

LAB # 3

“SOCKET PROGRAMMING WITH MULTI-THREADING”

In the previous lab we learnt to connect a client to server. In this lab we will study about socket programming with multithreading. Let's revise a few basic concepts first.

Thread:

A thread is a light-weight process that does not require much memory overhead, they are cheaper than processes.

Multi-threading:

Multithreading is a process of executing multiple threads simultaneously in a single process.

Multi-threading Modules In Socket Programming:

Following are the modules that can be used to achieve multi-threading in python:

- `_thread`
- `Threading`

These modules help in synchronization and provides lock. Lock has two states “locked” and “unlocked”.

`.acquire()` is used to change state to locked

`.release()` is used to change state to unlock.

`threading.Thread()` is used to start a new thread in socket programming.

It takes 2 arguments, first is the function upon which we want to apply multi-threading and second is a tuple holding the address of client.

Let's study client-server multithreading socket programming by code.

Multi-threaded Server Code:

Firstly, import the necessary modules.

```
import socket
import threading
```

```
print_lock = threading.Lock()
def createThread(c,addr ):
    print(f"new connection : {addr}")
    print(f"threadName : ", {threading.currentThread().getName()})
    while True:
        data = c.recv(1024)
        print('Received from the client :', str(data.decode('ascii')))
        if not data:
            print('Bye')
            print_lock.release()
            break
        data = "Today we will learn about Multithreading with Socket programming"
        c.send(data.encode('utf-8'))
    c.close()
def Main():
    host = "localhost"
    port = 2022
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.bind((host, port))
    print("socket binded to port", port)
    s.listen()
    print("Server is listening, Waiting for incoming ")
    while True:
        c, addr = s.accept()
        print_lock.acquire()
        print('Connected to :', addr[0], ':', addr[1])
        print(f"threadName , before creating thread: : ", {threading.currentThread().getName()})
        thread= threading.Thread(target=createThread, args=(c,addr))
        thread.start()
    s.close()
```

Don't forget to call Main().

Client Code:

```
import socket

host = '127.0.0.1'

port = 2022
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((host, port))
message = "Hey Server!!! What will we do in today's lab???"
while True:
    s.send(message.encode('utf-8'))
    data = s.recv(1024)
    print('Received from the server :', str(data.decode('utf-8')))
    reply = input('\nDo you want to continue(y/n) :')
    if reply == 'y':
        continue
    else:
        print("Bye Server")
        break
s.close()
```

Output of Server code:

```
socket binded to port 2022
Server is listening, Waiting for incoming
Connected to : 127.0.0.1 : 50376
threadName , before creating thread: : {'MainThread'}
new connection : ('127.0.0.1', 50376)
threadName : {'Thread-1'}
Received from the client : Hey Server!!! What will we do in today's lab???
Received from the client :
Bye
Connected to : 127.0.0.1 : 50377
threadName , before creating thread: : {'MainThread'}
new connection : ('127.0.0.1', 50377)
threadName : {'Thread-2'}
Received from the client : Hey Server!!! What will we do in today's lab???
Received from the client :
Bye
```

Client with port number **50376** is served by “**Thread-1**”

Client with port number **50377** is served by “**Thread-2**”

Output of Client/s Code:

```
Received from the server : Today we will learn about Multithreading with Socket programming
Do you want to continue(y/n) :y
Received from the server : Today we will learn about Multithreading with Socket programming
Do you want to continue(y/n) :n
Bye Server
```
