CS2003 Practical 1 Report

Tutor: Ozgur Akgun

Overview

For this practical I was asked to build a client and server programs using Java that follow a set of protocols and requirements outlined in the specification.

I achieved a successful client and server programs where the client is able to send data to the server and the server is able to send back the appropriate response in line with the protocols and requirements.

I have completed all parts fully, the client is able to send 'HELLO', 'WHAT IS THE TIME?', 'PLEASE ADD x y', and 'BYE' and receive the correct responses.

Design

To read and write between the client and server programs I decided to use a *BufferedReader* and *PrintWriter*. I chose these because *PrintWriter* has a parameter *autoFlush* that means that I didn't have to add *flush*() after each time I sent data between client and server as the *flush*() was done automatically. This method of reading in data was the simplest to do, compared to using for example *InputStream* and *OutputStream* directly.

When calculating the 'PLEASE ADD x y' part of the problem. To get the value of *x* and *y*. At first I read only the last and 3rd last character as that would be the last two values. This would be true if all x and y were single digit integers. However, to allow my program to be able to handle numbers of more than single digits. I started gathering the string from after the 'PLEASE ADD ' using *substring*().

To make sure only x and y, 2 integers were accepted I gathered the rest of the string after 'PLEASE ADD ', I split the string using .*split*() and used " " as the regex and made sure there were only 2 elements in the array. This didn't confirm that what has been entered for x and y were integers. To make sure they were integers I used *NumberFormatException*, and if this exception was caught 'PARDON' would be sent. I also used this in both my client and server programs when validating the port number is a number.

Testing

To carry out the tests on my programs I used text files and using the 'nc' command. The results of the tests are shown below:

What is being tested?	More details	Pre-conditions	Expected output	Actual output	TestData/.txt file
A normal conversation	Sending 'PLEASE ADD...' before 'HELLO' has been received	Start server and client	'PARDON'	'PARDON'	1-in.txt/ 1-out.txt
-	Sending 'BYE' before 'HELLO' has been received	Start server and client	'PARDON'	'PARDON'	1-in.txt/ 1-out.txt
-	Sending 'HELLO' before 'HELLO' has been received	Start server and client	'HI NICE TO MEET YOU'	'HI NICE TO MEET YOU'	1-in.txt/ 1-out.txt
-	Sending 'HELLO' after 'HELLO' has been received	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	1-in.txt/ 1-out.txt
-	Sending 'HI' after 'HELLO' has been received	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	1-in.txt/ 1-out.txt
-	Sending 'HELLOHELLO' before 'HELLO' has been received	Start server and client	'PARDON'	'PARDON'	1-in.txt/ 1-out.txt
-	Sending 'PLEASE ADD 10 12' after 'HELLO' has been received	Start server and client Send 'HELLO'	'THE SUM IS 22'	'THE SUM IS 22'	1-in.txt/ 1-out.txt
-	Sending 'PLEASE ADD -5 5' after 'HELLO' has been received	Start server and client Send 'HELLO'	'THE SUM IS 0'	'THE SUM IS 0'	1-in.txt/ 1-out.txt
-	Sending 'WHAT IS THE TIME?' after 'HELLO' has been received	Start server and client Send 'HELLO'	'THE TIME IS: 2022-09-27T18:09:44.191527376+01:00'	'THE TIME IS: 2022-09-27T18:09:44.191527376+01:00'	1-in.txt/ 1-out.txt

-	Sending 'PLEASE ADD CHEESE SAUSAGE' after 'HELLO' has been received	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	1-in.txt/ 1-out.txt
-	Sending 'PLEASE ADD 0 1 2' after 'HELLO' has been received	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	1-in.txt/ 1-out.txt
-	Sending 'PLEASE ADD012' after 'HELLO' has been received	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	1-in.txt/ 1-out.txt
-	Sending 'WHAT IS THE TIME TODAY?' after 'HELLO' has been received.	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	1-in.txt/ 1-out.txt
-	Sending 'WHAT IS THE TIME? LET ME KNOW' after 'HELLO' has been received	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	1-in.txt/ 1-out.txt
-	Sending 'GOODBYE' after 'HELLO' has been received	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	1-in.txt/ 1-out.txt
-	Sending 'BYE' after 'HELLO' has been received	Start server and client Send 'HELLO'	'GOODBYE' <Server and Client close>	'GOODBYE' <Server and Client close>	1-in.txt/ 1-out.txt
-	Sending 'PLEASE ADD 2 2' after 'BYE' has been received	Start server and client Send 'HELLO' Send 'BYE'	-	-	1-in.txt/ 1-out.txt
PLEASE ADD x y	Sending 'PLEASE ADD 45 55'	Start server and client Send 'HELLO'	'THE SUM IS 100'	'THE SUM IS 100'	2-in.txt/ 2-out.txt
-	Sending 'PLEASE ADD T R'	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	2-in.txt/ 2-out.txt
-	Sending 'PLEASE ADD R 7'	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	2-in.txt/ 2-out.txt
-	Sending 'PLEASE ADD 5 TT'	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	2-in.txt/ 2-out.txt

-	Sending 'PLEASE ADD 3 2 56'	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	2-in.txt/ 2-out.txt
-	SENDING 'PLEASE ADD'	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	2-in.txt/ 2-out.txt
-	SENDING 'PLEASE ADD -1 1'	Start server and client Send 'HELLO'	'THE SUM IS 0'	'THE SUM IS 0'	2-in.txt/ 2-out.txt
WHAT IS THE TIME?	Sending 'WHAT IS THE TIME'	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	3-in.txt/ 3-out.txt
-	Sending 'WHAT IS THE TIME ?'	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	3-in.txt/ 3-out.txt
-	Sending 'TIME?'	Start server and client Send 'HELLO'	'PARDON'	'PARDON'	3-in.txt/ 3-out.txt

Evaluation

From the tests I have carried out above, as the expected output and actual output match for all the tests and due to the variation of tests carried out I believe this shows I have tested my program thoroughly and that my program is successful in meeting the protocol and specification provided.

Conclusion

In this project I have produced a successful client server program where the client is able to send data to the server and the server is able to send the correct data back to the client following the protocol.

At first I found it difficult to get the data sent from the client to the sever to be able to get to the server, and have the server send something in response. After I figured this out I found the rest of the practical enjoyable and relatively simple to see the next steps to creating a solution.

Given more time I would have liked to add more functionality to the server. For example, I would have liked to add more operations such as subtraction, multiplication and division. I would have also have liked to be able to develop the date and time portion of the program, and having the user/client side have more control over what part of the date and/or time that they seen. Rather than being forced to see all of it.