Socket Programming in Python

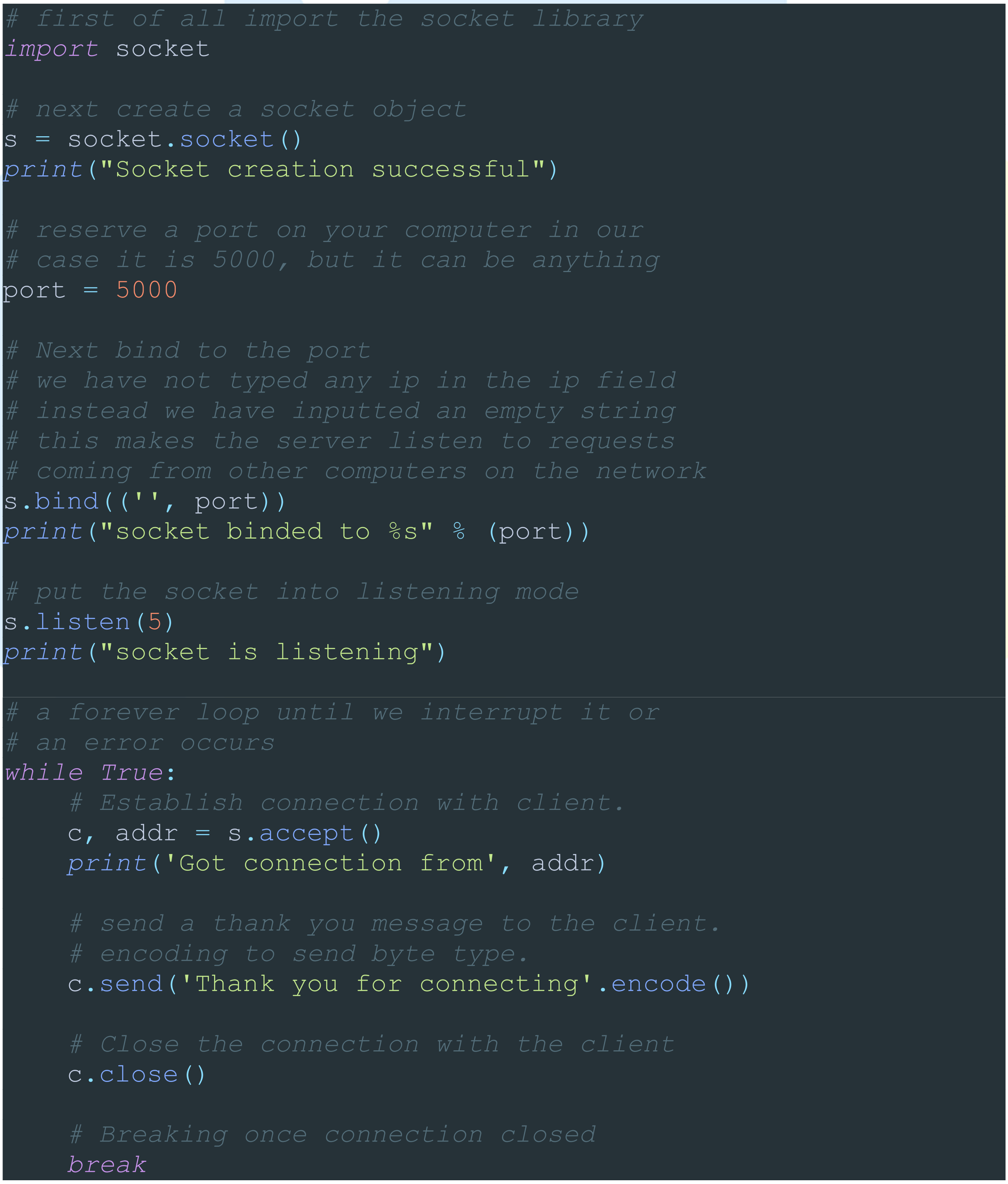
Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket(node) listens on a particular port at an IP, while the other socket reaches out to the other to form a connection. The server forms the listener socket while the client reaches out to the server.

They are the real backbones behind web browsing. In simpler terms, there is a server and a client.

Building a simple client-server program

SERVER:

A server has a bind() method which binds it to a specific IP and port so that it can listen to incoming requests on that IP and port. A server has a listen() method which puts the server into listening mode. This allows the server to listen to incoming connections. And last a server has an accept() and close() method. The accept method initiates a connection with the client and the close method closes the connection with the client. Create a new file and name it ***server.py***.



```
# first of all import the socket library
import socket

# next create a socket object
s = socket.socket()
print("Socket creation successful")

# reserve a port on your computer in our
# case it is 5000, but it can be anything
port = 5000

# Next bind to the port
# we have not typed any ip in the ip field
# instead we have inputted an empty string
# this makes the server listen to requests
# coming from other computers on the network
s.bind(('', port))
print("socket binded to %s" % (port))

# put the socket into listening mode
s.listen(5)
print("socket is listening")

# a forever loop until we interrupt it or
# an error occurs
while True:
    # Establish connection with client.
    c, addr = s.accept()
    print('Got connection from', addr)

    # send a thank you message to the client.
    # encoding to send byte type.
    c.send('Thank you for connecting'.encode())

    # Close the connection with the client
    c.close()

    # Breaking once connection closed
    break
```

- First, we import socket which is necessary.
- Then we made a socket object and reserved a port on our pc.
- After that, we bound our server to the specified port. Passing an empty string means that the server can listen to incoming connections from other computers as well. If we would have passed 127.0.0.1 then it would have listened to only those calls made within the local computer.
- After that we put the server into listening mode.5 here means that 5 connections are kept waiting if the server is busy and if a 6th socket tries to connect then the connection is refused.
- At last, we make a while loop and start to accept all incoming connections and close those connections after a thank you message to all connected sockets.

CLIENT

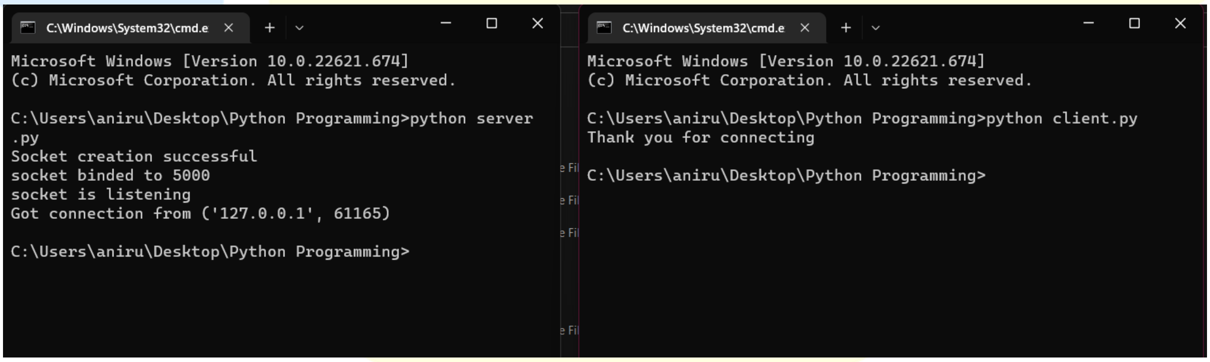
Now we need something with which a server can interact. We could tenet to the server like this just to know that our server is working.

Create a new file and name it ***client.py***.

```python
# Import socket module
import socket

# Create a socket object
s = socket.socket()

# Define the port on which you want to connect
port = 5000

# connect to the server on local computer
s.connect(('127.0.0.1', port))

# receive data from the server and decoding to get the string.
print(s.recv(1024).decode())
# close the connection
s.close()
```

- First, we make a socket object.
- Then we connect to localhost on port 12345 (the port on which our server runs) and lastly, we receive data from the server and close the connection.

Run the ***server.py*** file first and then the ***client.py***. See the output below



```
Microsoft Windows [Version 10.0.22621.674]
(c) Microsoft Corporation. All rights reserved.

C:\Users\aniru\Desktop\Python Programming>python server
.py
Socket creation successful
socket binded to 5000
socket is listening
Got connection from ('127.0.0.1', 61165)

C:\Users\aniru\Desktop\Python Programming>
```

```
Microsoft Windows [Version 10.0.22621.674]
(c) Microsoft Corporation. All rights reserved.

C:\Users\aniru\Desktop\Python Programming>python client.py
Thank you for connecting

C:\Users\aniru\Desktop\Python Programming>
```