COMSATS University Islamabad

Department of Computer Science

SOFTWARE CONSTRUCTION

Project Documentation

Submitted To:

Isma ul Hassan

Submitted by:

Maarooshaa Asim Maalik

Ikram ul Haq Qureshi

M. Zakaria Nazir

Registration No.:

FA16-BSE-067

FA16-BSE-054

FA16-BSE-147

1. Project Description

This is a client /server application.
File Transfer Application has a File Server and File Client. When the File Server is running, File Client can request a file from the server. File server will look in the Filebank, if the file exists, it will be simply send the file and the client will be able to download it to the specified directory, however, if the file does not exist in the server's directory, a simple message will be written to the console that "The File is not found in the server."
A client can ask for only one file in one connection.
The server can connect to multiple clients once it is run, using threads.

2. Concurrent Activities of your project

The project utilizes the concurrency by allowing multiple simultaneous connections at one time to the server.

The server starts a new thread for every client contacting it hence solve the issues of scalability and make the whole process less time consuming as the client will not have to wait for the server to be free to make a request.

3. Fundamental issues Faces in managing the changes in your project

- Race condition when creating variables
- Possibility of a deadlock occurring when assigning resources to multiple clients at the same time

4. Solution/Approaches used to handle those issues

- Ensured that there are no global variables that are used without sleep/lock functionality.

5. Original Code

```
/*
 * 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 FileServer;
```

```
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.net.Socket;

/**
 *
 * @author stygianlgdonic
 */
public class ClientThread extends Thread {

    private Socket s;
    private String filename;
    private DataInputStream din;
    private DataOutputStream dout;

    public ClientThread(Socket s, String filename,
            DataInputStream din, DataOutputStream dout) {
        this.s = s;
        this.filename = filename;
        this.din = din;
        this.dout = dout;
    }

    @Override
    public void run() {
            try {
```

```
System.out.println("Sending File: " + filename);

byte check;

//server's directory to search from
File f = new File("C:\\Users\\mbird\\Desktop\\FileBank\\"
        + filename);

if (f.exists() == true) {

    // if requested file exists
    check = 1;
    dout.writeByte(check);
    dout.flush();

} else {

    // if requested file exists
    check = 0;
    dout.writeByte(check);
    dout.flush();
    System.out.println("Error: FileNotFound");

    closeConnection();

}

dout.writeUTF(filename);
```

```
    dout.flush();

    // For Sending Server's File
    FileInputStream fin = new FileInputStream(f);
    long sz = (int) f.length();
    byte b[] = new byte[1024];
    int read;
    dout.writeUTF(Long.toString(sz));
    dout.flush();
    System.out.println("Size: " + sz);
    System.out.println("Buf size: " + s.getReceiveBufferSize());

    while ((read = fin.read(b)) != -1) {
        dout.write(b, 0, read);
        dout.flush();
    }
    fin.close();

    System.out.println("..ok");
    dout.flush();

    System.out.println("Send Complete");
    dout.flush();
    closeConnection(); // Closing Connection
} catch (Exception e) {
    e.printStackTrace();
    System.out.println("An error occured");

}
```

```
    }

    public void closeConnection() throws IOException {
        System.out.println("Connection has been closed with "
                + s.getInetAddress().toString());
        s.close();
        dout.close();
        din.close();
    }
}
/*
 * 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 FileClient;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.EOFException;
import java.io.File;
import java.io.FileOutputStream;
import java.net.Socket;
import java.util.Scanner;

/**
 *
 * @author stygianlgdonic
 */
public class MainClient {

    public static void main(String args[]) throws Exception {

        // Client's file request
        System.out.println("Enter File Name: ");
        Scanner sc = new Scanner(System.in);
        String filename1 = sc.nextLine();

        // Server's address
        String address = "";
        System.out.println("Enter Server Address: ");
        address = sc.nextLine();

        Socket s = new Socket(address, 5000);
        //connection setup

        DataInputStream din = new DataInputStream(s.getInputStream());
        DataOutputStream dout = new DataOutputStream(s.getOutputStream());

        dout.writeUTF(filename1);

        String filename = "";

        try {
            dout.flush();

            byte check = din.readByte();
```

```
if (check == 1) {
  // if file found Start recieving

  filename = din.readUTF();
  System.out.println("Receving file: " + filename);
  filename = "client" + filename;
  System.out.println("Saving as file: " + filename);

  long sz = Long.parseLong(din.readUTF());
  System.out.println("File Size: " + (sz / (1024 * 1024)) + " MB");

  byte b[] = new byte[1024];
  System.out.println("Receving file..");

  // Client's save directory
  FileOutputStream fos = new FileOutputStream(new File(
      "C:\\Users\\mbird\\Desktop\\" + filename), true);

  long bytesRead;

  // Saving file to Client's save directory
  do {
    bytesRead = din.read(b, 0, b.length);
    fos.write(b, 0, b.length);
    System.out.println(".");
  } while (!(bytesRead < 1024));

  System.out.println("Completed");
  System.out.println("Closing Connection");
```

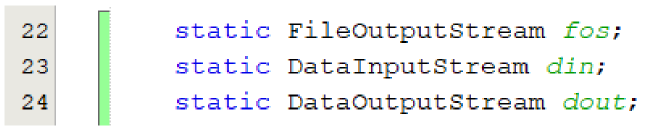
```
                fos.close();
                dout.close();
                s.close();
            } else {
                // if file not found Close Connection
                System.out.println("FileNotFound");
                System.out.println("Closing Connection");
                dout.close();
                din.close();
                s.close();
            }

        } catch (EOFException e) {
            System.out.println("Some Error Has occured");
            System.out.println("Closing Connection");
            dout.close();
            din.close();
            s.close();
        }
    }
}
/*
 * 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 FileServer;
```

```
import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.io.IOException;

import java.net.ServerSocket;

import java.net.Socket;


/**

 *

 * @author stygianlgdonic

 */

public class MainServer {


    public static void main(String args[]) throws IOException {


        // Creating socket on port 5000

        ServerSocket ss = new ServerSocket(5000);


        String filename;


        while (true) {


            Socket s = null;


            try {

                System.out.println("Waiting for request");


                s = ss.accept();


                DataInputStream din = new DataInputStream(s.getInputStream());
                DataOutputStream dout = new DataOutputStream(s.getOutputStream());

                filename = din.readUTF();
                System.out.println("Connected With " + s.getInetAddress().toString()
                        + "requesting for file:" + filename);

                // Initializing Thread
                Thread ct;
                ct = new ClientThread(s, filename, din, dout);
                ct.start(); // thread start

            } catch (Exception ex) {
                s.close(); // Closing Socket
                ex.printStackTrace();
            }
        }
    }
}
```

6. Refactored and Tuned Code

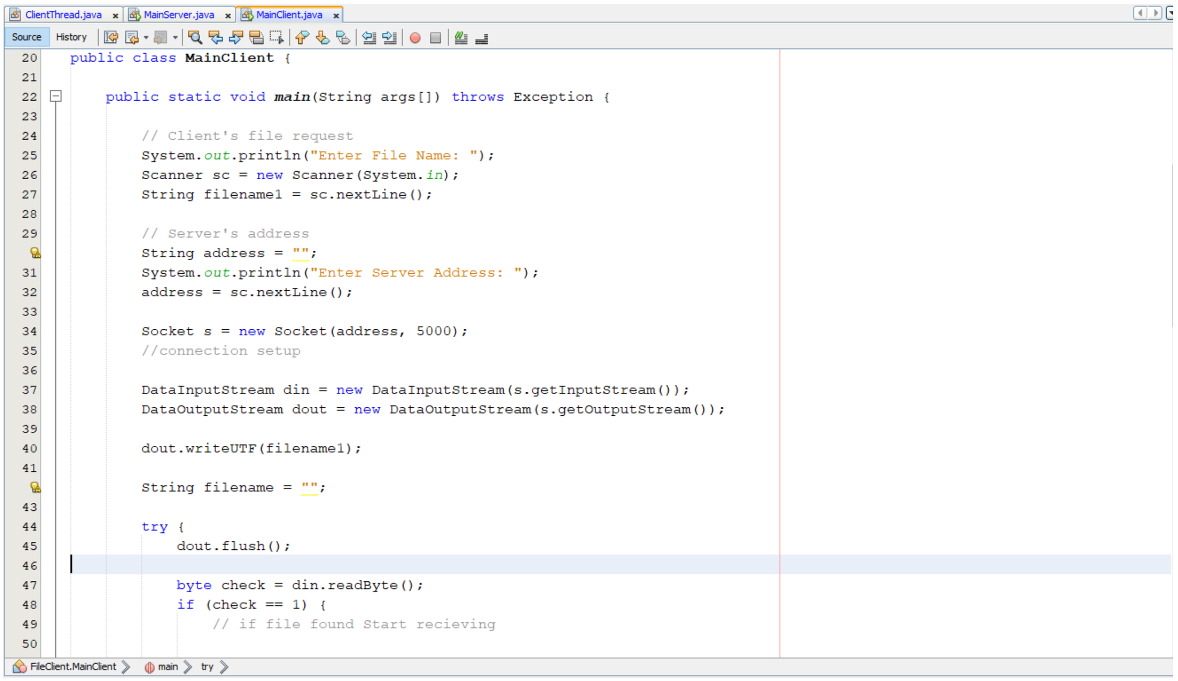
MainClient.java

Declaring as class variables so we can use them in methods

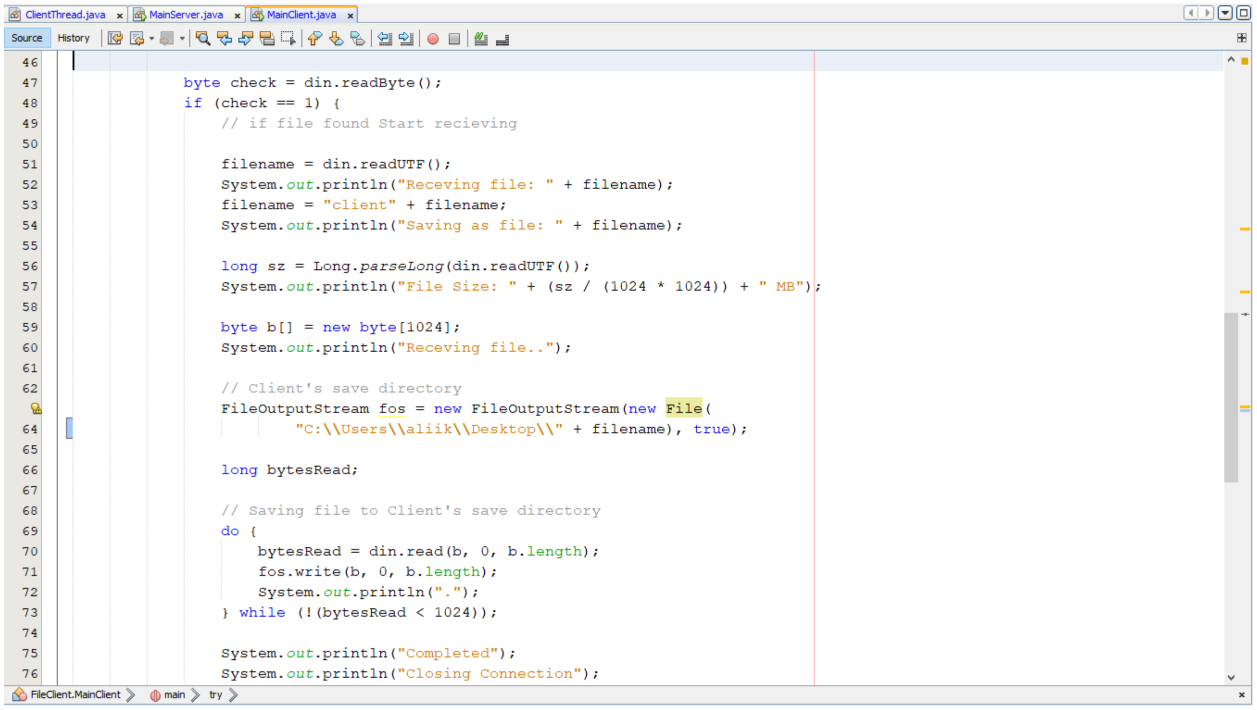


```
    static FileOutputStream fos;
    static DataInputStream din;
    static DataOutputStream dout;
```

Main method is performing 2 functions, i.e. connection setup and file receiving.

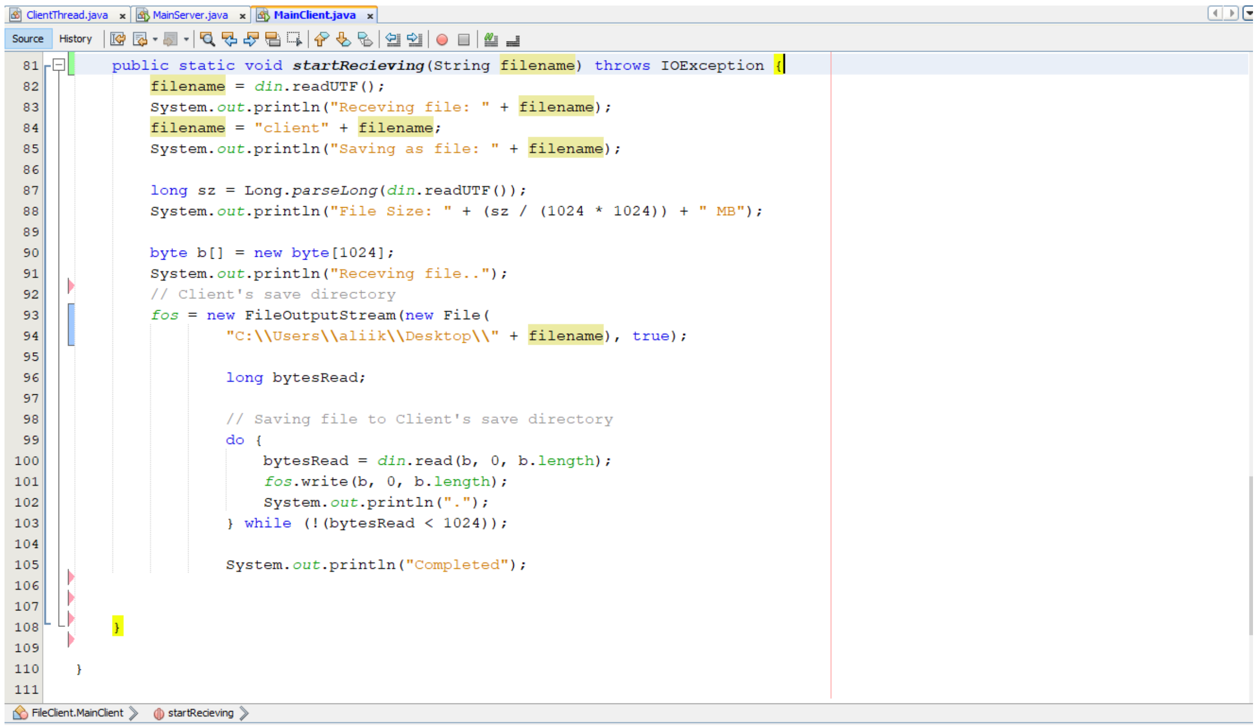


```
public class MainClient {

    public static void main(String args[]) throws Exception {

        // Client's file request
        System.out.println("Enter File Name: ");
        Scanner sc = new Scanner(System.in);
        String filename1 = sc.nextLine();

        // Server's address
        String address = "";
        System.out.println("Enter Server Address: ");
        address = sc.nextLine();

        Socket s = new Socket(address, 5000);
        //connection setup

        DataInputStream din = new DataInputStream(s.getInputStream());
        DataOutputStream dout = new DataOutputStream(s.getOutputStream());

        dout.writeUTF(filename1);

        String filename = "";

        try {
            dout.flush();

            byte check = din.readByte();
            if (check == 1) {
                // if file found Start recieving
```



```

            byte check = din.readByte();
            if (check == 1) {
                // if file found Start recieving

                filename = din.readUTF();
                System.out.println("Receving file: " + filename);
                filename = "client" + filename;
                System.out.println("Saving as file: " + filename);

                long sz = Long.parseLong(din.readUTF());
                System.out.println("File Size: " + (sz / (1024 * 1024)) + " MB");

                byte b[] = new byte[1024];
                System.out.println("Receving file..");

                // Client's save directory
                FileOutputStream fos = new FileOutputStream(new File(
                        "C:\\Users\\aliik\\Desktop\\" + filename), true);

                long bytesRead;

                // Saving file to Client's save directory
                do {
                    bytesRead = din.read(b, 0, b.length);
                    fos.write(b, 0, b.length);
                    System.out.println(".");
                } while (!(bytesRead < 1024));

                System.out.println("Completed");
                System.out.println("Closing Connection");
```

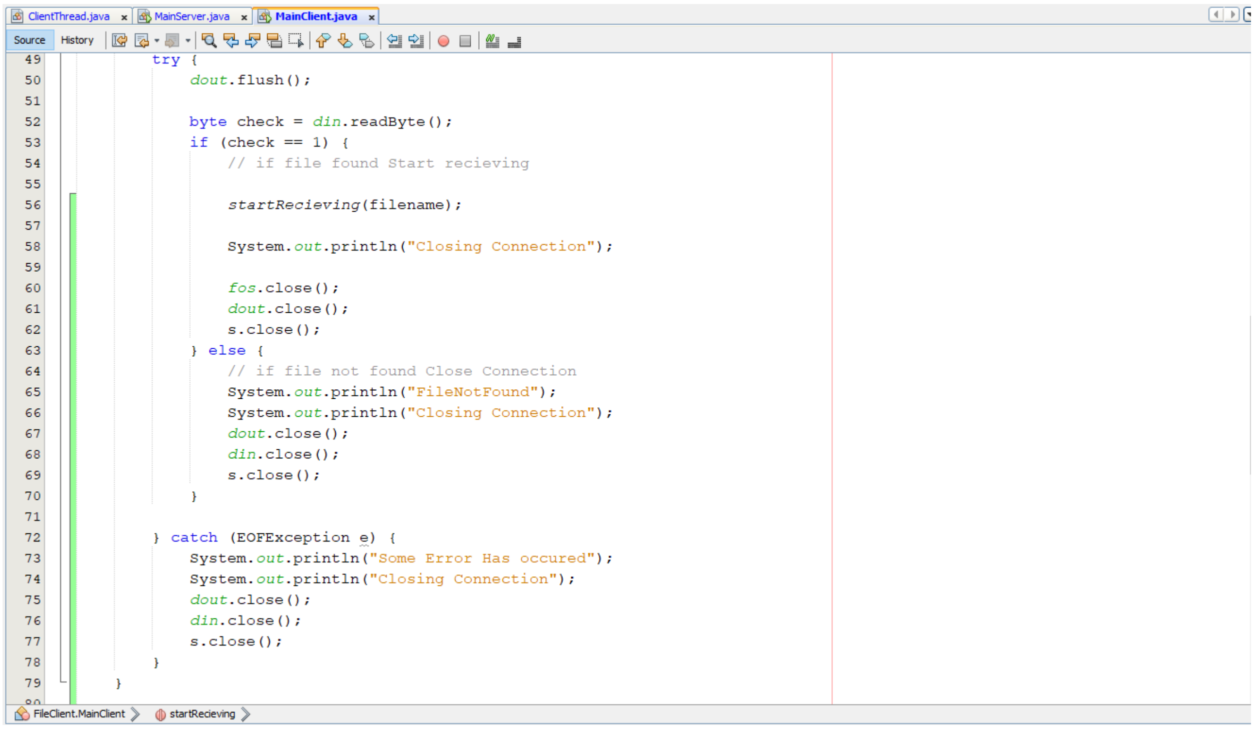
A method 'startRecieving' is created to perform the file receiving and is called from main


ClientThread.java × MainServer.java × MainClient.java ×

```
    public static void startRecieving(String filename) throws IOException {
        filename = din.readUTF();
        System.out.println("Receving file: " + filename);
        filename = "client" + filename;
        System.out.println("Saving as file: " + filename);

        long sz = Long.parseLong(din.readUTF());
        System.out.println("File Size: " + (sz / (1024 * 1024)) + " MB");

        byte b[] = new byte[1024];
        System.out.println("Receving file..");
        // Client's save directory
        fos = new FileOutputStream(new File(
                "C:\\Users\\aliik\\Desktop\\" + filename), true);

                long bytesRead;

                // Saving file to Client's save directory
                do {
                    bytesRead = din.read(b, 0, b.length);
                    fos.write(b, 0, b.length);
                    System.out.println(".");
                } while (!(bytesRead < 1024));

                System.out.println("Completed");


    }

}
```

FileClient.MainClient > startRecieving >

After creating method for receiving the try-catch in main method is not needed

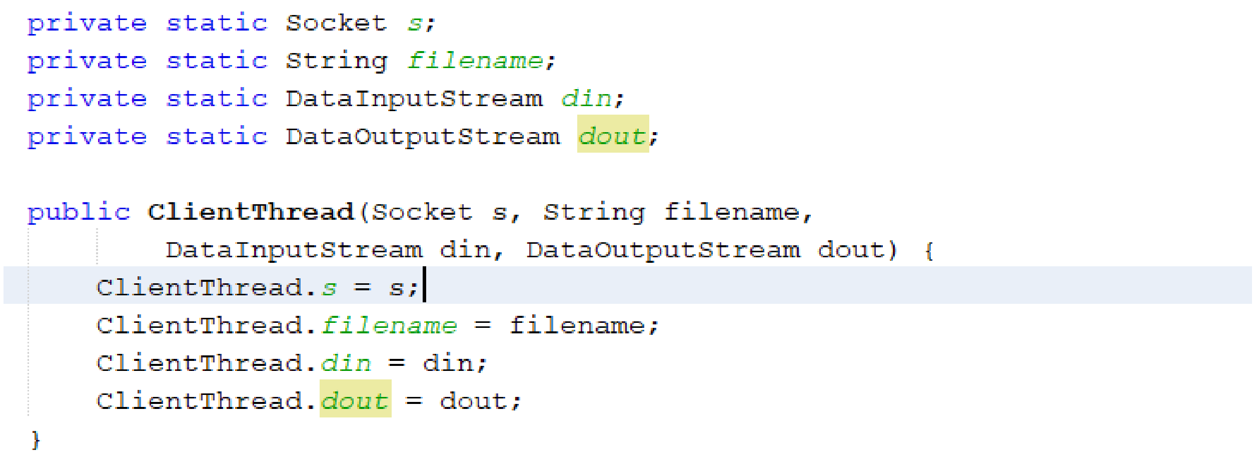

ClientThread.java × MainServer.java × MainClient.java ×

```
            try {
                dout.flush();

                byte check = din.readByte();
                if (check == 1) {
                    // if file found Start recieving

                    startRecieving(filename);

                    System.out.println("Closing Connection");

                    fos.close();
                    dout.close();
                    s.close();
                } else {
                    // if file not found Close Connection
                    System.out.println("FileNotFound");
                    System.out.println("Closing Connection");
                    dout.close();
                    din.close();
                    s.close();
                }

            } catch (EOFException e) {
                System.out.println("Some Error Has occured");
                System.out.println("Closing Connection");
                dout.close();
                din.close();
                s.close();
            }
        }
```

FileClient.MainClient > startRecieving >

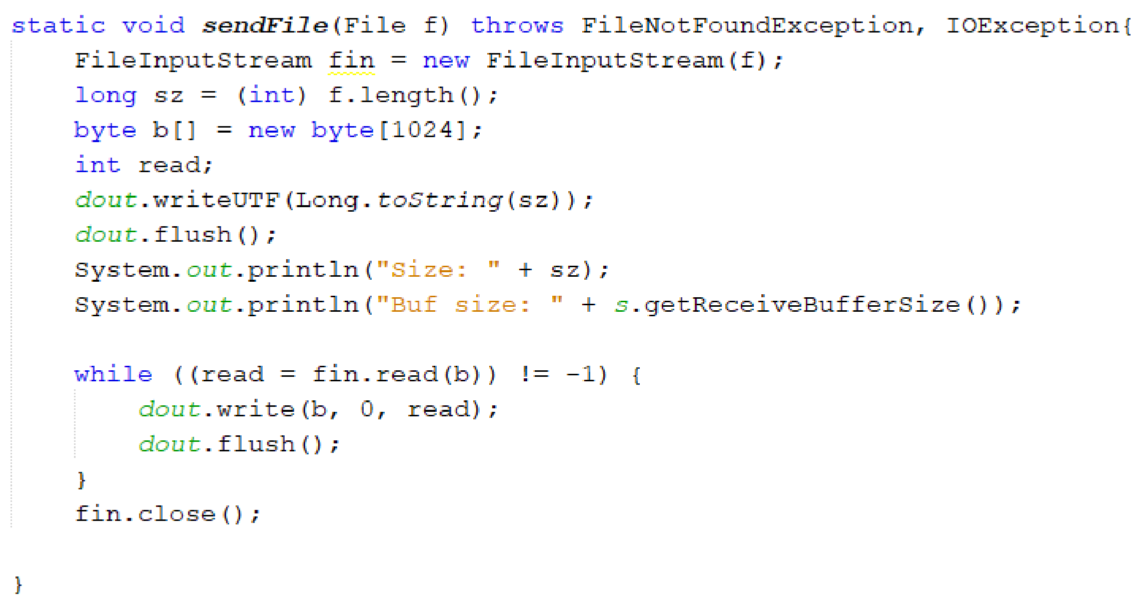
ClientThread.java

Variables declared as static to increase scope



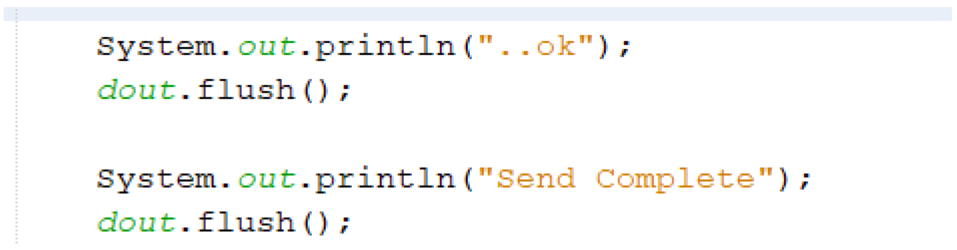
```
private static Socket s;
private static String filename;
private static DataInputStream din;
private static DataOutputStream dout;

public ClientThread(Socket s, String filename,
        DataInputStream din, DataOutputStream dout) {
    ClientThread.s = s;
    ClientThread.filename = filename;
    ClientThread.din = din;
    ClientThread.dout = dout;
}
```

Created a ‘sendFile’ method that would be called from main



```
static void sendFile(File f) throws FileNotFoundException, IOException{
    FileInputStream fin = new FileInputStream(f);
    long sz = (int) f.length();
    byte b[] = new byte[1024];
    int read;
    dout.writeUTF(Long.toString(sz));
    dout.flush();
    System.out.println("Size: " + sz);
    System.out.println("Buf size: " + s.getReceiveBufferSize());

    while ((read = fin.read(b)) != -1) {
        dout.write(b, 0, read);
        dout.flush();
    }
    fin.close();

}
```

Extra flush statements removed



```
    System.out.println("..ok");
    dout.flush();

    System.out.println("Send Complete");
    dout.flush();
```

Variables to be renamed:

1. **ClientThread.java**

```
private static Socket s;

private static DataInputStream din;

private static DataOutputStream dout;

File f = new File(filename);

   FileInputStream fin = new FileInputStream(f);

   long sz = (int) f.length();

   byte b[] = new byte[1024];
```

2. **MainServer.java**

```
   ServerSocket ss = new ServerSocket(5000);

      Socket s = null;

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

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

         Thread ct;
```

3. **MainClient.java**

```
static FileOutputStream fos;

static DataInputStream din;

static DataOutputStream dout;

   Scanner sc = new Scanner(System.in);

   Socket s = new Socket(address, 5000);

   long sz = Long.parseLong(din.readUTF());

   byte b[] = new byte[1024];
```

Variables after rename:

1. **ClientThread.java**

```
private static Socket clientSocket;
private static DataInputStream dataIn;
private static DataOutputStream dataOut;
File file = new File(filename);
    FileInputStream fileIn = new FileInputStream(file);
    long size = (int) file.length();
    byte bytes[] = new byte[1024];
```

2. **MainServer.java**

```
    ServerSocket socket_Server = new ServerSocket(5000);
        Socket serverSocket = null;
            DataInputStream dataIn = new DataInputStream(serverSocket.getInputStream());
            DataOutputStream dataOut = new
DataOutputStream(serverSocket.getOutputStream());
            Thread clientThread;
```

3. **MainClient.java**

```
static FileOutputStream fileOut;
static DataInputStream dataIn;
static DataOutputStream dataOut;
    Scanner input = new Scanner(System.in);
    Socket clientSocket = new Socket(address, 5000);
    long size = Long.parseLong(dataIn.readUTF());
    byte bytes[] = new byte[1024];
```

Code:

```
/*
 * 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 FileClient;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.net.Socket;
import java.util.Scanner;

/**
 *
 * @author stygianlgdonic
 */
public class MainClient {

    static FileOutputStream fos;
    static DataInputStream din;
    static DataOutputStream dout;

    public static void main(String args[]) throws Exception {

        // Client's file request
        System.out.println("Enter File Name: ");
        Scanner sc = new Scanner(System.in);
```

```
String filename1 = sc.nextLine();


// Server's address

String address = "";

System.out.println("Enter Server Address: ");

address = sc.nextLine();


Socket s = new Socket(address, 5000);

//connection setup


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

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


dout.writeUTF(filename1);


String filename = "";


  dout.flush();


  byte check = din.readByte();

  if (check == 1) {

    // if file found Start recieving


    startRecieving(filename);


    System.out.println("Closing Connection");


    fos.close();
```

```
            din.close();

            dout.close();

            s.close();

        } else {

            // if file not found Close Connection

            System.out.println("FileNotFound");

            System.out.println("Closing Connection");

            dout.close();

            din.close();

            s.close();


        }

    }


    public static void startRecieving(String filename) throws IOException {

        filename = din.readUTF();

        System.out.println("Receving file: " + filename);

        filename = "client" + filename;

        System.out.println("Saving as file: " + filename);


        long sz = Long.parseLong(din.readUTF());

        System.out.println("File Size: " + (sz / (1024 * 1024)) + " MB");


        byte b[] = new byte[1024];

        System.out.println("Receving file..");

        // Client's save directory

        fos = new FileOutputStream(new File(

                "C:\\Users\\aliik\\Desktop\\" + filename), true);
            long bytesRead;

            // Saving file to Client's save directory
            do {
                bytesRead = din.read(b, 0, b.length);
                fos.write(b, 0, b.length);
                System.out.println(".");
            } while (!(bytesRead < 1024));

            System.out.println("Completed");
    }
}
/*
 * 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 FileServer;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.net.Socket;
```

```
/**
 *
 * @author stygianlgdonic
 */
public class ClientThread extends Thread {

   private static Socket s;
   private static String filename;
   private static DataInputStream din;
   private static DataOutputStream dout;

   public ClientThread(Socket s, String filename,
         DataInputStream din, DataOutputStream dout) {
      ClientThread.s = s;
      ClientThread.filename = filename;
      ClientThread.din = din;
      ClientThread.dout = dout;
   }

   @Override
   public void run() {
      try {

         System.out.println("Sending File: " + filename);

         byte check;
```

```
//server's directory to search from
File f = new File(filename);

if (f.exists() == true) {

   // if requested file exists
   check = 1;
   dout.writeByte(check);
   dout.flush();

} else {

   // if requested file exists
   check = 0;
   dout.writeByte(check);
   dout.flush();
   System.out.println("Error: FileNotFound");

   closeConnection();

}

dout.writeUTF(filename);
dout.flush();

// For Sending Server's File
sendFile(f);
```

```
        System.out.println("Send Complete");

        closeConnection(); // Closing Connection

    } catch (Exception e) {

        e.printStackTrace();

        System.out.println("An error occured");


    }


}


static void sendFile(File f) throws FileNotFoundException, IOException{

    FileInputStream fin = new FileInputStream(f);

    long sz = (int) f.length();

    byte b[] = new byte[1024];

    int read;

    dout.writeUTF(Long.toString(sz));

    dout.flush();

    System.out.println("Size: " + sz);

    System.out.println("Buf size: " + s.getReceiveBufferSize());


    while ((read = fin.read(b)) != -1) {

        dout.write(b, 0, read);

        dout.flush();

    }

    fin.close();


}
```

```
    public void closeConnection() throws IOException {
        System.out.println("Connection has been closed with "
                + s.getInetAddress().toString());
        s.close();
        dout.close();
        din.close();
    }
}
/*
 * 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 FileServer;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;

/**
 *
 * @author stygianlgdonic
 */
public class MainServer {

    public static void main(String args[]) throws IOException {
```

```java
// Creating socket on port 5000
ServerSocket ss = new ServerSocket(5000);

String filename;

while (true) {

  Socket s = null;

  try {
    System.out.println("Waiting for request");

    s = ss.accept();

    DataInputStream din = new DataInputStream(s.getInputStream());
    DataOutputStream dout = new DataOutputStream(s.getOutputStream());

    filename = din.readUTF();
    System.out.println("Connected With " + s.getInetAddress().toString()
        + "requesting for file:" + filename);

    // Initializing Thread
    Thread ct;
    ct = new ClientThread(s, filename, din, dout);
    ct.start(); // thread start

  } catch (Exception ex) {
```

```
                s.close(); // Closing Socket
                ex.printStackTrace();
            }
        }
    }

}
```