

Socket programming in python

Python provides two levels of access to network services. At a low level, you can access the basic socket support in the underlying operating system, which allows you to implement clients and servers for both connection-oriented and connectionless protocols.

Python also has libraries that provide higher-level access to specific application-level network protocols, such as FTP, HTTP, and so on.

This chapter gives you understanding on most famous concept in Networking - Socket Programming.

Sockets are the endpoints of a bidirectional communications channel. Sockets may communicate within a process, between processes on the same machine, or between processes on different continents.

Sockets may be implemented over a number of different channel types: Unix domain sockets, TCP, UDP, and so on. The *socket* library provides specific classes for handling the common transports as well as a generic interface for handling the rest.

Client

To write Internet servers, we use the **socket** function available in socket module to create a socket object. A socket object is then used to call other functions to setup a socket server.

Now call **connet(hostname, port)** function to specify a *port* for your service on the given host.

Server

Let us write a very simple client program which opens a connection to a given port 8000 and given host. This is very simple to create a socket client using Python's *socket* module function.

The **socket.connect(hosname, port)** opens a TCP connection to *hostname* on the *port*. Once you have a socket open, you can read from it like any IO object. When done, remember to close it, as you would close a file.

Code for server

```
import socket

skt = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
skt.bind((socket.gethostname(), 8000))
skt.listen(5)

while True:
    clientsocket, address = skt.accept()
```

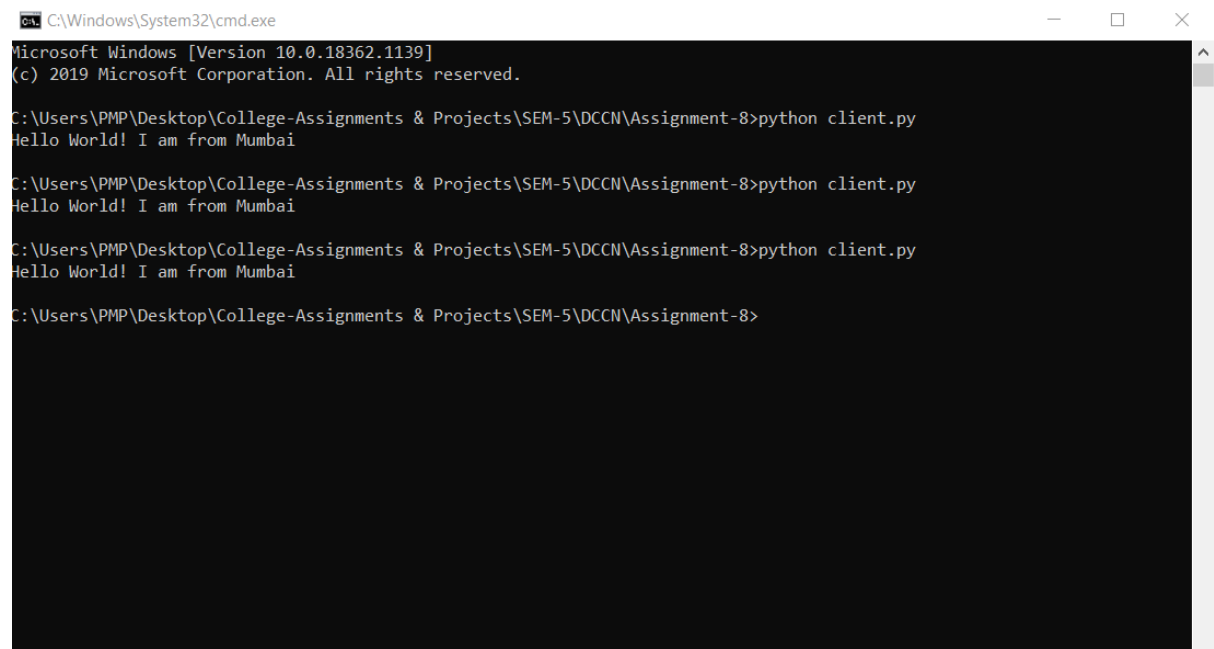
```
print(f'Connection successfully established with the address {address}')
clientsocket.send(bytes('Hello World! I am from Mumbai', 'utf-8'))
clientsocket.close()
```

Code for client

```
import socket

skt = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
skt.connect((socket.gethostname(), 8000))
msg = skt.recv(1024)
print(msg.decode('utf-8'))
```

Client cmd



The screenshot shows a Windows command prompt window titled "C:\Windows\System32\cmd.exe". The window displays the output of running a Python script named "client.py" three times. Each time the script is executed, it prints "Hello World! I am from Mumbai". The command prompt shows the current directory as "C:\Users\PMP\Desktop\College-Assignments & Projects\SEM-5\DCCN\Assignment-8".

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.18362.1139]
(c) 2019 Microsoft Corporation. All rights reserved.

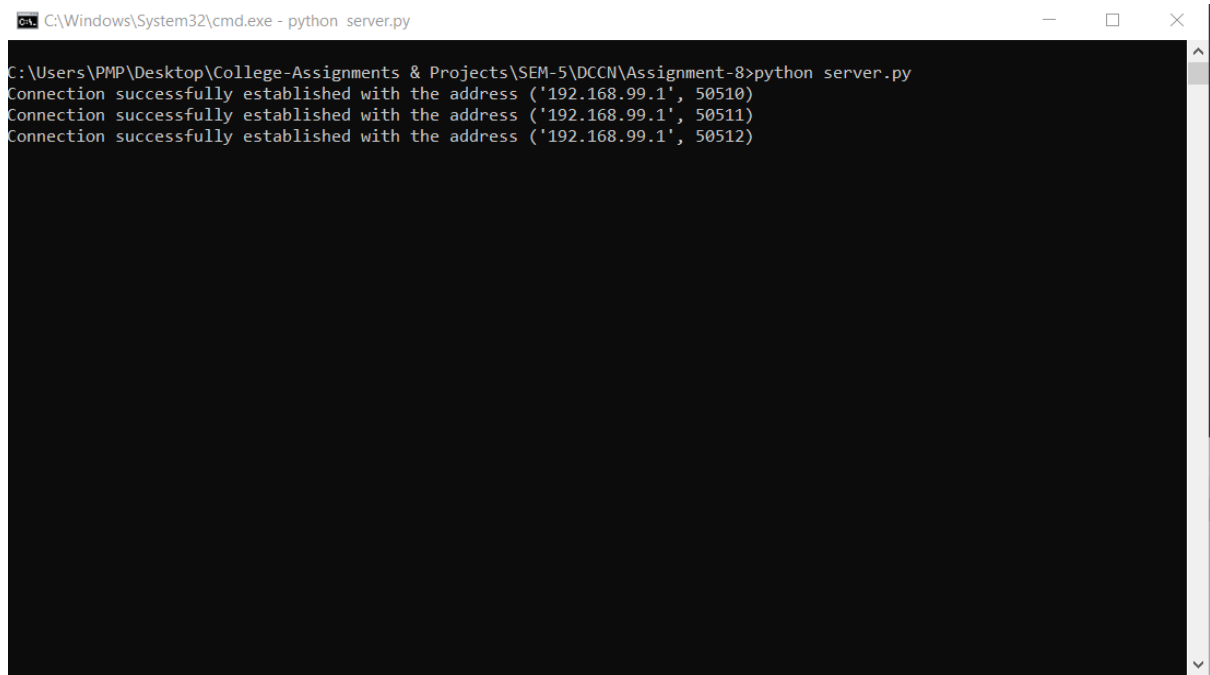
C:\Users\PMP\Desktop\College-Assignments & Projects\SEM-5\DCCN\Assignment-8>python client.py
Hello World! I am from Mumbai

C:\Users\PMP\Desktop\College-Assignments & Projects\SEM-5\DCCN\Assignment-8>python client.py
Hello World! I am from Mumbai

C:\Users\PMP\Desktop\College-Assignments & Projects\SEM-5\DCCN\Assignment-8>python client.py
Hello World! I am from Mumbai

C:\Users\PMP\Desktop\College-Assignments & Projects\SEM-5\DCCN\Assignment-8>
```

Sever cmd

A screenshot of a Windows command prompt window. The title bar at the top reads "C:\Windows\System32\cmd.exe - python server.py". The command prompt shows the following text:
C:\Users\PMP\Desktop\College-Assignments & Projects\SEM-5\DCCN\Assignment-8>python server.py
Connection successfully established with the address ('192.168.99.1', 50510)
Connection successfully established with the address ('192.168.99.1', 50511)
Connection successfully established with the address ('192.168.99.1', 50512)
The rest of the window is black, indicating the program has finished execution or is waiting for input. A vertical scrollbar is visible on the right side of the command prompt area.

```
C:\Windows\System32\cmd.exe - python server.py
C:\Users\PMP\Desktop\College-Assignments & Projects\SEM-5\DCCN\Assignment-8>python server.py
Connection successfully established with the address ('192.168.99.1', 50510)
Connection successfully established with the address ('192.168.99.1', 50511)
Connection successfully established with the address ('192.168.99.1', 50512)
```

Conclusion –

After completing the above experiment, I have understood that how a client and server works and how do they communicate.