

There are very few libraries that were needed to successfully implement the code. However, those that were included were invaluable to the project

State has been placed into a stand alone file to ensure that it can be accessed by an file that requires that code

This also supports the integrity of the code, as ensures that State.py is not modified accidentally when working on the client or server code

The socket function provides a method to connect two different processes on the same machine. This is used a number of times throughout the project to ensure that each section connects in the three way handshake

The sys library provides system-specific parameters for python. This is included to allow command line arguments to be input. This allows the user greater control of the program, without needing to interact with the code directly

The TCP transition class demonstrates what would happen in the event of an error. Each of the errors is then printed to the console to alert the user. This provides valuable insight to the user when debugging the code

The synSent class in client.py provides the user with further feedback in the console, through the use of the print function. This class has an initializer and a number of functions defined within the class. The trigger function uses a number of if statements to send the correct information. This includes "Syn" and "Syn+Ack". The trigger method also uses try and except as error handling to deal with false values

The Closed class handles the code for exiting the program. When the project closes a print statement will be displayed to the user in the console from the synSent function. After this the trigger function will be called. It will close the socket and set the connection address to zero. When this has been completed a statement will be displayed to the console to let the user know that the connection has ended. The trigger function also has a try and except method to catch an error within the code

The Established class is called when the program has finished waiting. It then alerts the user that this process has completed. It will also send "Ack" to be encoded. After which the print method is then called to notify the user that the connection has been established via the console

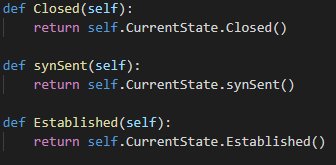
The finWait1 and 2 classes demonstrate what happens during the waiting period. This is highlighted during the simulation by printing that the console has finished its wait and is attempting to connect to the server. The trigger methods in both FinWait1 and 2 both use if/else statements to return either a true or false value

The finWait1 and 2 classes demonstrate what happens during the waiting period. This is highlighted during the simulation by printing that the console has finished its wait and is attempting to connect to the server. The trigger methods in both finWait1 and 2 both use if/else statements to return either a true or false value

The timedWait class sends the encoded "Ack" and prints that the program is waiting to the console. When this has been completed the program will close

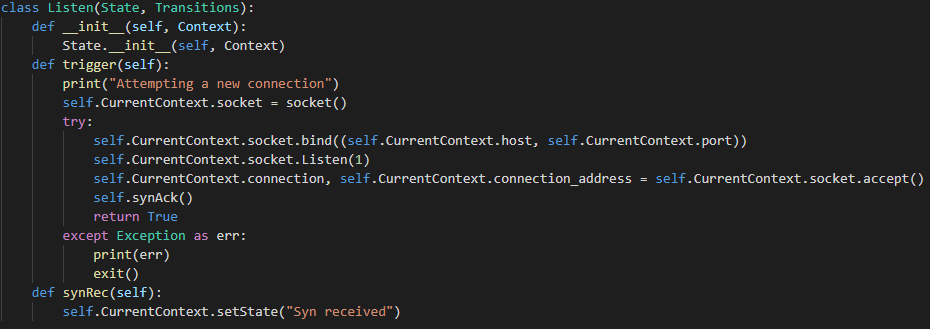
The TCPClient class initiates users the self parameter to reference the instance of the class and to access variables that belongs to the class. This is to ensure that the interpreter can receive the instance

The final section of client.py is a list of functions to call to transition between the functions. This will allow the simulation to run



The server.py file also has a Closed class, but its output is different to that of the client.py file. The structure of the code is similar, but the functionality is different. A try and except method is used for error handling, but the final part of the code is not a print statement to alert the user that the connecting has ended, but to call the Listen function.

The Listen class exists to bind the host and the port together to allow the connection to function. When this has happened the synAck function will be called. The user will be notified that the program is attempting to connect, as a message will be displayed within the console, informing the user that the program is trying a new connection.



The closeWait class handles the end of the program code from the server side. It will alert the user that the program has finished by printing that the connection is ending to the console. It will also print further information for the user if the if condition is satisfied.

The synRec class handles the code for when Syn is received. The user will first become aware that this process has completed when a message is displayed in the console that reads "Syn received". The program will the send a further message of "Syn+Ack" and then progress to an if statement for further progress.

The Established class in server provides the option for the user to input the message that they wish to send. First the program will provide information via the command line to inform the user that the connection has been successfully established. Then, using the input function, the user will be prompted to enter information. This data is then sent. The if statement will provide the relevant response, depending on what the client has responded with

The lastAck class provides the last section of the three way handshake. In effect it is a wait state for closing. When the program has completed the functions within the class, it can end the connection.

