

Computer Networks Lab 3

SAHIL BONDRE: U18CO021

Server:

```
import java.io.BufferedReader;
import java.io.DataInputStream;
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;

public class Server {
    private Socket socket;
    private ServerSocket server;
    private DataInputStream in;
    private boolean serverStarted = false;

    public Server(int port) {
        // setup server socket
        try {
            serverStarted = false;
            server = new ServerSocket(port);
            serverStarted = true;
        } catch (Exception e) {
            System.out.println(e);
        }
    }

    public void listen() throws IOException {
        if (!serverStarted) {
            // server socket not setup
            throw new IOException("Server not started on any port");
        }
        System.out.println("Server started");
        System.out.println("Waiting for a client ...");
        socket = server.accept();
        System.out.println("Client accepted");
        // listen for the client
        in = new DataInputStream(new
BufferedReader(new DataInputStream(socket.getInputStream())));
        String line = "";
        // read message from client until "exit" is sent
        while (!line.equals("exit")) {
```

```

        try {
            line = in.readUTF();
            System.out.println(line);
        } catch (Exception i) {
            System.out.println(i);
        }
    }
    System.out.println("Closing connection");
    // close connection
    socket.close();
    in.close();
}

public static void main(String args[]) {
    Server server = new Server(5000);
    try {
        server.listen();
    } catch (Exception e) {
        System.out.println(e);
    }
}
}
}

```

Client:

```

import java.net.ConnectException;
import java.net.Socket;
import java.util.Scanner;
import java.io.DataOutputStream;
import java.io.IOException;

class Client {
    private Socket socket;
    private Scanner input;
    private DataOutputStream output;
    private boolean connectionEstablished = false;

    public Client(String address, int port) {
        try {
            // Initialise socket and IO Streams
            connectionEstablished = false;

```

```

        socket = new Socket(address, port);
        System.out.println("Connected");

        input = new Scanner(System.in);
        output = new DataOutputStream(socket.getOutputStream());
        connectionEstablished = true;
    } catch (Exception e) {
        System.out.println(e);
    }
}

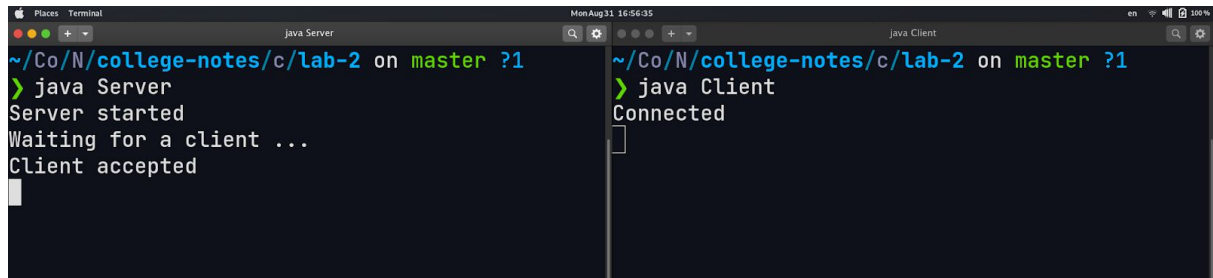
public void connect() throws ConnectException, IOException {
    if (!connectionEstablished) {
        // socket not setup
        throw new ConnectException("Connection Not Established");
    }
    String line = "";
    while (!line.equals("exit")) {
        try {
            line = input.nextLine();
            output.writeUTF(line);
        } catch (Exception e) {
            System.out.println(e);
            break;
        }
    }

    input.close();
    output.close();
}

public static void main(String[] args) {
    Client c = new Client("127.0.0.1", 5000);
    try {
        c.connect();
    } catch (Exception e) {
        System.out.println(e);
    }
}
}

```

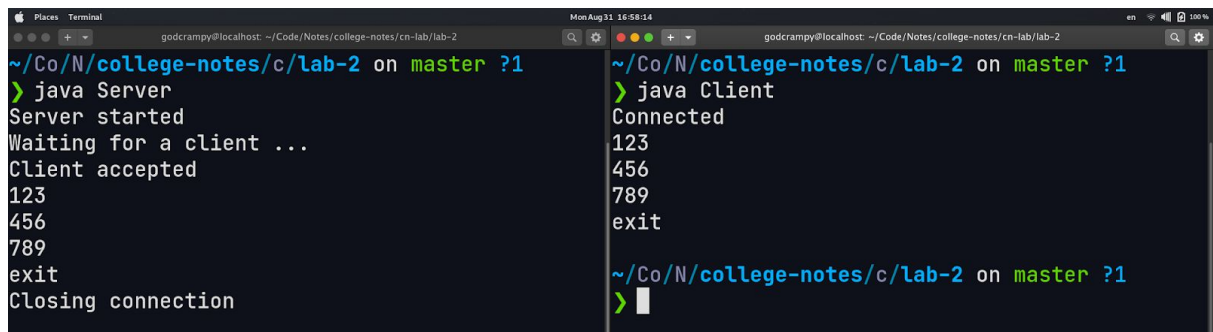
Client and Server Started



The screenshot shows two terminal windows side-by-side. The left window, titled 'java Server', shows the command 'java Server' being executed, followed by the output 'Server started', 'Waiting for a client ...', and 'Client accepted'. The right window, titled 'java Client', shows the command 'java Client' being executed, followed by the output 'Connected'.

```
~/Co/N/college-notes/c/lab-2 on master ?1  
> java Server  
Server started  
Waiting for a client ...  
Client accepted  
~  
~/Co/N/college-notes/c/lab-2 on master ?1  
> java Client  
Connected  
~
```

Sending text from client to server. Server closes on sending exit command



The screenshot shows two terminal windows side-by-side. The left window, titled 'java Server', shows the command 'java Server' being executed, followed by the output 'Server started', 'Waiting for a client ...', and 'Client accepted'. It then shows the client sending the text '123', '456', and '789', followed by 'exit' and 'Closing connection'. The right window, titled 'java Client', shows the command 'java Client' being executed, followed by the output 'Connected'. It then shows the client sending the text '123', '456', and '789', followed by 'exit'.

```
~/Co/N/college-notes/c/lab-2 on master ?1  
> java Server  
Server started  
Waiting for a client ...  
Client accepted  
123  
456  
789  
exit  
Closing connection  
~  
~/Co/N/college-notes/c/lab-2 on master ?1  
> java Client  
Connected  
123  
456  
789  
exit  
~/Co/N/college-notes/c/lab-2 on master ?1  
> ~
```