Python Network Chat Tutorial

Contents

Time required: 30 minutes

Objective: Write a cross platform Python script that creates a local area network chat server and client.

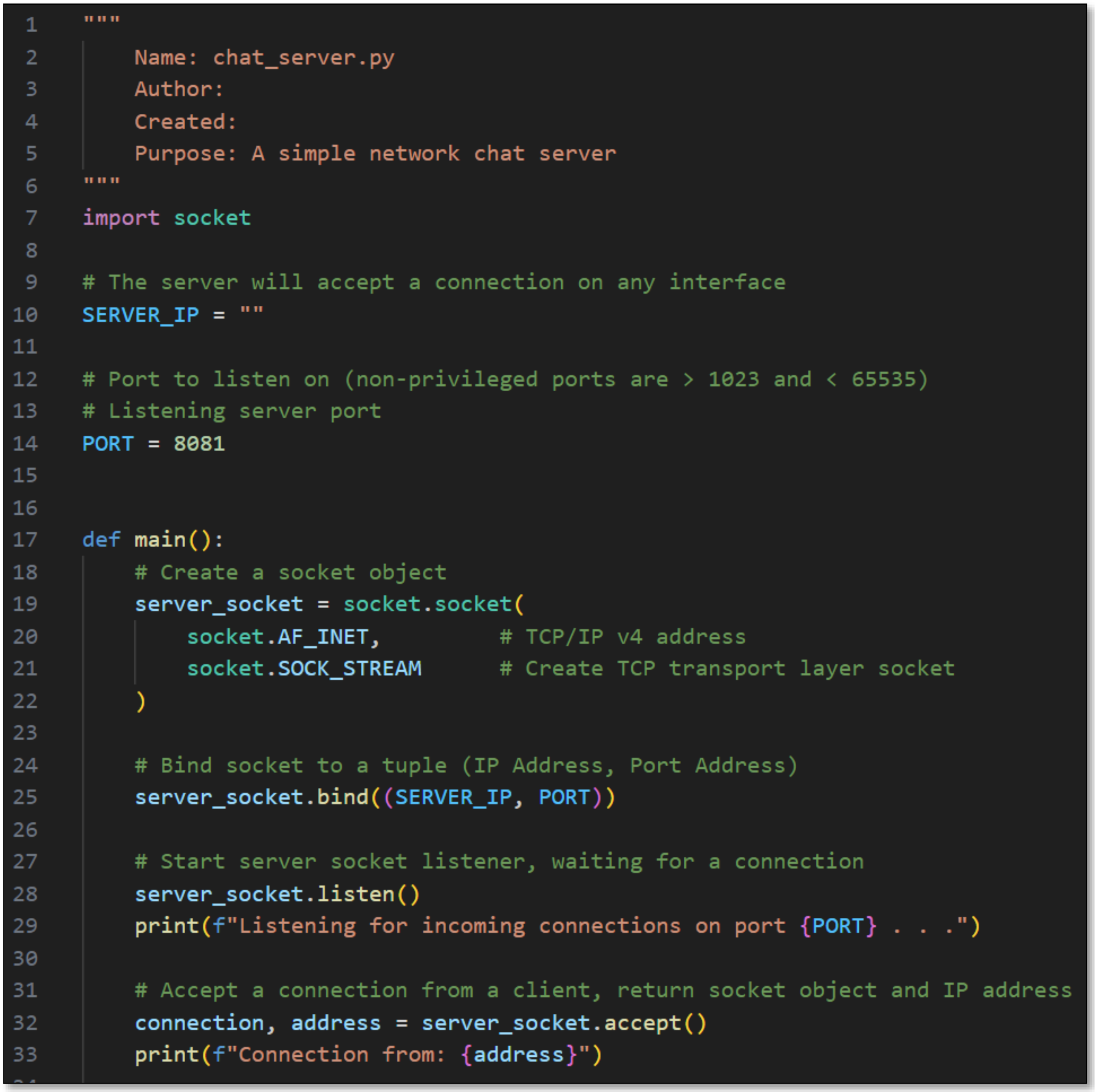
Network Chat Server

If you have previously completed Network Client Server, start with those files.

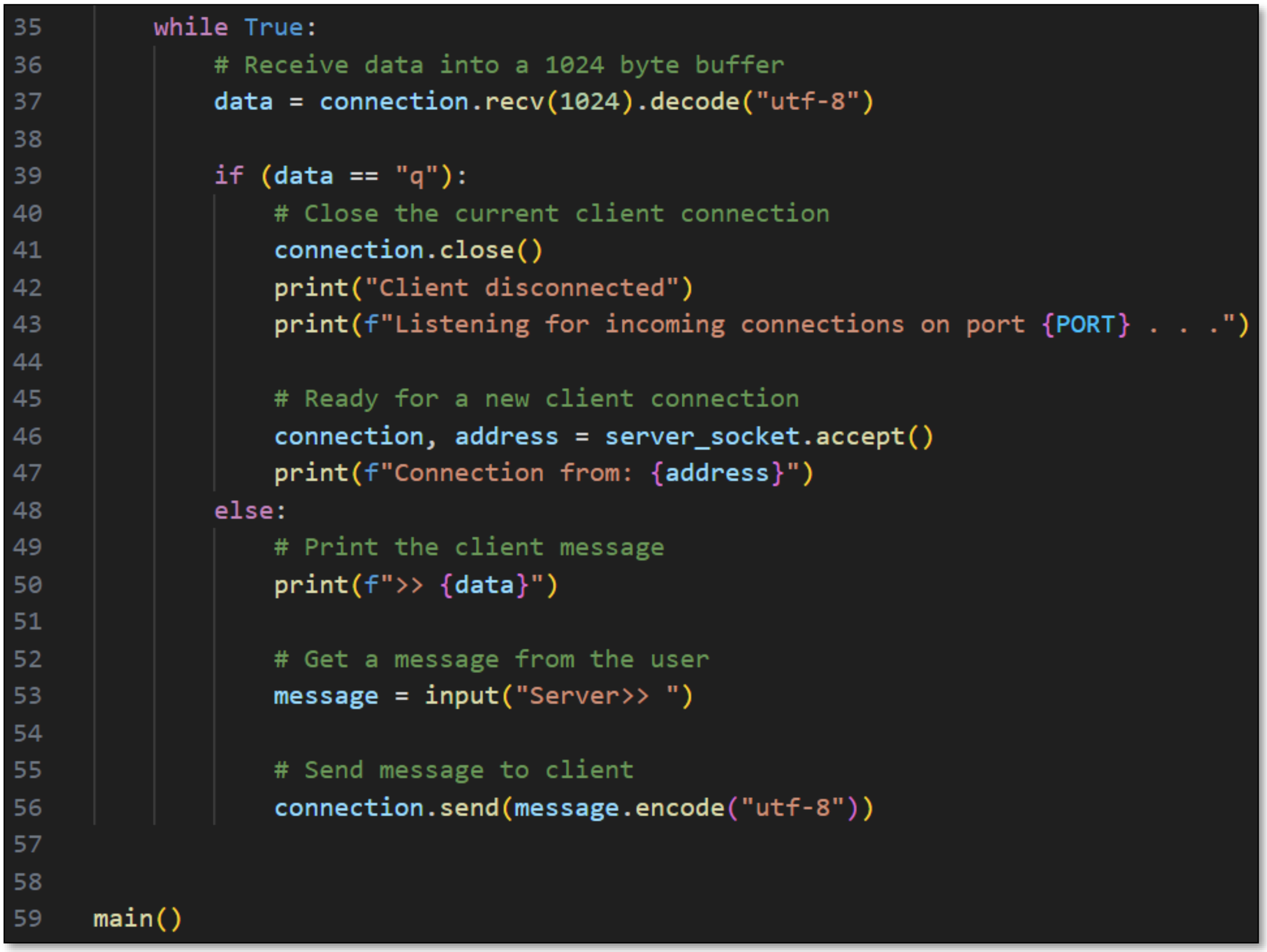
We are going to create a simple client server chat system.

Tutorial 1: Chat Server

1. Enter the following code including the comments



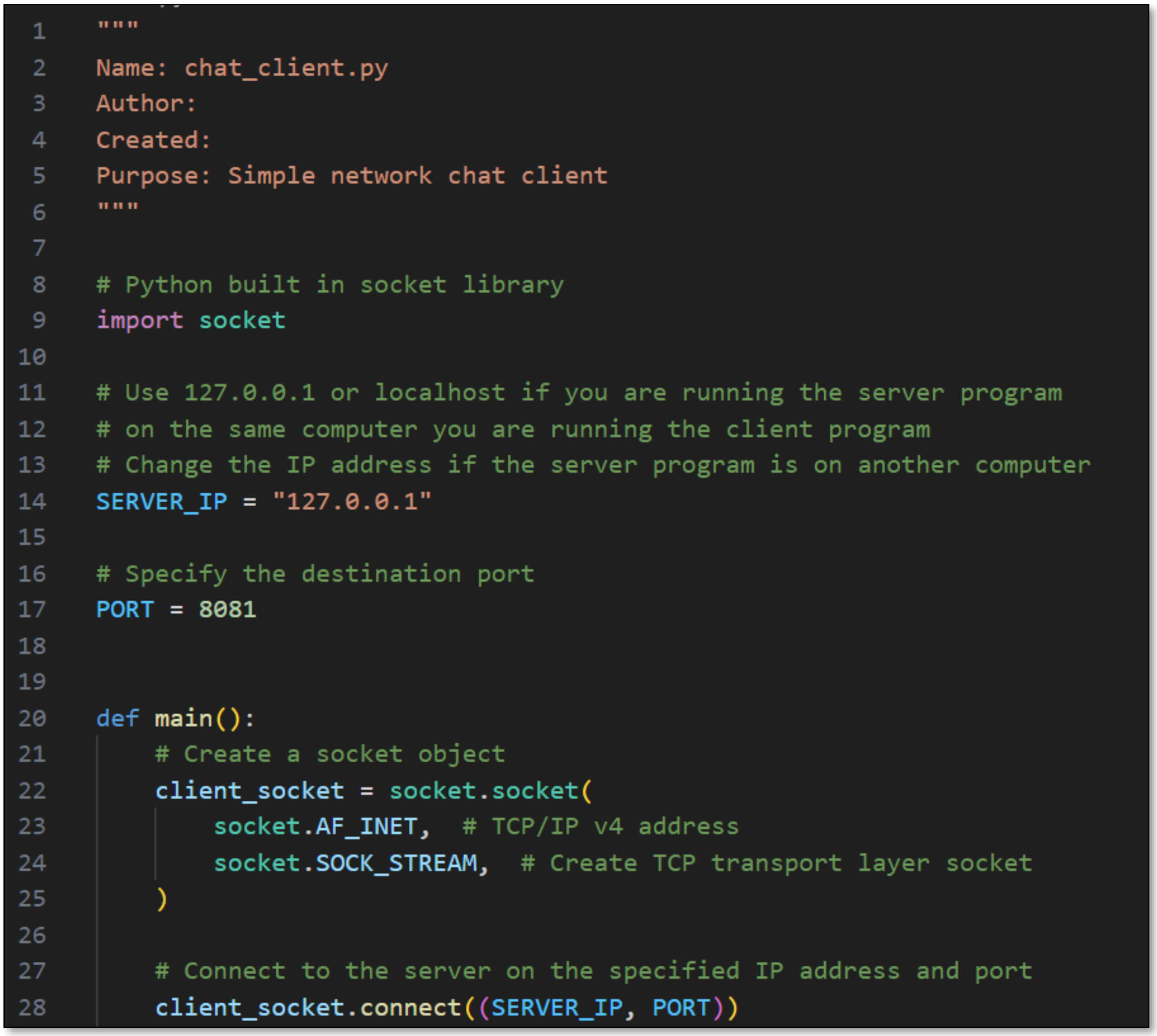
```python
"""
    Name: chat_server.py
    Author:
    Created:
    Purpose: A simple network chat server
"""
import socket

# The server will accept a connection on any interface
SERVER_IP = ""

# Port to listen on (non-privileged ports are > 1023 and < 65535)
# Listening server port
PORT = 8081


def main():
    # Create a socket object
    server_socket = socket.socket(
        socket.AF_INET,         # TCP/IP v4 address
        socket.SOCK_STREAM      # Create TCP transport layer socket
    )

    # Bind socket to a tuple (IP Address, Port Address)
    server_socket.bind((SERVER_IP, PORT))

    # Start server socket listener, waiting for a connection
    server_socket.listen()
    print(f"Listening for incoming connections on port {PORT} . . .")

    # Accept a connection from a client, return socket object and IP address
    connection, address = server_socket.accept()
    print(f"Connection from: {address}")
```



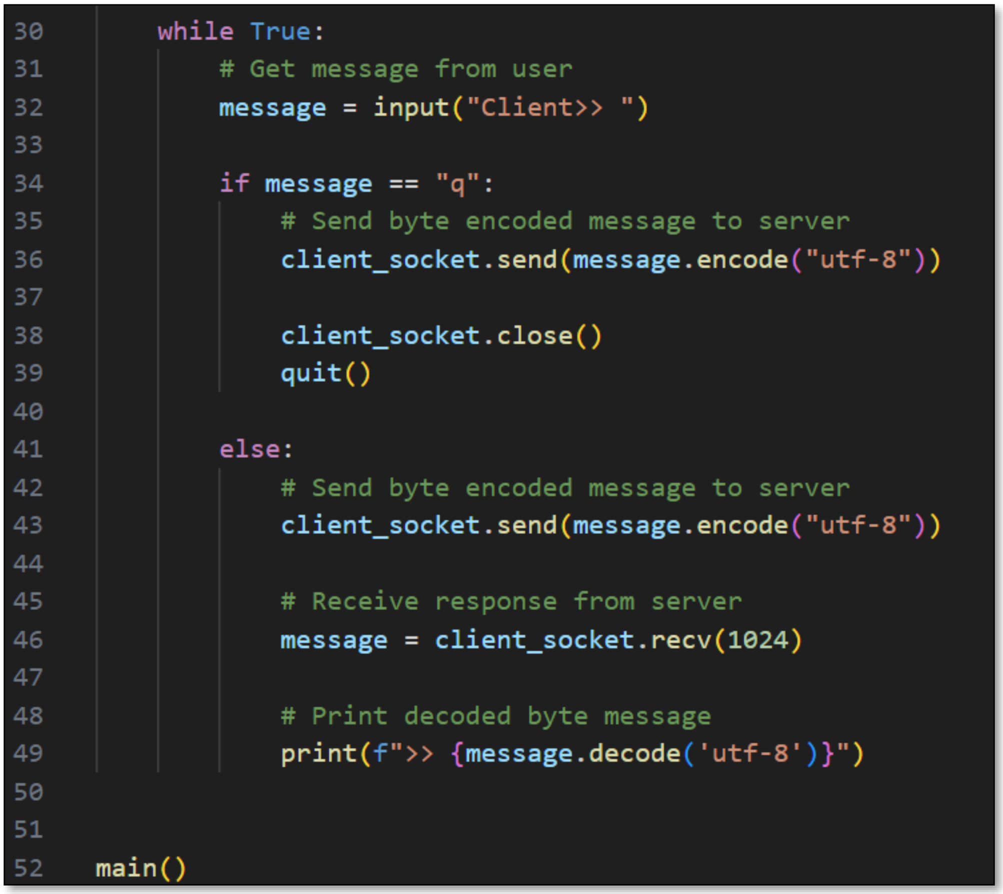
```
    while True:
        # Receive data into a 1024 byte buffer
        data = connection.recv(1024).decode("utf-8")

        if (data == "q"):
            # Close the current client connection
            connection.close()
            print("Client disconnected")
            print(f"Listening for incoming connections on port {PORT} . . .")

            # Ready for a new client connection
            connection, address = server_socket.accept()
            print(f"Connection from: {address}")
        else:
            # Print the client message
            print(f">> {data}")

            # Get a message from the user
            message = input("Server>> ")

            # Send message to client
            connection.send(message.encode("utf-8"))


main()
```

Tutorial 2: Chat Client

Enter the following code.



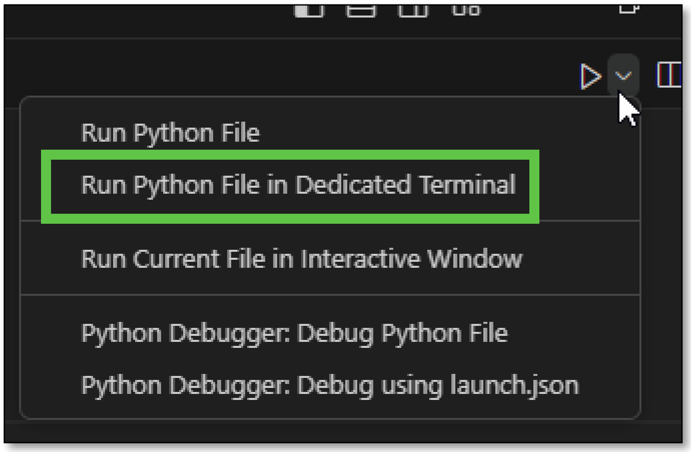
```python
"""
Name: chat_client.py
Author:
Created:
Purpose: Simple network chat client
"""

# Python built in socket library
import socket

# Use 127.0.0.1 or localhost if you are running the server program
# on the same computer you are running the client program
# Change the IP address if the server program is on another computer
SERVER_IP = "127.0.0.1"

# Specify the destination port
PORT = 8081


def main():
    # Create a socket object
    client_socket = socket.socket(
        socket.AF_INET,  # TCP/IP v4 address
        socket.SOCK_STREAM,  # Create TCP transport layer socket
    )

    # Connect to the server on the specified IP address and port
    client_socket.connect((SERVER_IP, PORT))
```



```python
    while True:
        # Get message from user
        message = input("Client>> ")

        if message == "q":
            # Send byte encoded message to server
            client_socket.send(message.encode("utf-8"))

            client_socket.close()
            quit()

        else:
            # Send byte encoded message to server
            client_socket.send(message.encode("utf-8"))

            # Receive response from server
            message = client_socket.recv(1024)

            # Print decoded byte message
            print(f">> {message.decode('utf-8')}")


main()
```

How to Run the Server and the Client

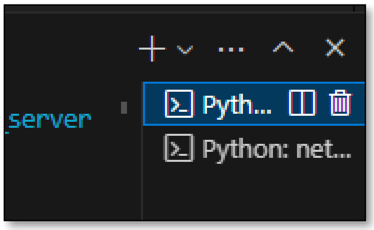
1. Run the server program first in a dedicated terminal.



Run Python File
Run Python File in Dedicated Terminal
Run Current File in Interactive Window
Python Debugger: Debug Python File
Python Debugger: Debug using launch.json

2. Run the client program normally.

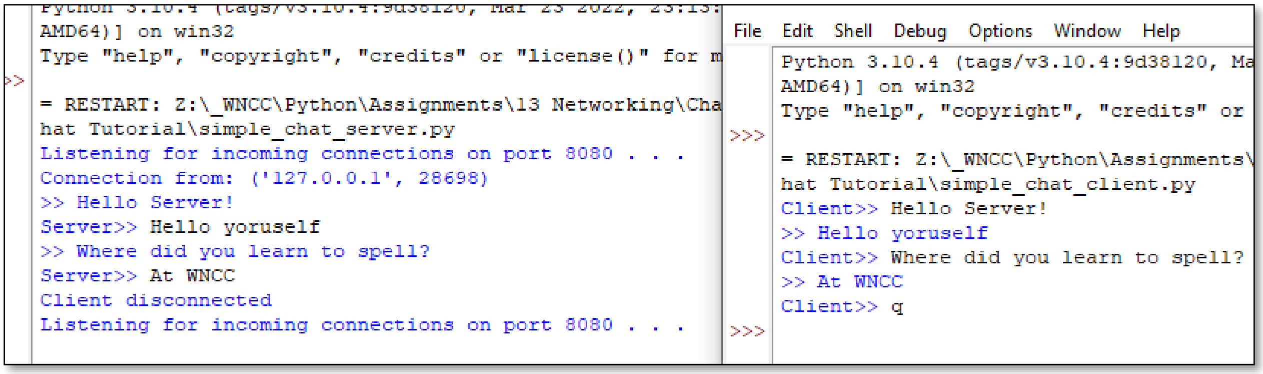
3. You will see the two terminals to the right middle side. Switch from one to the other to see the different program runs.



Assignment 1: Chat Time

1. Start the chat server.
2. Start the chat client.
3. Type a message in the client. It should show up in the server.
4. Type a message in the server. It should show up in the client.
5. Do this a few times.
6. Insert a screenshot with the assignment showing this conversation.

Example run:



```
Python 3.10.4 (tags/v3.10.4:9d38120, Mar 23 2022, 23:13:
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for m
>>
= RESTART: Z:\_WNCC\Python\Assignments\13 Networking\Cha
hat Tutorial\simple_chat_server.py
Listening for incoming connections on port 8080 . . .
Connection from: ('127.0.0.1', 28698)
>> Hello Server!
Server>> Hello yoruself
>> Where did you learn to spell?
Server>> At WNCC
Client disconnected
Listening for incoming connections on port 8080 . . .
```

```
Python 3.10.4 (tags/v3.10.4:9d38120, Ma
AMD64)] on win32
Type "help", "copyright", "credits" or
>>>
= RESTART: Z:\_WNCC\Python\Assignments\
hat Tutorial\simple_chat_client.py
Client>> Hello Server!
>> Hello yoruself
Client>> Where did you learn to spell?
>> At WNCC
Client>> q
>>>
```

Assignment Submission

1. Attach the program files.
2. Attach screenshots showing the successful operation of the program.
3. Submit in Blackboard.