Advanced Programming – Assignment

Part 2 - Dublin Events

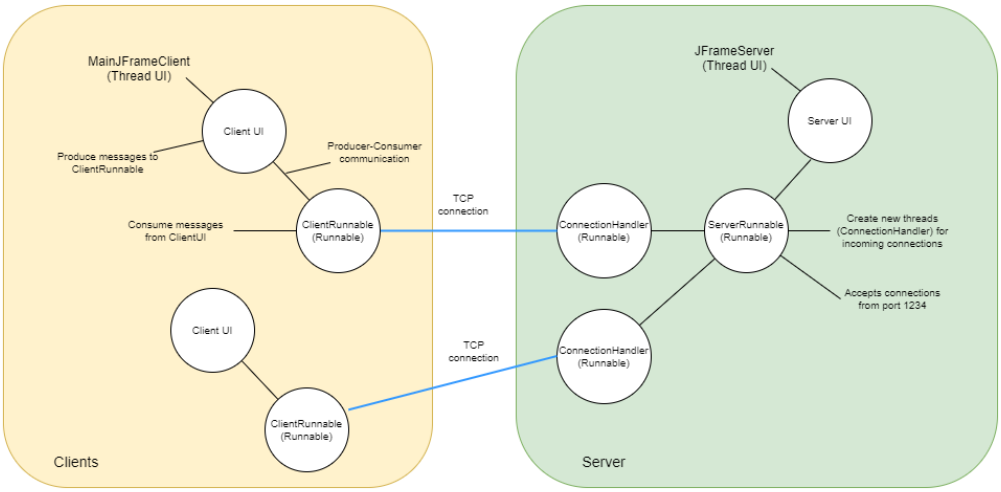
Student: Maximiliano Herrera
Student number: 20103212

Contents

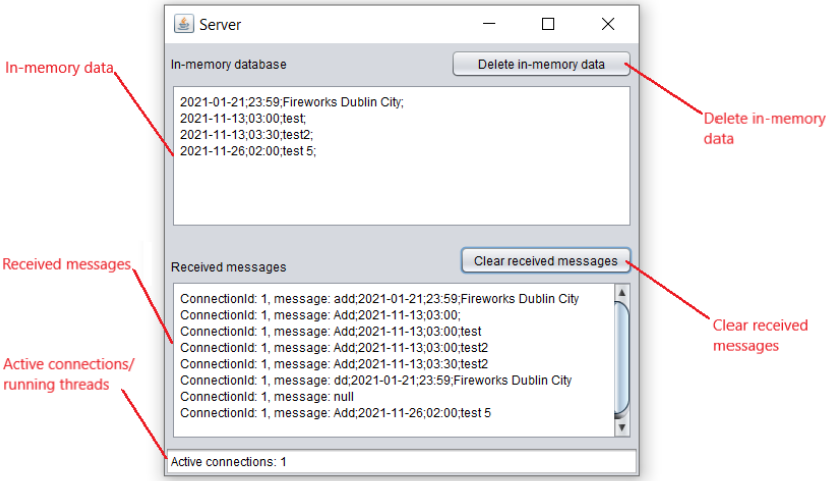
Architecture 1
Server 2
Client 2
Multiple connections 2
Adding an event 3
Removing events 3
Before removing event 3
After removing event 3
Multiple messages within a connection 3
Links 4
Client 4
Server 4

Architecture

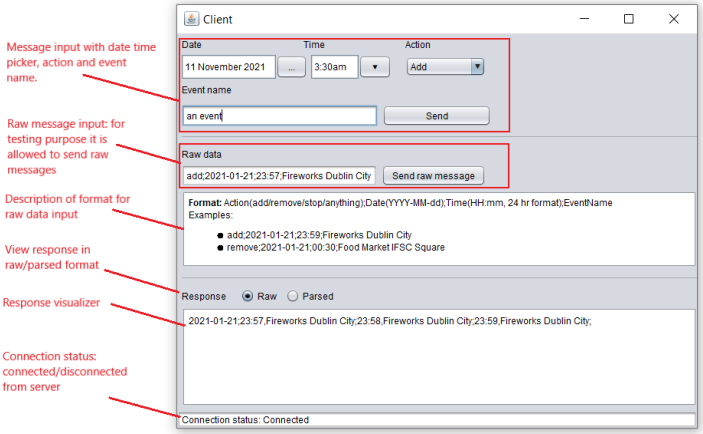
Client-Server architecture which supports multiple clients, each of them handling its own TCP connection to the only one server. Below it is shown the architecture for an example of two clients and one server from a Runnable point of view, which means each circle is a thread.



Server



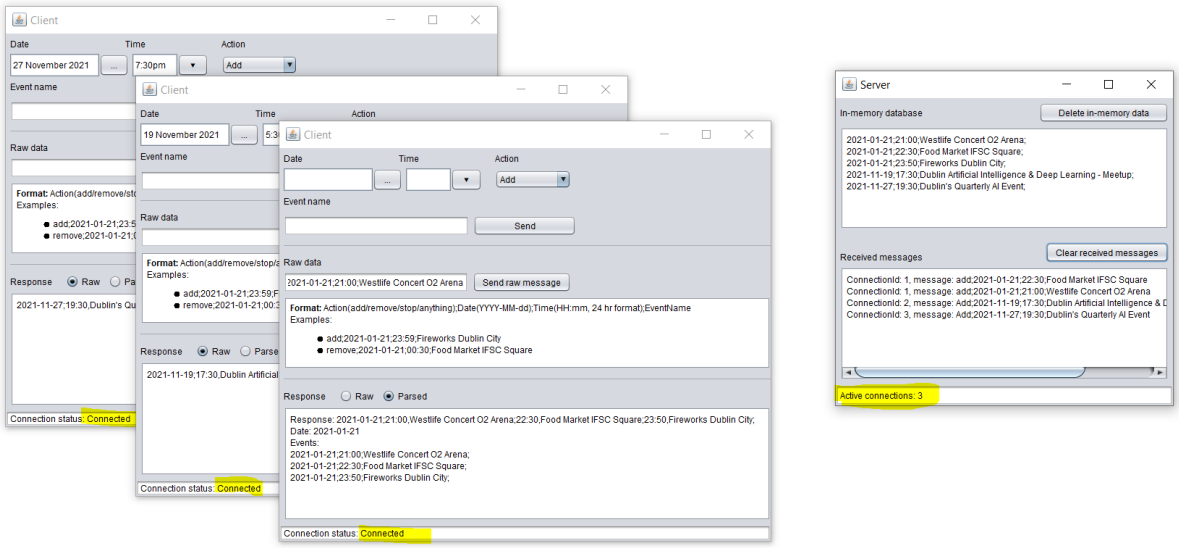
Client



Multiple connections

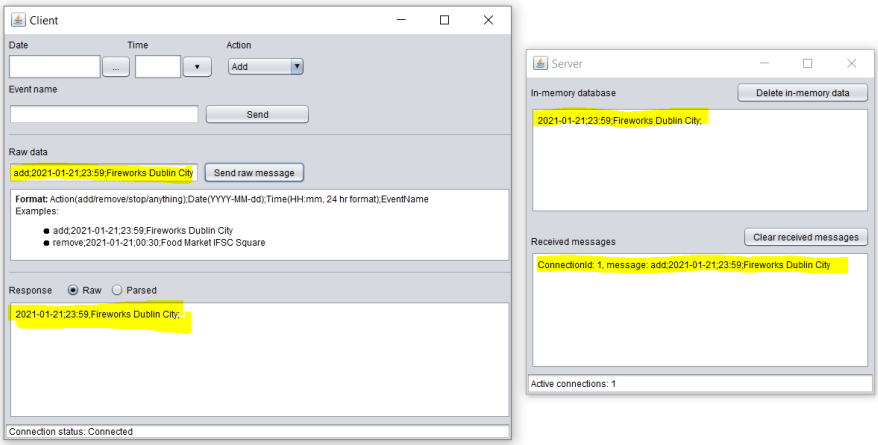
The server displays how many clients are currently connected through the “Active connections” status bar.

The Client shows if it is connected or disconnected to a server on the “Connection status” bar.



Adding an event

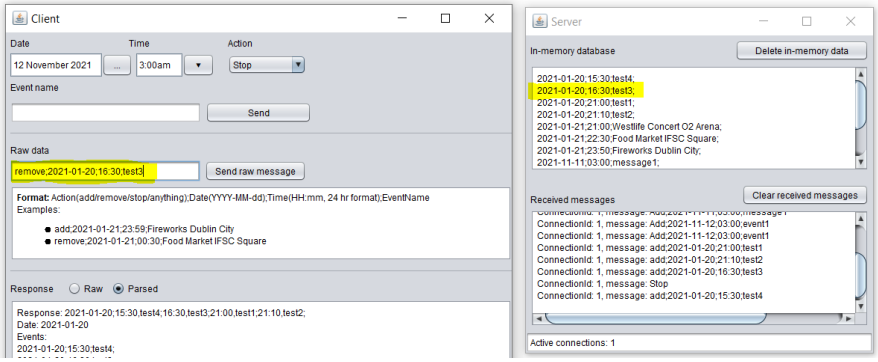
Example of adding an event and the response shown on client side as well as the in-memory database content on server side.



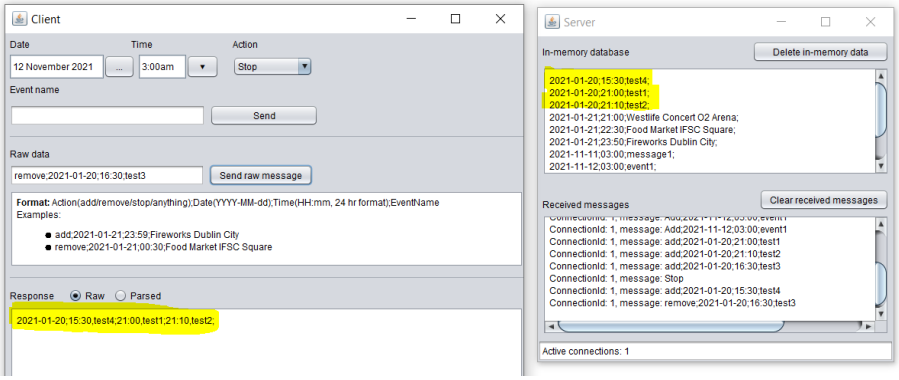
Removing events

This is an example of removing an event and showing how the UI looks like before and after in both client and server.

Before removing event

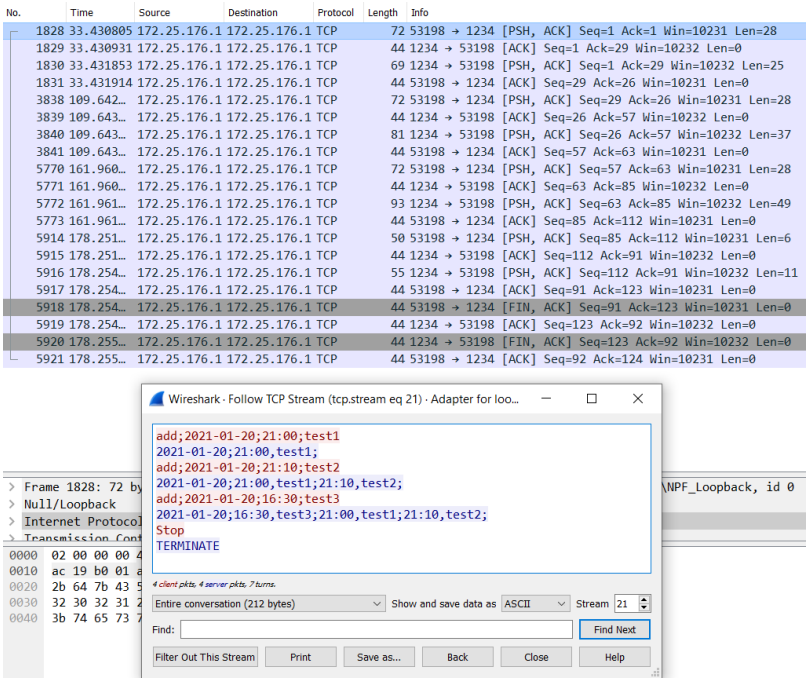


After removing event



Multiple messages within a connection

We can see from Wireshark that Client can send multiple messages within a TCP connection until the Stop-Terminate sequence is performed and the TCP connection is closed. As evidence of this please visualize the package No. 5918 and TCP stream window which are shown below:



Links

[Github repository]

Client

[MainJFrameClient.java]: Starting point, Client UI.

[ClientRunnable.java]: Receive messages from Client UI and send those messages to server through a TCP connection. The TCP connection is open until receiving TERMINATE message from server.

[ResponseCommandBuilderTest.java]: Test cases for command response builder component.

Server

[JFrameServer.java]: Starting point, Server UI.

[ServerRunnable.java]: Listen on port 1234 for incoming connections and create a tuple socket-thread (ConnectionHandler) for each of them.

[ConnectionHandler.java]: Handle the messages from client and execute commands until receiving Stop message.

[InMemoryDatabase.java]: Store in-memory data, defines add and remove methods both have linear time complexity O(N).