TCP and UDP Sockets

De Renzis Nicolò

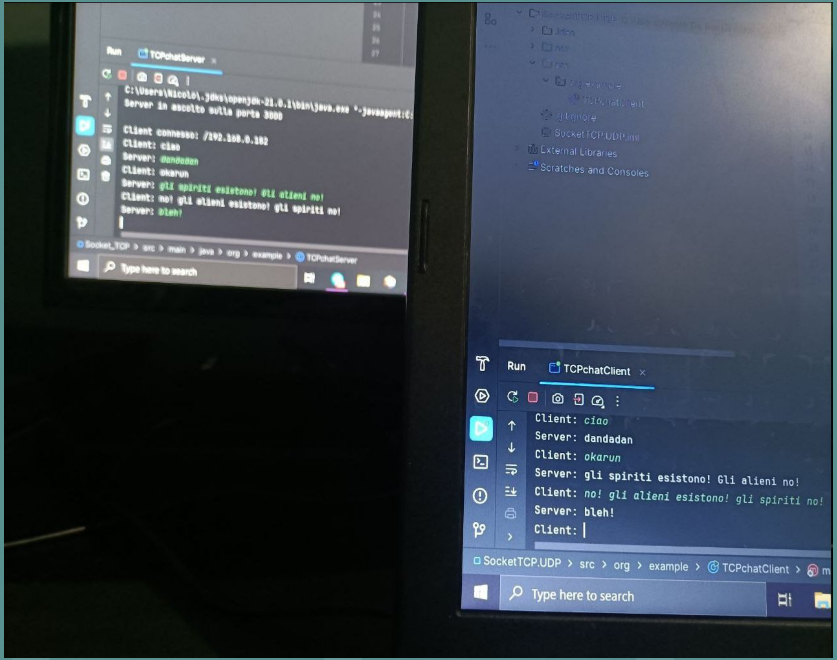
TCP server

The TCP server is configured to listen for connections on the specified port, in this case 3000. After the connection is established, bidirectional communication is maintained between the server and the client. This approach uses a connection-oriented model, where every message is sent in a continuous and ordered stream, ensuring that data arrives in the correct sequence. After accepting the connection, the server enters a loop in which it reads the received messages and allows the server user to reply. The connection continues until it is terminated, enabling continuous communication without message loss.

TCP client

The TCP client connects to a server that is assumed to already be listening on the specified port and IP address. Using the TCP protocol, the client sends messages to the server and receives responses continuously and sequentially, ensuring the order of data packets. The use of a continuous loop allows the client to remain connected and exchange messages with the server in real time. Since the TCP connection is persistent, there is no need to establish a new connection for each message, allowing the creation of a stable and reliable chat session.

TCP Client-Server Communication

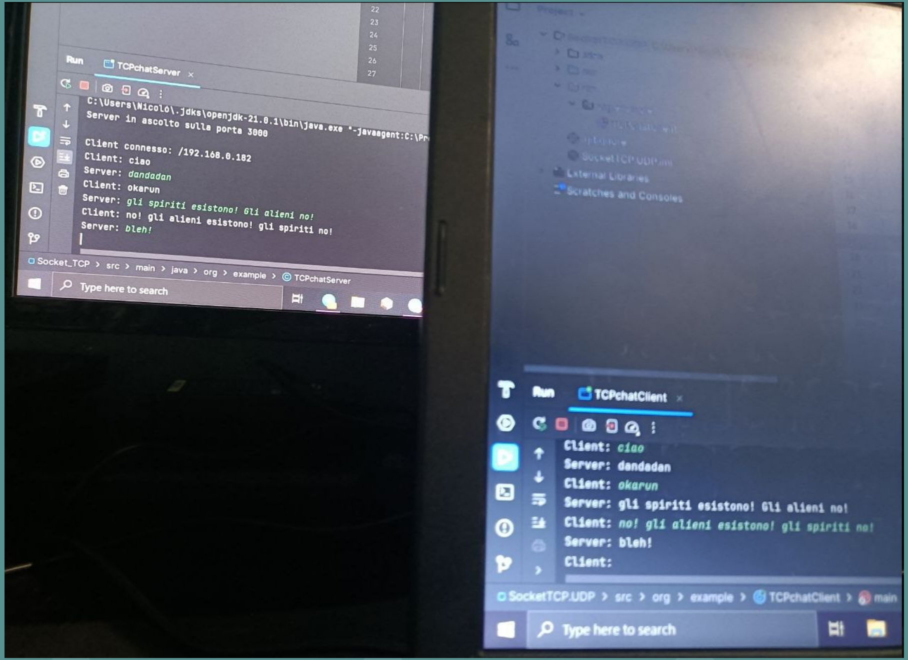


```
C:\Users\Nicolo\.jdks\openjdk-21.0.1\bin\java.exe "-javaagent:C:
Server in ascolto sulla porta 3000

Client connesso: /192.168.0.182
Client: ciao
Server: dandadan
Client: okarun
Server: gli spiriti esistono! Gli alieni no!
Client: no! gli alieni esistono! gli spiriti no!
Server: bleh!
```

```
Client: ciao
Server: dandadan
Client: okarun
Server: gli spiriti esistono! Gli alieni no!
Client: no! gli alieni esistono! gli spiriti no!
Server: bleh!
Client:
```

Run
TCPchatServer
Socket_TCP > src > main > java > org > example > TCPchatServer
Type here to search
External Libraries
Scratches and Consoles
Run
TCPchatClient
SocketTCP.UDP > src > org > example > TCPchatClient > m
Type here to search



```
C:\Users\Nicolo\.jdks\openjdk-21.0.1\bin\java.exe "-javaagent:C:\Pr
Server in ascolto sulla porta 3000

Client connesso: /192.168.0.182
Client: ciao
Server: dandadan
Client: okarun
Server: gli spiriti esistono! Gli alieni no!
Client: no! gli alieni esistono! gli spiriti no!
Server: bleh!
```

```
Client: ciao
Server: dandadan
Client: okarun
Server: gli spiriti esistono! Gli alieni no!
Client: no! gli alieni esistono! gli spiriti no!
Server: bleh!
Client:
```

Run
TCPchatServer
Socket_TCP > src > main > java > org > example > TCPchatServer
Type here to search
External Libraries
Scratches and Consoles
Run
TCPchatClient
SocketTCP.UDP > src > org > example > TCPchatClient > main
Type here to search

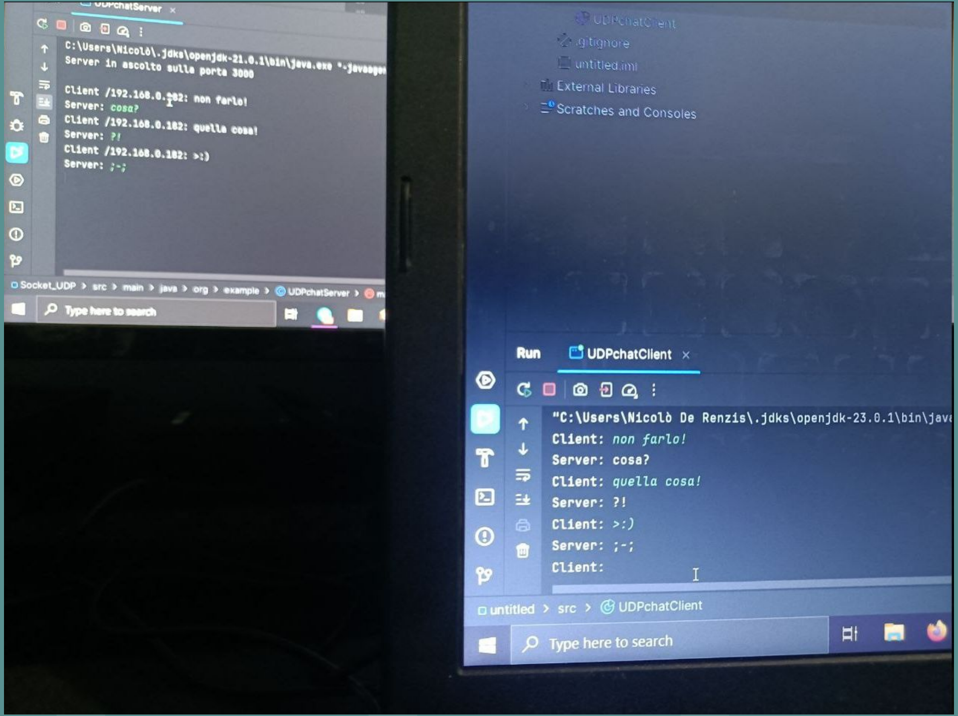
UDP server

The UDP server is configured to listen for packets on the specified port without establishing a persistent connection with the client, thanks to the connectionless nature of the UDP protocol. Data packets are received, stored, and read into a buffer, then decoded as text messages. Since UDP is connectionless, it does not guarantee message order or integrity, but it allows faster and lighter communication. The UDP server responds to the client through a packet sent to the IP address and port from which the original packet was received. This approach is ideal for applications where speed is more important than delivery guarantees, but less suitable for chat applications where message order is important.

UDP client

The UDP client, unlike the TCP client, sends messages to the server without establishing a persistent connection, taking advantage of UDP’s connectionless communication. Each message is treated as an independent packet, without any guarantee of reception or order. The simplicity of UDP allows fast packet transmission, making it ideal for lightweight and occasional communications where TCP reliability is not needed. However, in a chat system, this could result in lost or reordered messages, since UDP does not guarantee that packets arrive in the correct sequence—or arrive at all.

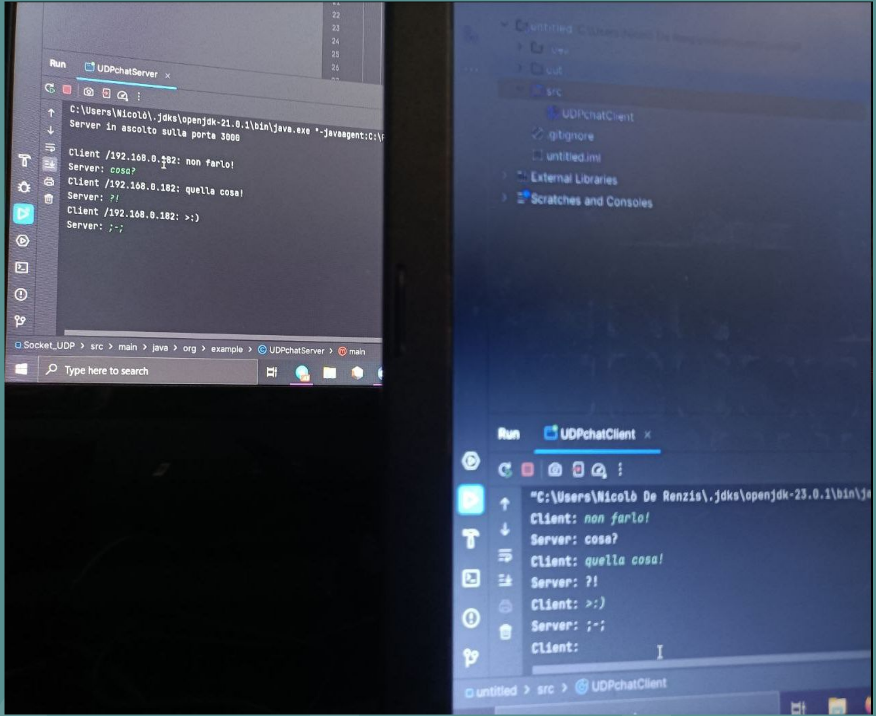
UDP Client-Server Communication



```
C:\Users\Nicolò\.jdks\openjdk-21.0.1\bin\java.exe "-javaagen
Server in ascolto sulla porta 3000

Client /192.168.0.182: non farlo!
Server: cosa?
Client /192.168.0.182: quella cosa!
Server: ?!
Client /192.168.0.182: >:)
Server: ;-;
```

```
"C:\Users\Nicolò De Renzis\.jdks\openjdk-23.0.1\bin\java
Client: non farlo!
Server: cosa?
Client: quella cosa!
Server: ?!
Client: >:)
Server: ;-;
Client:
```



```
C:\Users\Nicolò\.jdks\openjdk-21.0.1\bin\java.exe "-javaagent:C:\
Server in ascolto sulla porta 3000

Client /192.168.0.182: non farlo!
Server: cosa?
Client /192.168.0.182: quella cosa!
Server: ?!
Client /192.168.0.182: >:)
Server: ;-;
```

```
"C:\Users\Nicolò De Renzis\.jdks\openjdk-23.0.1\bin\je
Client: non farlo!
Server: cosa?
Client: quella cosa!
Server: ?!
Client: >:)
Server: ;-;
Client:
```

Explanation of the Steps to Establish a Connection

I ensured that both the device acting as the server (my desktop computer) and the client (laptop) were connected to the same network. I then entered the server’s IP address to configure the client, allowing the client to send data packets directly to the listening server. In UDP, the client sends packets without establishing a connection, while with TCP a connection is established using the accept

Differenze nei comportamenti tra TCP e UDP

TCP establishes and maintains a continuous connection between client and server for the entire data exchange. In the case of the TCP server (*TCPchatServer*) and the TCP client (*TCPchatClient*), the server accepts a connection and keeps it active to allow a continuous and orderly chat.

With UDP, data packets are sent individually without establishing a persistent connection. In the UDP server (*UDPchatServer*) and the UDP client (*UDPchatClient*), each packet is independent, and there is no stable session between client and server.

TCP guarantees that data is received in the correct order and without losses. If a packet is lost, TCP automatically retransmits it.

UDP does not guarantee either the order or the delivery of packets. In UDP chat, if a packet were lost, it would not be retransmitted, and packets might arrive out of order.

TCP is slower compared to UDP because it includes control mechanisms to ensure reliable delivery and correct ordering of data. This reliability makes TCP preferable for applications where data must be precise and orderly.

UDP is faster and lighter since it avoids reliability and connection checks. This makes it suitable for situations where speed is more important than data completeness, such as applications that can tolerate occasional data loss.