John Marker

Tyler Paquet

CIS 457 Project 1 Documentation

The first implementation of this project didn’t allow for diverse user input on the client side. Given the specifications for the second part of the project, we needed to introduce a new structure to the program to allow the user to not only request a file, but get a list of available files, requests one among them, and exit when desired. We explicitly tell the client each time it opens a connection to the server the available commands and retrieve user input:

**inFromUser = new BufferedReader(new InputStreamReader(System.in));**

**System.out.println("Commands: exit, send <filename>, list");**

**sentence = inFromUser.readLine();**

To allow the user on the client side to decide when the program closes, the solution was simply to keep essentially all of the client side code inside of a loop with the condition that the user input didn’t equal “exit”: **while (!exit)**. To implement a way to request files, we prompted the user to use the keyword “send” followed by the filename requested.

**if (sentence.substring(0, 4).toLowerCase().equals("send"));**

Similarly, if the client wanted a list of files on the server, we prompted the user to use the keyword “list”

**else if (sentence.substring(0, 4).toLowerCase().equals("list"));**

The user input taken from the client is sent to the server. It then finds it’s way to the appropriate location given similar constraints applied to the client side and is handled accordingly. Since interaction between the client and server is more extensive, we thought it was necessary to send acknowledgments from the server for every client command.

Client Side:

**outToServer.writeBytes(sentence + '\n');**

**serverSentence = inFromServer.readLine();**

Server Side:

**clientSentence = inFromClient.readLine();**

**System.out.println("Client said: " + clientSentence);**

**outToClient.writeBytes(clientSentence + '\n');**