CS 4390.502 Computer Networks

Akaanksh Raj Kambalimath – AXK180122

Nicolas Chevrie – NJC180030

Math Server Project Design Document

Project Approach

Our project is a client-server system which uses TCP. The server constantly monitors incoming commands on its 6789 port, evaluating those commands in FIFO order. We initially designed a multi-threaded system with an evaluator thread, but decided instead to use a single-threaded NIO evaluator since it guaranteed FIFO.

Client Application Pseudocode

1. Prompts the user for their name
2. The Main class creates a TCPClient using the user name and the hardcoded IP and port number for the server (localhost, port # 6789)
    a. The TCPClient attempts to establish a connection with the server through a "hello" command.
    b. If the server acknowledges the request and the client ensures the acknowledgement is correct, then the connection is established.
    c. Register the shutdown hook, allowing the user to shut the client down at any point by inputting Ctrl+C, which will send the server an "exit" command and shut down the client

3. With the connection established, prompt the user to enter a command
4. Check format of user input and make sure it's not over the byte limit
    a. If help, h, usage or u is entered, print help info
    b. If exit, e, quit or q is entered, run the Shutdown hook, which sends the server a "exit" command closes itself
    c. If command is unrecognized, assume it is an equation and send to server
        i. If server is up and running, receive response from server and display response, then repeat from step 3
        ii. If server's socket is not writable, assume server has shut down and close client

Server Application Pseudocode

1. Opens the TCPServer at the hardcoded port (6789)
2. The TCPServer starts up its logging function
3. The TCPServer sets up non-blocking IO, with the accept event on the hardcoded port
4. The TCPServer initializes a ClientStore that helps keep track of connected clients, their name, their connection duration and the command they have sent
5. Register the shutdown hook, allowing the user to shut the server down at any point by inputting Ctrl+C, which will run through all the clients in the ClientStore, and for each client it will, close the socket and log a client exit
6. The TCPServer enters an infinite loop
    a. Look to see if it needs to handle a event from the non-blocking IO

i. If a new client has arrived, accept a non-blocking connection and register the read event

ii. If the server can read something from the client

    1. Read it into a buffer of length 2048, and add the command to the ClientStore
    2. If the command has arrived fully (new-line terminated), then process it and log command info
        a. If the command is “hello”, respond with “Hello, <name>”, and log a new client
        b. If the command is “math”, evaluate the equation and send the evaluation response back
            i. Evaluation parses numbers and an operator from the input string, then calculates a response. If there is an issue with the input, it will return an appropriate error message instead
        c. If the command is “exit”, respond with “Bye, <name”, and log a client exit

iii. Go to step 6a