Dylan Schneider, Roark Myers, Josias Polonia

Network Design

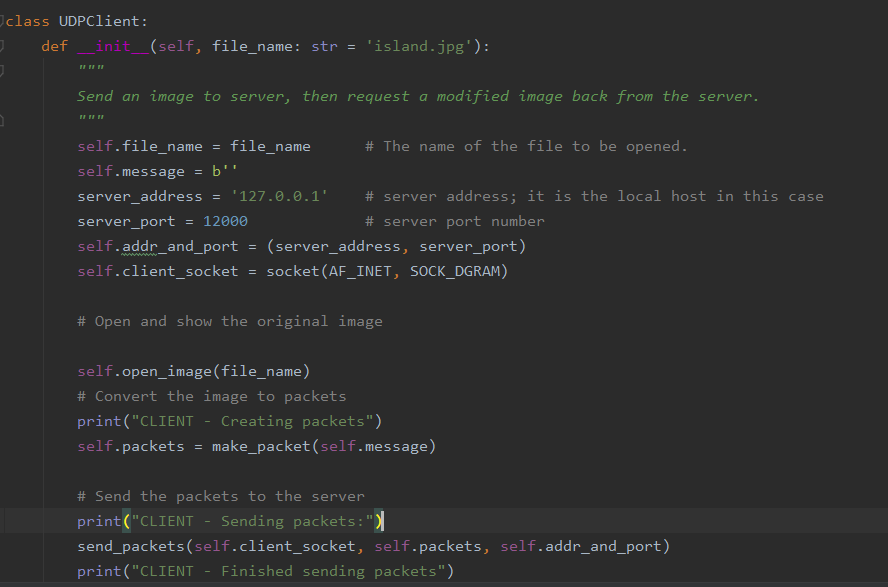
EECE 4830

Vokkarane

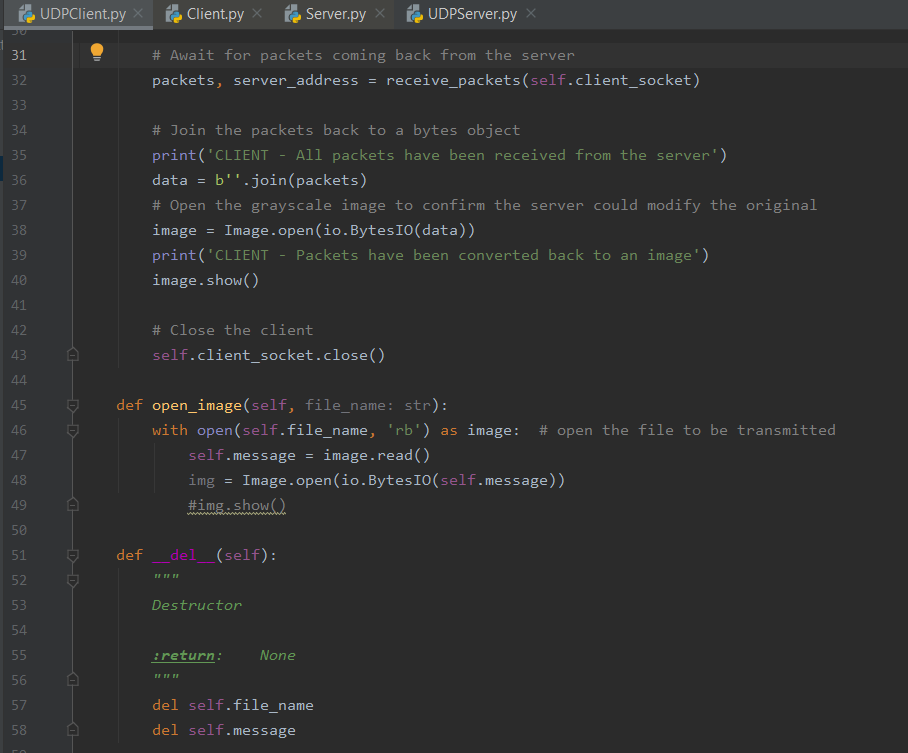
Network Design Project: Phase 2

The purpose of this phase is to transfer an image file such as a bmp between a UDP client and server like phase 1, but time we provide reliable data transfer. With this method the image file transferred is split into several packets and those packets are sent over one at a time. Also, the one receiving the packets will continue to read in the packets until it receives a terminator message; after that, it will display the image and then repeat the process to send the image back to the sender (whether client or server).

*Client:*

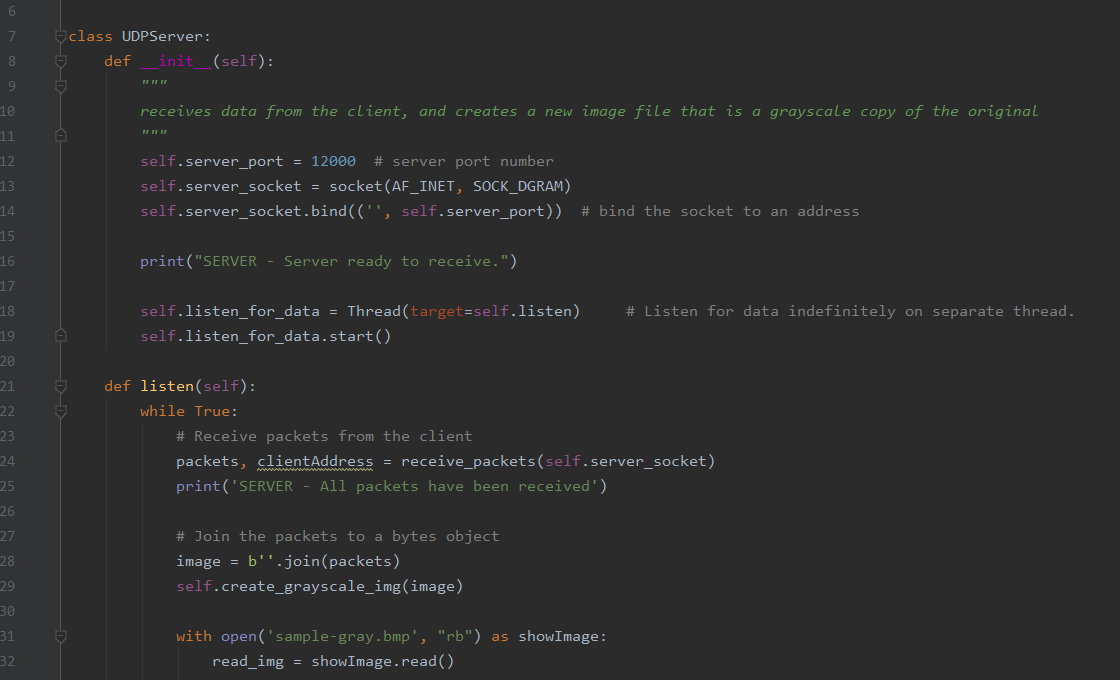


The UDPClient class creates a definition for the image file name prior to sending it to the server. After defining the address and port of the server, self creates access to the class. Line 17 assigns the client socket, address family and that it is a UDP socket equal to the client socket. The image is converted to packets by the make\_packet function. Then the packets are sent over as seen in line 28 using that particular function until all of them are finally sent.

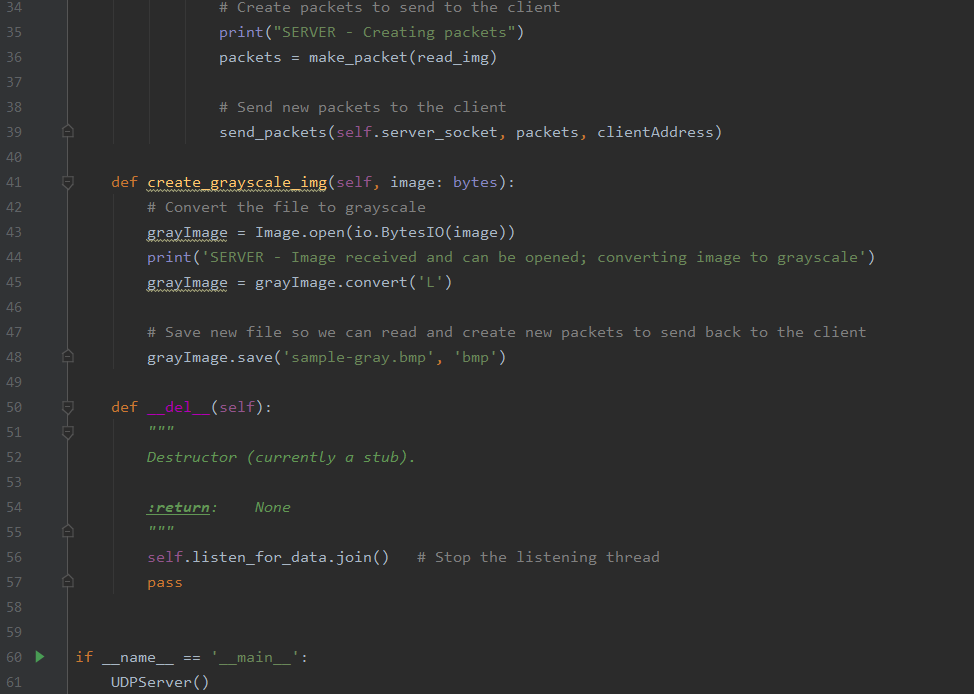


As to be seen later, the server receives the packets and then eventually sends the packets back over. Once it is being received back, the packets are then stored back into the client to a byte object. The image is then converted to grayscale image and finally closed. The Line 45 contains a function to open complete image, read it in once again. Finally, the last function is a destructor that gets called to let the sender know that all the packets have been sent.

*Server*



The UDPServer class contains the \_init\_(self) function that deals with receiving process from the client. It makes a new image becomes a grayscale from the original. Line 14 connects the other socket to the address. Line 18 lets the function listen for data function run on its own and then it begins. The server continues to read in packets from the client until all the packets have been sent over. Packets are then combined into a byte object .



New packets are then sent back to the client in line 39. The next function in line 41 converts the image file to a grayscale and then it is saved; we save it for future evaluation and sending it back to the client. Like the client side, the destructor function is called when all the packets are sent from one side to the other as an indicator.

*Execution Example*

*Client:*





*Server:*





*Client:*

