Assignment 6

- Problem Statement:
  Write a program using TCP Socket for wired network for following
  a) say Hello to each other.
  b) File transfer.
  c) Calculator (Trigonometry)

- Learning Objective
  To learn TCP socket & implement client & server program.

- Learning Outcome:
  - learn cacept of TCP socket prog.
  - Implement program for client & server interaction.

- S/W or H/W Req:
  - Fedora 20 with Pentium IV & above.
  - 1GB RAM, 120GB HDD, Monitor, keyboard, Mouse, Eclipse.

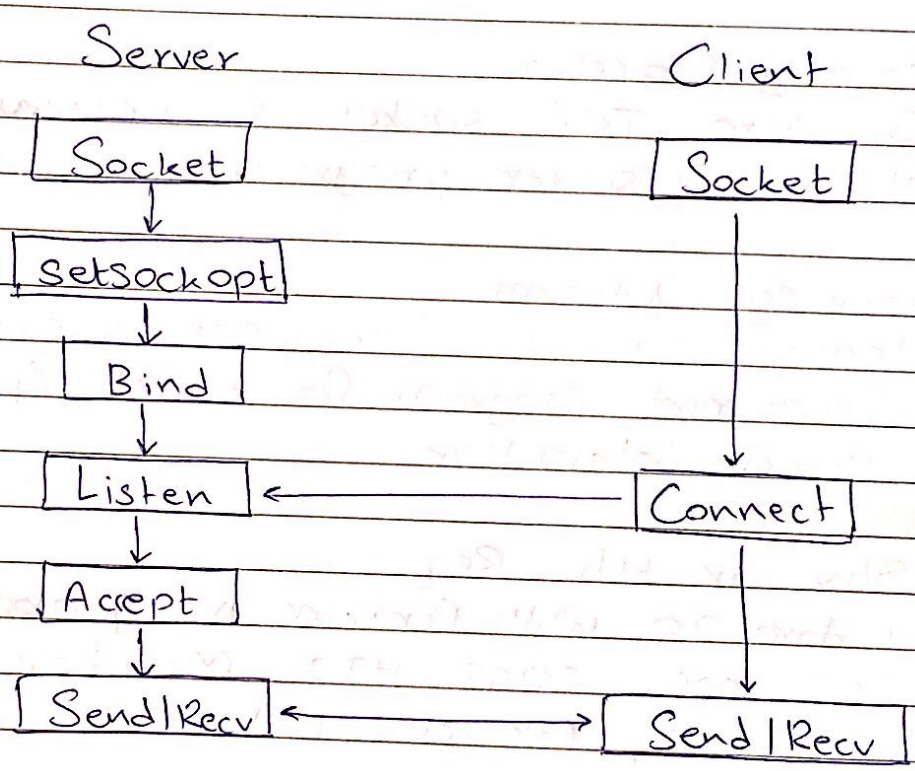
- Theory:

  - TCP socket prog. for wired network.

  If we are creating a connection between client & server using TCP then it has few functionality like, TCP is suited for application

that require high reliability & transmission time is reliability less critical. It is used by other protocols like HTTP, HTTPS, FTP, SMTP, Telnet.

The process can be broken down into following steps.



- TCP Server:

1) Use create(), create TCP socket
2) Use bind(), bind socket to server address.
3) Use listen(), put the server socket in a passive mode, where it waits for the client to approach the server to make connection.

4) Using accept()
5) Go back to step 3.

. TCP client.
1) Create TCP socket
2) Connect newly created client socket to server.

. Running Socket programs:
1) Run server.cpp file
2) Type g++ server.cpp
3) run by command ./a.out
4) Open new terminal
5) Run g++ client.cpp & ./a.out.

. File transfer:
A TCP client initiates the communication with server, which is waiting for the connection. TCP is connection oriented and UDP is connectionless which means that UDP sockets donot need to coonnectied before being used.
Another difference between TCP & UDP is that there is no garuntee that a message sent via a UDP socket will arrive at it's destination, and messages can be delivered in a different order than they were sent.

The TCP listener is created & starts listening to the specified part. Again the buffer size is set to 1024 bytes. A TCP listener ran pre-check to see if there are any connections pending before calling the Accept TCP client method.

- Conclusion:

Thus, we successfully implemented TCP socket program with client server interaction.

CODE :

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

public class Client {
    private static final Scanner scanner = new Scanner(System.in);
    private static DataOutputStream dataOutputStream;
    private static DataInputStream dataInputStream;

    public static void main(String[] args) {
        try(Socket socket = new Socket("localhost",5000)) {
            System.out.println("connected: "+socket.getInetAddress()+":"+socket.getPort());

            dataInputStream = new DataInputStream(socket.getInputStream());
            dataOutputStream = new DataOutputStream(socket.getOutputStream());

            System.out.println("\nOptions: /chat, /calculator, /transfer, /exit");
            System.out.print("> ");
            String choice = scanner.nextLine();
            dataOutputStream.writeUTF(choice);
            while(!choice.equalsIgnoreCase("/exit")){
                switch (choice){
                    case "/chat":
                        chat();
                        break;

                    case "/calculator":
                        calculator();
                        break;

                    case "/transfer":
                        fileTransfer();
                        break;

                    default:
                        System.out.println("Invalid Option: "+choice);
                }

                System.out.println("\nOptions: /chat, /calculator, /transfer, /exit");
                System.out.print("> ");
                choice = scanner.nextLine();
                dataOutputStream.writeUTF(choice);
            }
```

```
    } catch (Exception e){
        System.out.println("Error: "+e.toString());
    }
}

private static void chat() throws Exception{
    System.out.println("\n/chat");

    String reply, message;
    while(true){
        System.out.print("message: ");
        message = scanner.nextLine();
        dataOutputStream.writeUTF(message);
        if(message.equalsIgnoreCase("/exit"))
            break;

        reply = dataInputStream.readUTF();
        System.out.println("reply: "+reply);
        if(reply.equalsIgnoreCase("/exit"))
            break;
    }
}

private static void calculator() throws Exception{
    System.out.println("\n/calculator");

    String expression;
    String value;
    while (true){
        System.out.print("Expression: ");
        expression = scanner.nextLine();
        dataOutputStream.writeUTF(expression);
        if(expression.equalsIgnoreCase("/exit"))
            break;
        value = dataInputStream.readUTF();
        System.out.println("value: "+value);
    }
}

private static void fileTransfer() throws Exception{
    System.out.println("\n/transfer");

    System.out.print("File Path: ");
    String path = scanner.nextLine();
    dataOutputStream.writeUTF(path);
    if(path.equalsIgnoreCase("/exit"))
        return;

    int bytes = 0;
```

```
        File file = new File(path);
        FileInputStream fileInputStream = new FileInputStream(file);
        dataOutputStream.writeLong(file.length());
        byte[] buffer = new byte[4*1024];
        while ((bytes=fileInputStream.read(buffer))!=-1){
            dataOutputStream.write(buffer,0,bytes);
            dataOutputStream.flush();
        }
        fileInputStream.close();
        System.out.println("File Sent");
    }
}



import javax.script.ScriptEngine;
import javax.script.ScriptEngineManager;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.FileOutputStream;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Scanner;

public class Server {
    private static final Scanner scanner = new Scanner(System.in);
    private static DataOutputStream dataOutputStream;
    private static DataInputStream dataInputStream;

    public static void main(String[] args) {
        try(ServerSocket serverSocket = new ServerSocket(5000)) {
            System.out.println("listening to port:5000");
            Socket socket = serverSocket.accept();
            System.out.println("connected: "+socket.getInetAddress()+":"+socket.getPort());

            dataInputStream = new DataInputStream(socket.getInputStream());
            dataOutputStream = new DataOutputStream(socket.getOutputStream());

            String choice = dataInputStream.readUTF();
            while(!choice.equalsIgnoreCase("/exit")){
                switch (choice){
                    case "/chat":
                        chat();
                    break;

                    case "/calculator":
                        calculator();
                    break;
```

```
                case "/transfer":
                    fileTransfer();
                break;

                default:
                    System.out.println("Invalid Option: "+choice);
            }
            choice = dataInputStream.readUTF();
        }

    } catch (Exception e){
        System.out.println("Error: "+e.toString());
    }
}

private static void chat() throws Exception{
    System.out.println("\n/chat");

    String reply, message;
    while(true){
        reply = dataInputStream.readUTF();
        System.out.println("reply: "+reply);
        if(reply.equalsIgnoreCase("/exit"))
            break;

        System.out.print("message: ");
        message = scanner.nextLine();
        dataOutputStream.writeUTF(message);
        if(message.equalsIgnoreCase("/exit"))
            break;
    }
}

private static void calculator() throws Exception{
    System.out.println("\n/calculator");

    String expression;
    String value;
    ScriptEngineManager engineManager = new ScriptEngineManager();
    ScriptEngine engine = engineManager.getEngineByName("js");

    while (true){
        expression = dataInputStream.readUTF();
        System.out.println("Expression: "+expression);
        if(expression.equalsIgnoreCase("/exit"))
            break;
        try {
            value = String.valueOf(engine.eval(expression));
        }catch (Exception e){
```

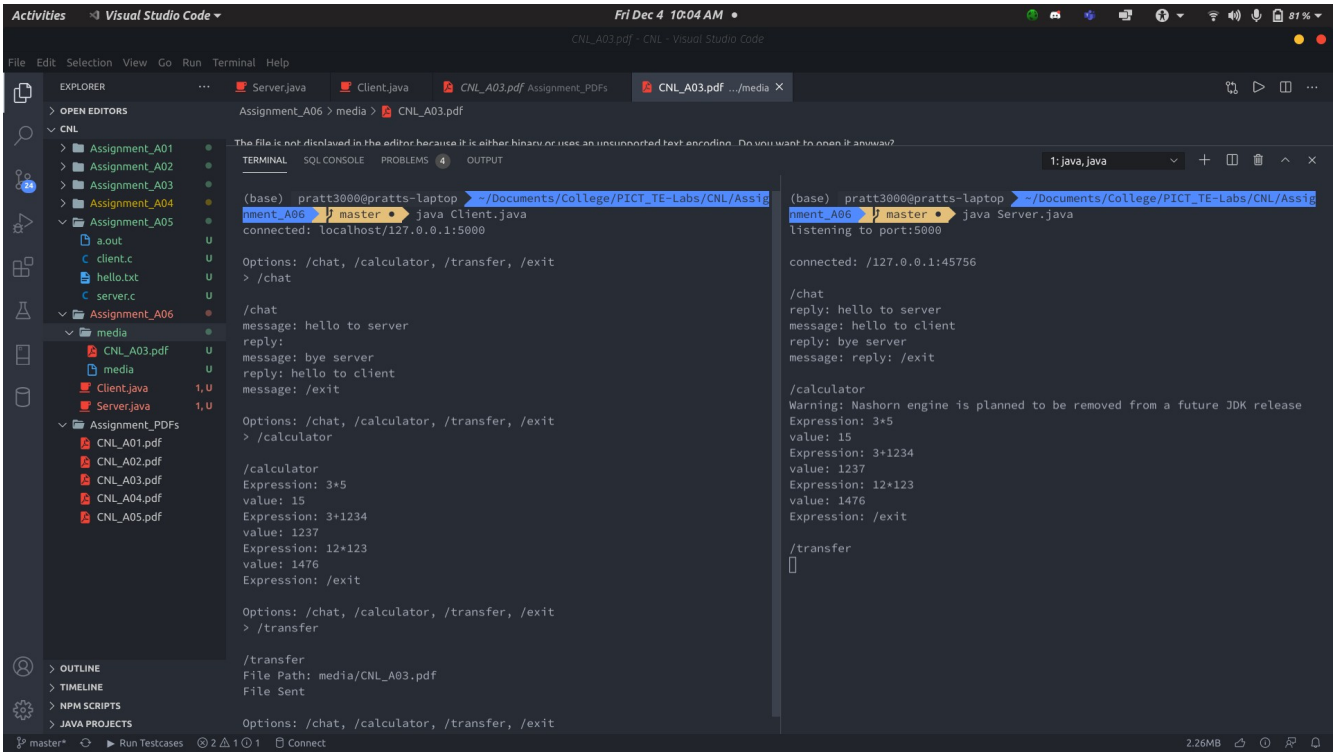
```
            value = "Invalid Expression";
          }
          dataOutputStream.writeUTF(value);
          System.out.println("value: "+value);
        }
    }

    private static void fileTransfer() throws Exception{
        System.out.println("\n/transfer");

        String path = dataInputStream.readUTF();
        if(path.equalsIgnoreCase("/exit"))
            return;

        String fileName = path.substring(path.lastIndexOf('/') + 1);
        int bytes = 0;
        FileOutputStream fileOutputStream = new FileOutputStream("media/"+fileName);

        long size = dataInputStream.readLong();     // read file size
        byte[] buffer = new byte[4*1024];
        while (size > 0 && (bytes = dataInputStream.read(buffer, 0, (int)Math.min(buffer.length, size))) !
= -1) {
            fileOutputStream.write(buffer,0,bytes);
            size -= bytes;      // read upto file size
        }
        fileOutputStream.close();
        System.out.println("Received File: "+fileName);
    }
}
```


Activities
Visual Studio Code
Fri Dec 4 10:04 AM
CNL_A03.pdf - CNL - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
Server.java
Client.java
CNL_A03.pdf Assignment_PDFs
CNL_A03.pdf .../media
OPEN EDITORS
CNL
Assignment_A01
Assignment_A02
Assignment_A03
Assignment_A04
Assignment_A05
a.out
client.c
hello.txt
server.c
Assignment_A06
media
CNL_A03.pdf
media
Client.java 1, U
Server.java 1, U
Assignment_PDFs
CNL_A01.pdf
CNL_A02.pdf
CNL_A03.pdf
CNL_A04.pdf
CNL_A05.pdf
OUTLINE
TIMELINE
NPM SCRIPTS
JAVA PROJECTS
Assignment_A06 > media > CNL_A03.pdf
The file is not displayed in the editor because it is either binary or uses an unsupported text encoding. Do you want to open it anyway?
TERMINAL
SQL CONSOLE
PROBLEMS 4
OUTPUT
1: java, java
(base) pratt3000@pratts-laptop ~/Documents/College/PICT_TE-Labs/CNL/Assignment_A06 master • java Client.java
connected: localhost/127.0.0.1:5000
Options: /chat, /calculator, /transfer, /exit
> /chat
/chat
message: hello to server
reply:
message: bye server
reply: hello to client
message: /exit
Options: /chat, /calculator, /transfer, /exit
> /calculator
/calculator
Expression: 3*5
value: 15
Expression: 3+1234
value: 1237
Expression: 12*123
value: 1476
Expression: /exit
Options: /chat, /calculator, /transfer, /exit
> /transfer
/transfer
File Path: media/CNL_A03.pdf
File Sent
Options: /chat, /calculator, /transfer, /exit
(base) pratt3000@pratts-laptop ~/Documents/College/PICT_TE-Labs/CNL/Assignment_A06 master • java Server.java
listening to port:5000
connected: /127.0.0.1:45756
/chat
reply: hello to server
message: hello to client
reply: bye server
message: reply: /exit
/calculator
Warning: Nashorn engine is planned to be removed from a future JDK release
Expression: 3*5
value: 15
Expression: 3+1234
value: 1237
Expression: 12*123
value: 1476
Expression: /exit
/transfer
master*
Run Testcases
Connect
2.26MB