Noah Jaffe

CMSC 481

11/05/2018

TCP Socket Programming - The tic tac toe protocol:

Tic-Tac-Toe Protocol Explanation:

SENDING FROM SERVER TO CLIENT:

|  |  |
| --- | --- |
| pack message response length '!I' | packed\_msg\_len = struct.pack('!I', <unsigned int>) |
| SEND PACKED: MESSAGE RESPONSE LENGTH | connection.sendall(packed\_msg\_len) |
| SEND: MESSAGE | connection.sendall(<string>.encode()) |
| pack expecting response val '!I' | packed\_response\_val =  struct.pack('!I', <unsigned int>) |
| SEND PACKED: EXPECTING RESPONSE VAL | connection.sendall(packed\_response\_val) |

RECEIVING FROM SERVER TO CLIENT:

|  |  |
| --- | --- |
| RECV PACKED: MESSAGE RESPONSE LENGTH | packed\_msg\_len = connection.receive(4) |
| unpack '!I' and .recv that many bytes | msg\_len = struct.unpack('!I', packed\_msg\_len) |
| RECV: MESSAGE | message = connection.receive(msg\_len).decode() |
| RECV PACKED: EXPECTING RESPONSE VAL | packed\_response\_val = connection.receive(4) |
| unpack '!I' | response\_val =  struct.unpack('!I', packed\_response\_val) |

SENDING FROM CLIENT TO SERVER:

|  |  |
| --- | --- |
| pack single digit val '!I' | packed\_int\_val = struct.pack('!I', <unsigned int>) |
| SEND PACKED: SINGLE DIGIT VAL | connection.sendall(packed\_int\_val) |

RECEIVING FROM CLIENT TO SERVER:

|  |  |
| --- | --- |
| RECV PACKED: SINGLE DIGIT VAL | packed\_int\_val = connection.recv(4) |
| unpack '!I' | int\_val = struct.unpack('!I', packed\_int\_val) |

CONSTANTS AND GLOBALS:

TTT\_SERVER\_PORT = 13037

TTT\_PRTCL\_TERMINATE = 0

TTT\_PRTCL\_EXPECTING\_NO\_RESPONSE = 1

TTT\_PRTCL\_EXPECTING\_INT\_RESPONSE = 2

TTT\_PRTCL\_EXPECTING\_FIRST\_ARGS\_RESPONSE = 3

TTT\_PRTCL\_PACKED\_UNSIGNED\_INT\_SIZE = 4 #4 is the size of a packed '!I' value

TTT\_PRTCL\_REQUEST\_FIRST\_ARGS =

Please send an unsigned int representing if the client wishes to make the first move.

0 -- sever should go first

1 -- client should go first

TTT\_PRTCL\_GOT\_FIRST\_ARGS\_ERR =

Failed to receive proper game initiation arguments. Terminating connection.

Next time Please send an unsigned int representing if the client wishes to make the first move.

0 -- sever should go first

1 -- client should go first

TTT\_PRTCL\_INSTRUCTIONS =

Welcome to Tic Tac Toe!

Enter [0-8] for the position of your move, or 9 to quit:

0|1|2

-----

3|4|5

-----

6|7|8

TTT\_PRTCL\_INVALID\_CLIENT\_INPUT =

Invalid input, try again.

TTT\_PRTCL\_REQUEST\_CLIENT\_TURN =

| | 0|1|2

----- -----

| | 3|4|5

----- -----

| | 6|7|8

Enter [0-8] for the position of your move, or 9 to quit:

TTT\_PRTCL\_CLIENT\_ERR =

Sorry, that was invalid input. Please try again.

TO CREATE YOUR OWN CLIENT:

I have provided a ttt.py that defines the four functions described above and has all the constants defined inside.

To use it you just need to import it.

example usage:

import ttt

...

ttt.SEND\_FROM\_CLIENT\_TO\_SERVER(my\_socket\_connection, my\_data)