jalankan pada server.



1. Jalankan pada client.



**KESIMPULAN**

Dari laporan di atas, kita dapat menyimpulkan beberapa poin penting terkait dengan Java socket programming dan aplikasi pengiriman file antara server dan client:

1. **Java Socket Programming:** Java menyediakan kelas-kelas untuk mendukung komunikasi antara perangkat yang berbeda menggunakan socket. Socket merupakan endpoint dalam komunikasi antara dua perangkat.
2. **Kelas-kelas Java yang Digunakan:**
   * **File:** Mewakili sekumpulan file dan alamat direktori secara abstrak. Digunakan untuk operasi seperti pembuatan file, pencarian file, dan penghapusan file.
   * **FileInputStream:** Berisi input byte dari sebuah file dalam file system.
   * **FileOutputStream:** Merupakan output stream untuk menuliskan data ke dalam sebuah file atau file descriptor.
   * **InputStream dan OutputStream:** Kelas-kelas ini merupakan superclass yang berisi beberapa kelas yang mewakili input stream dan output stream dari suatu byte.
3. **Alat/Bahan dan Prosedur Praktikum:**
   * Alat/bahan yang diperlukan meliputi komputer/laptop, NetBeans IDE, dan Java SE Development Kit 13.
   * Prosedur praktikum mencakup persiapan komputer/laptop, instalasi sistem operasi Linux Mint, pembuatan package dan JFrame form menggunakan NetBeans IDE.
4. **Implementasi Server dan Client:**
   * Pembuatan dua JFrame form, yaitu **server** dan **client**, untuk aplikasi pengiriman file.
   * Pada server, terdapat tombol untuk menyimpan file, pengaturan port, dan fungsi untuk menerima file dari client.
   * Pada client, terdapat tombol untuk memilih file, pengaturan alamat IP dan port server, dan fungsi untuk mengirim file ke server.
5. **Penggunaan Kelas-kelas I/O dan Socket:**
   * Pada server, menggunakan **ServerSocket** untuk menerima koneksi dari client, **InputStream** untuk menerima data, dan **OutputStream** untuk menulis data ke file.
   * Pada client, menggunakan **Socket** untuk membuat koneksi ke server, **InputStream** untuk membaca file, dan **OutputStream** untuk mengirim data ke server.
6. **Kesimpulan Praktikum:**
   * Aplikasi ini memanfaatkan konsep socket programming untuk mengirim file antara server dan client.
   * Penggunaan kelas-kelas I/O seperti **FileInputStream** dan **FileOutputStream** memungkinkan manipulasi file.
   * Keseluruhan aplikasi memanfaatkan konsep client-server dengan menggunakan protokol UDP.

Dengan demikian, praktikum ini memberikan pemahaman tentang implementasi dasar socket programming dalam Java dan bagaimana aplikasi pengiriman file antara server dan client dapat diimplementasikan dengan menggunakan kelas-kelas yang disediakan oleh Java.

Top of Form