

컴퓨터 네트워크

- 03. HTTP -

※ 과제 해결 방법

- tcp_server.py -

```
import socket
import os
from threading import Thread
import traceback

ip_addr = '127.0.0.1'
port_num = 2345

server_sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_sock.bind((ip_addr, port_num))
print("Server socket open...")

print("Listening...")
server_sock.listen(5)

def clnt_thread(clnt_sock):
    data = clnt_sock.recv(5000)

    de_data = data.decode()
    parsing_data = de_data.split(' ')


    is_file = 'HTTP/1.1 200 OK\r\n\r\n'.encode()
    none_file = 'HTTP/1.1 404 Not Found\r\n\r\n'.encode()


    file_name = parsing_data[1].split('/')[1]

    if parsing_data[0] == "GET":
        if os.path.isfile(file_name):
            f = open(file_name, 'r')
            line = f.read()
            is_file += line
            f.close()
            print(is_file)
            clnt_sock.send(is_file)

        else:
            clnt_sock.send(none_file)

    elif parsing_data[0] == "POST":
        print("Next Time")
```

 : listen(5)를 실행해서 클라이언트를 5개를 받을 수 있도록 한다.

 : clnt_thread 함수를 선언한다. 여기서는 쓰레드가 생성된 후 한 개의 요청을 처리해주는 함수이다.

- ① 클라이언트로부터 데이터를 받는다.
- ② 받은 데이터를 디코딩해서 공백을 기준으로 나눈다.
- ③ 파일이 있을 경우, 없을 경우 일 때의 헤더를 변수로 저장해준다.
- ④ 파싱한 데이터의 1번째 인덱스에 위치한 값을 다시 '/'으로 나눠준다.
- ⑤ 요청한 HTTP 메소드가 GET인지 POST인지 판별한다.
- ⑥ GET이라면 요청한 파일이 존재하는지 확인 후 있다면 파일을 읽어서 파일이 존재할 때의 헤더를 붙인다.
- ⑦ 클라이언트에게 데이터를 보내준다.


```

while True:
    clnt_sock, addr = server_sock.accept()
    # print("Connected with " + "client" + str(i))

    try:
        Thread(target=clnt_thread, args=(clnt_sock,)).start()
    except:
        print("Thread did not start.")
        traceback.print_exc()

print("Send Message back to client")
clnt_sock.close()

```

 : 여러 개의 클라이언트를 받기 위해서 스레드를 생성해준다. 반복문을 실행해서 여러 개의 스레드를 생성할 수 있도록 한다.

- tcp_client.py -

```

import socket
import sys

serverIP = '127.0.0.1'
serverPort = 2345

clnt_sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
clnt_sock.connect((serverIP, serverPort))
print("Connect to Server...")


clnt_msg = sys.argv[1] + " " + sys.argv[2]
clnt_sock.send(clnt_msg.encode())
print("Send Message to Server...")

recv_data = (clnt_sock.recv(1024)).decode('utf-8')
status = recv_data.split(' ')[1]

if status == "200":
    f = open("recv_data.html", 'w')
    f.write(recv_data.split('\n\n')[1])
    f.close()

elif status == "404":
    print("404 Not Found")

```

 : 클라이언트에서는 서버에게 요청을 하고 서버에게 데이터를 받아서 상태가 200이면 파일을 써준다. 404이면 Not Found를 출력해준다.

