3

Lab

**BÁO CÁO BÀI THỰC HÀNH SỐ 3**

**Lập Trình Mạng Với Java dùng Socket**

**Môn học: NT109.O21.MMCL**

|  |  |
| --- | --- |
| **Sinh viên thực hiện** | Trương Đình Kha (21522180) |
| **Khó khăn** | * Ban đầu không thể chạy được code vì em để port sai |
| **Số câu đã hoàn thành** | 2/2 |

**TRẢ LỜI CÁC CÂU HỎI**

**Làm thêm:** *Viết hàm ghi ra một file log theo yêu cầu của thầy*



1. Hình ảnh hàm ghi log.

**Bài 1:** *Thực hiện kết nối socket TCP và giao tiếp giữa Server và Client.*

* **Ở phía Server:**

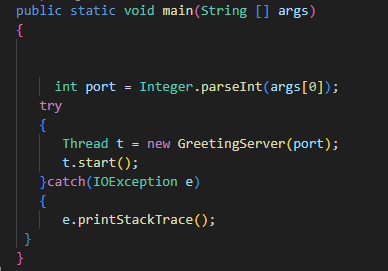


1. Đầu tiên, em tạo một biến serverSocket để nhận giá trị port là số cổng mà server sẽ lắng nghe và em đã cấu hình cho socket một thời gian chờ để client có thể kết nối vào khi quá thời gian này thì server sẽ tự ngắt kết nối.



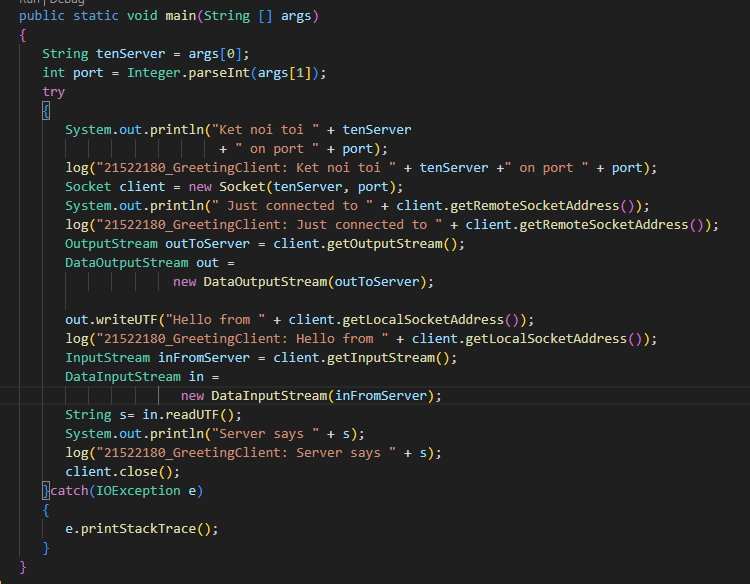
1. Ở hàm này chạy một server socket để đợi kết nối từ phía client. Khi có kết nối, server sẽ nhận và in ra thông điệp từ client, sau đó gửi một thông điệp phản hồi và đóng kết nối. Phương thức này xử lý các ngoại lệ như timeout và lỗi I/O.

* **Cụ thể:**
  + serverSocket.accept(): Phương thức này làm chương trình chờ đợi và chấp nhận một kết nối từ client. Khi có kết nối được thiết lập, nó trả về một đối tượng Socket mới.
  + DataInputStream in = new DataInputStream(server.getInputStream()): Tạo một DataInputStream để đọc dữ liệu từ client thông qua socket.
  + System.out.println(in.readUTF()): Đọc một chuỗi Unicode từ DataInputStream và in ra nó trên terminal.
  + DataOutputStream out = neư DataOutputStream(server.getOutputStream()): Tạo một DataOutputStream để gửi dữ liệu đến client thông qua socket.
  + out.writeUTF("Thank you for connecting to " + server.getLocalSocketAddress() + "\nGoodbye!"): Gửi một chuỗi Unicode đến client để thông báo rằng kết nối đã được thiết lập thành công và đóng kết nối sau đó.
  + server.close(): Đóng socket và kết thúc phiên kết nối với client.
  + catch(SocketTimeoutException s) và catch(IOException e): Xử lý các ngoại lệ có thể xảy ra trong quá trình chạy.



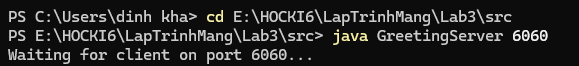
1. Ở hàm main, em lấy số port cần chạy từ terminal sau đó, em tạo 1 luồng thread mới cho lớp GreetingServer và truyền số đó vào server socket để mở socket.

* **Ở phía Client:**



1. Ở phía client, nhận hai đối số dòng lệnh (command-line arguments) là tên của server và số port để kết nối tới. Sau đó, phương thức tạo một kết nối socket tới server được chỉ định trên cổng đã có.

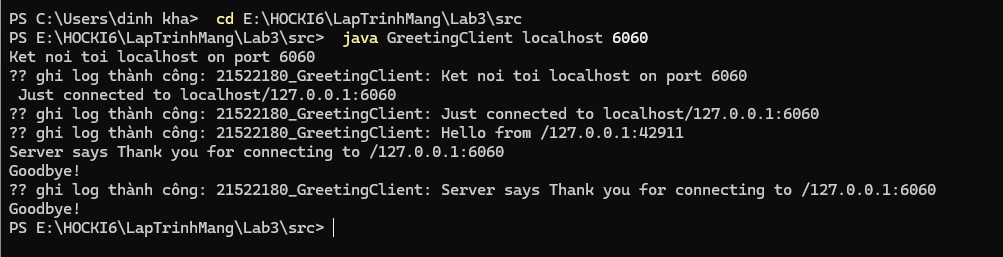
* **Cụ thể:** 
  + String tenServer = args[0]: lấy tên của server từ đối số dòng lệnh, và int port = Integer.parseInt(args[1]) lấy số cổng từ đối số dòng lệnh.
  + Socket client = new Socket(tenServer, port): tạo một socket để kết nối tới server với tên và cổng đã chỉ định.
  + Sau khi kết nối thành công, client gửi một thông điệp "Hello" tới server thông qua DataOutputStream và đọc phản hồi từ server thông qua DataInputStream. Cả hai thông điệp đều được in ra terminal và ghi vào file log.
  + Khi giao tiếp hoàn thành, client đóng kết nối với server thông qua client.close().
* **Khi chạy chương trình:**



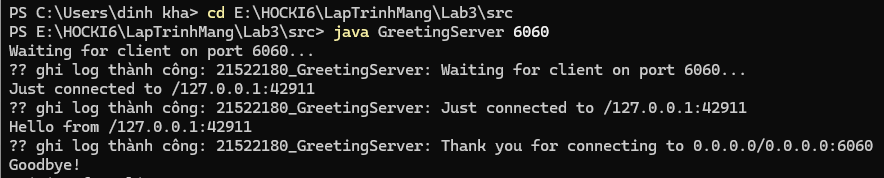
1. Đầu tiên mở 1 socket cho server và đợi ở port số 6060.



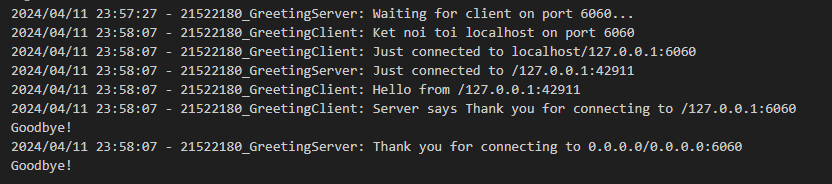
1. Tiếp theo em tạo 1 socket cho Client với name là localhost và port 6060.



1. Sau khi kết nối thành công với server thì cả hai bắt đầu giao tiếp với nhau đồng thời log sẽ được ghi lại, sau đó kết nối sẽ bị ngắt.



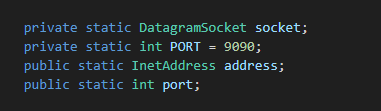
1. Bên phía Server sau khi kết nối thành công.



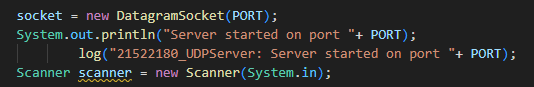
1. File log sau khi thực hiện xong.

**Bài 2:** *Thực hiện kết nối socket UDP và có thể nhắn tin giao tiếp giữa Server và Client.*

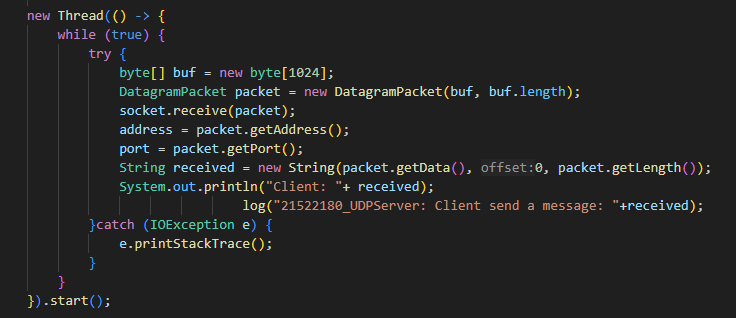
* **Ở phía Server:**



1. Đầu tiên, khởi tạo các giá trị như Socket, port server dùng, port client sẽ kết nối đến, biến để lưu địa chỉ mà client kết nối đến .

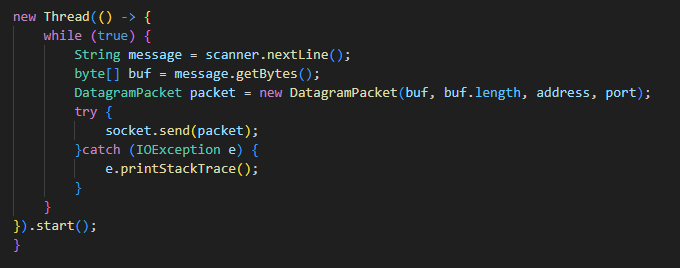


1. Tiếp theo trong hàm main, em bắt đầu mở socket UDP với port đã khai báo trước đó và thông báo tới người dùng, đồng thời ghi lại log.



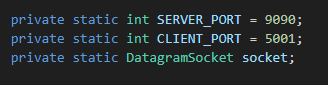
1. Đây là luồng để nhận tin nhắn từ Client gửi tới và đưa nó vào log.

* **Cụ thể:**
* Trong khi chương trình đang chạy, luồng mới sẽ lặp vô hạn để lắng nghe các tin nhắn gửi đến.
* Hàm *socket.receive(packet):* được sử dụng để nhận dữ liệu từ gói tin UDP được gửi đến máy chủ. Dữ liệu được đọc vào mảng byte buf.
* Sau đó, dữ liệu được chuyển đổi thành một chuỗi sử dụng *String received = new String(packet.getData(), 0, packet.getLength()).*
* Dòng *System.out.println("Client: "+ received)* sẽ in ra dữ liệu nhận được từ máy khách lên cửa sổ console.
* Dòng *log("21522180\_UDPServer: Client send a message: "+received)* sẽ ghi lại log.

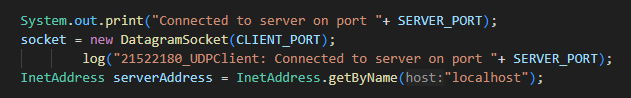


1. Luồng Thread để gửi tin nhắn qua cho Client.

* **Cụ thể:**
  + Tạo một Thread mới để gửi tin nhắn đến các client. Thread này sẽ luôn chạy trong vòng lặp vô hạn, lấy tin nhắn từ người dùng thông qua đối tượng Scanner và gửi nó đến client thông qua DatagramPacket.
  + Message nhập được chuyển đổi thành mảng byte bằng cách sử dụng message.getBytes().
  + Một DatagramPacket mới sẽ được tạo để chứa tin nhắn với địa chỉ và cổng của máy chủ đã được xác định trước (address và port).
  + Tin nhắn được gửi từ client tới máy chủ thông qua socket bằng cách sử dụng socket.send(packet).
* **Ở phía Client:**

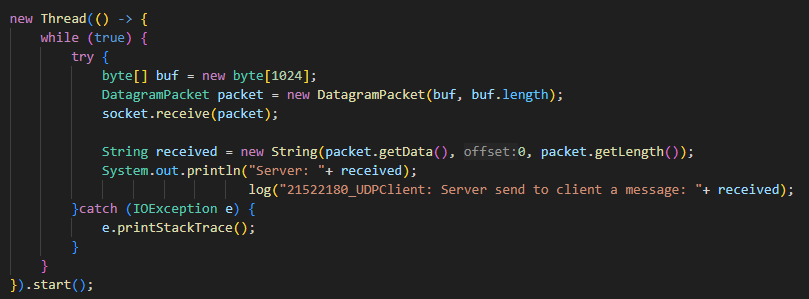


1. Đầu tiên, khởi tạo các giá trị như địa chỉ port của Server và Client, biến DatagramSocket để gọi hàm gửi nhận gói tin UDP.



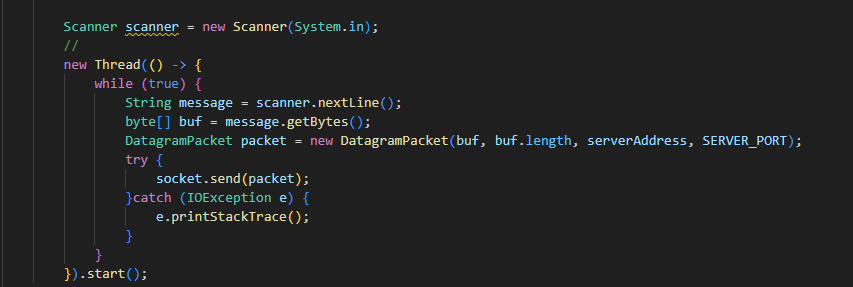
1. Tiếp theo em thiết lập kết nối với server.

* **Cụ thể:**
  + In ra màn hình đã kết nối thành công bằng Server Port.
  + Sau đó tạo 1 Client socket để có thể truyền tin qua lại với server bằng UDP
  + Tiếp theo em cho tìm địa chỉ IP của Server dựa vào tên của máy chủ (Ở đây là “localhost”).
  + Đồng thời ghi lại log.



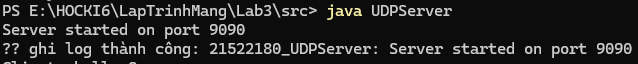
1. Tạo một luồng thread để nhận tin nhắn từ Server

* **Cụ thể:**
  + Đầu tiên, tạo 1 vòng lập vô hạn để nghe và nhận tin nhắn từ Server.
  + Hàm socket.receive(packet); được sử dụng để nhận dữ liệu từ gói tin UDP được gửi đến client. Dữ liệu được đọc vào mảng byte buf.
  + Sau đó, dữ liệu được chuyển đổi thành một chuỗi sử dụng String received = new String(packet.getData(), 0, packet.getLength());.
  + Dòng System.out.println("Server: "+ received); sẽ in ra dữ liệu nhận được từ máy chủ lên cửa sổ console.
  + Cuối cùng, dữ liệu sẽ được ghi lại vào file log.



1. Tạo một luồng Thread để gửi dữ liệu sang cho Server.

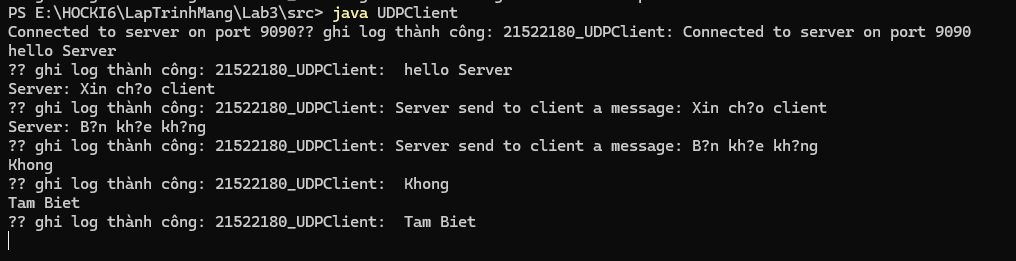
* **Cụ thể:**
  + Đầu tiên, tạo 1 vòng lập vô hạn để đọc dữ liệu từ người dùng nhập vào và gửi nó cho server.
  + Tin nhắn được nhập được chuyển đổi thành mảng byte bằng cách sử dụng message.getBytes().
  + Sau đó, một DatagramPacket mới được tạo để chứa tin nhắn với địa chỉ và cổng của máy chủ đã được xác định trước với serverAddress và serverPort.
  + Tin nhắn được gửi từ client tới máy chủ thông qua UDP socket bằng cách sử dụng socket.send(packet).
* **Khi chạy chương trình:**

**

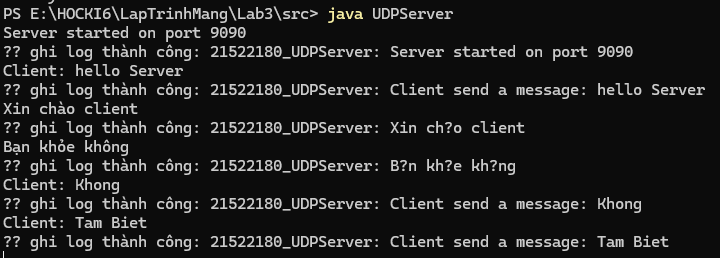
1. Đầu tiên, khởi tạo UDP Socket trên port đã set sẵn.



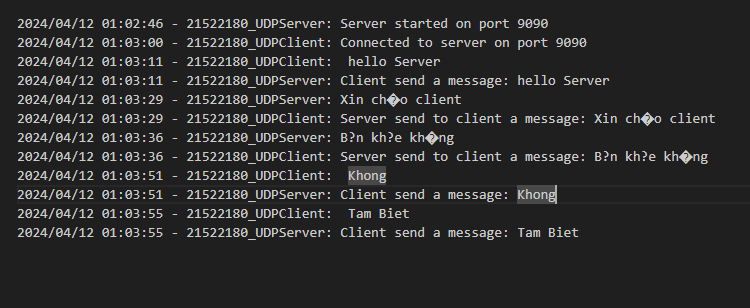
1. Sau đó em khởi tạo UDP socket ở phía Client và đã kết nối thành công với Server.



1. Sau đó, có thể nhắn tin giao tiếp với server và tất cả các message đều sẽ được lưu vào file log.



1. Hình ảnh bên phía Server.



1. Đồng thời trong lúc đó File log sẽ ghi lại toàn bộ các message giao tiếp giữa server và client.