

컴퓨터 네트워크

TCP 기반 Socket Programming
HTTP Request / Response

소스 파일 및 동작 환경

[개발 환경]

운영체제
작성언어
IDE

MS Windows 11 (10 ↑)
Python 3.7.9
VS Code

[Python Module]

socket
datetime

[실행 환경]

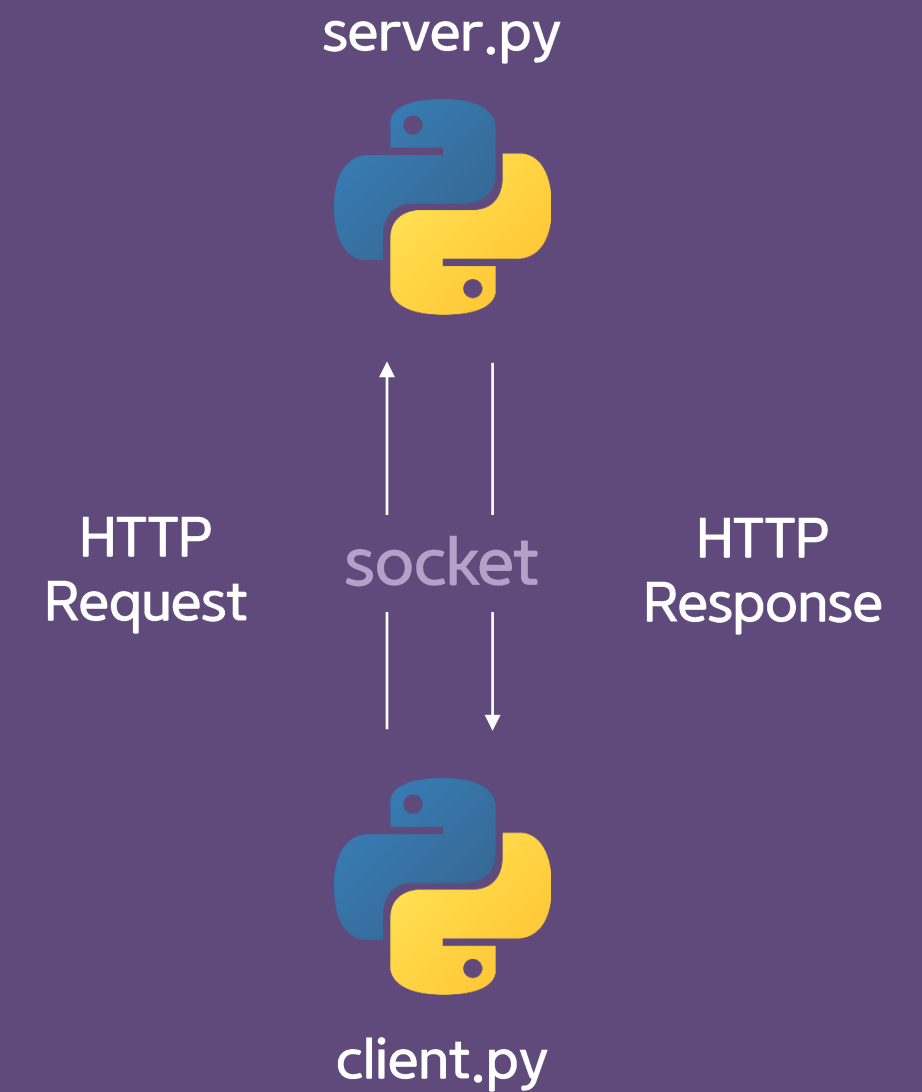
Server
Client

Labtop
Labtop
Postman

[디렉토리 구조]

```
src/  
└── server.py (서버 사이드 프로그램)  
    client.py (클라이언트 사이드 프로그램)  
  
.gitattributes (깃허브 파일)  
.gitignore (깃허브 파일)  
README.md (사용법)
```

[동작 흐름]



실행 방법 및 가이드

서버 실행

python server.py



클라이언트 실행

python client.py



클라이언트 요청

```
METHOD >>  
    GET | POST | DELETE  
    (외에는 405 Error)  
URL >>  
    /  
    (URL 중첩은 404 Error)  
VERSION >>  
    HTTP/1.1  
    (상관 없음)  
METHOD >>  
    서버로 전송할 메시지  
    (POST 인 경우에만 가능)
```

HTTP Request / Response Message

Request

Header

[METHOD] [URL] [VERSION]
Host: HOST:PORT
Content-Type: text/plain
User-Agent: CLIENT_NAME
Connection: keep-alive

Response

Header

[VERSION] [STATUS_CODE]
Server: SERVER_NAME
Date: DATETIME
Content-Type: text/plain
Connection: keep-alive

Body

...
TEXT_MESSAGE
...

Request / Response Message의 Header 필드는 최대한 간단하게 구성

Postman을 활용한 요청에서 Socket.close()을 하지 않으면 다음 요청을 처리하지 못하는 오류가 있어 **매번 연결-처리-해제** 하도록 코드 작성

소스 파일 설명

client.py

```
from socket import *

HOST, PORT = "127.0.0.1", 80      # 호스트 IP, PORT 설정
USER_AGENT = "..."             # 클라이언트 AGENT 이름

while True:
    # TCP Socket 생성
    CLIENT_SOCKET = socket(AF_INET, SOCK_STREAM)
    # TCP Connection
    CLIENT_SOCKET.connect((HOST, PORT))

    ...

    # HTTP Request Message 송신
    CLIENT_SOCKET.sendall(REQUEST.encode())

    # HTTP Response Message 수신
    RAW_RESPONSE = CLIENT_SOCKET.recv(65535)
    print(RAW_RESPONSE.decode())

    CLIENT_SOCKET.close() # TCP 연결 해제
```

server.py

```
from socket import *
from datetime import datetime

MESSAGE = []                      # 메시지 저장 변수
HOST, PORT = "127.0.0.1", 80      # 호스트 IP, PORT 설정
SERVER_NAME = "..."             # 서버 AGENT 이름

# TCP Socket 생성/바인딩/TCP_CONNECTION 설정
SERVER_SOCKET = socket(AF_INET, SOCK_STREAM)
SERVER_SOCKET.bind((HOST, PORT))
SERVER_SOCKET.listen(0)

while True:
    # Client 연결 대기
    CLIENT_SOCKET, ADDR = SERVER_SOCKET.accept()

    # HTTP Request Message 수신
    RAW_REQUEST = CLIENT_SOCKET.recv(65535)

    ...METHOD PROCESSING...

    # HTTP Response Message 송신
    CLIENT_SOCKET.sendall(RESPONSE.encode())

    CLIENT_SOCKET.close()          # TCP 연결 해제 (Client)
    SERVER_SOCKET.close()          # TCP 소켓 해제 (Server)
```

(1) GET Method - 200 OK

정상적으로 GET을 요청한 경우

✓ URL == /

(요청 URL이 루트)

서버 메모리에 저장 중인 메시지를 읽어옵니다.

클라이언트 사이드 프로그램

```
C:\Users\kimc9\Desktop\Github\ComputerNetwork\src>python client.py
METHOD >> GET
URL >> /
HTTP VERSION >> HTTP/1.1
HTTP/1.1 200 OK
Server: Python Socket Program Server
Date: Mon, 02 May 2022 17:33:51 KST
Content-Type: text/plain
Connection: keep-alive

HTTP GET Method (Success)
```

서버 사이드 프로그램

```
C:\Users\kimc9\Desktop\Github\ComputerNetwork\src>python server.py
GET / HTTP/1.1
Host: 127.0.0.1:80
Content-Type: text/plain
User-Agent: Python Socket Program Client
Connection: keep-alive
```

(2) GET Method - 404 Not Found

GET을 잘못 요청한 경우

- ✓ URL == /wrong/
(요청 URL이 잘못된 경우)

HTTP GET Method (Failed) 메시지 반환

클라이언트 사이드 프로그램

```
METHOD >> GET
URL >> /wrong/
HTTP VERSION >> HTTP/1.1
HTTP/1.1 404 Not Found
Server: Python Socket Program Server
Date: Mon, 02 May 2022 17:49:46 KST
Content-Type: text/plain
Connection: keep-alive

HTTP GET Method (Failed)
```

서버 사이드 프로그램

```
GET /wrong/ HTTP/1.1
Host: 127.0.0.1:80
Content-Type: text/plain
User-Agent: Python Socket Program Client
Connection: keep-alive
```

(3) POST Method - 201 Created

정상적으로 POST를 요청한 경우

✓ URL == /

(요청 URL이 루트)

MESSAGE = ...

(서버 메모리에 저장할 메시지)

서버 메모리에 메시지를 작성(추가)합니다.

클라이언트 사이드 프로그램

```
METHOD >> POST
URL >> /
HTTP VERSION >> HTTP/1.1
MESSAGE >> Computer Network
HTTP/1.1 201 Created
Server: Python Socket Program Server
Date: Mon, 02 May 2022 17:56:18 KST
Content-Type: text/plain
Connection: keep-alive

HTTP POST Method (Success)

Computer Network
```

서버 사이드 프로그램

```
POST / HTTP/1.1
Host: 127.0.0.1:80
Content-Type: text/plain
User-Agent: Python Socket Program Client
Connection: keep-alive

Computer Network
```


(4) POST Method - 404 Not Found

POST를 잘못 요청한 경우

- ✓ URL == /wrong/
(요청 URL이 잘못된 경우)

HTTP POST Method (Failed) 메시지 반환

클라이언트 사이드 프로그램

```
METHOD >> POST
URL >> /wrong/
HTTP VERSION >> HTTP/1.1
MESSAGE >> Computer Network
HTTP/1.1 404 Not Found
Server: Python Socket Program Server
Date: Mon, 02 May 2022 18:01:55 KST
Content-Type: text/plain
Connection: keep-alive

HTTP POST Method (Failed)
```

서버 사이드 프로그램

```
POST /wrong/ HTTP/1.1
Host: 127.0.0.1:80
Content-Type: text/plain
User-Agent: Python Socket Program Client
Connection: keep-alive
```

(5) DELETE Method - 200 OK

정상적으로 DELETE를 요청한 경우

✓ 조건 없음

(HOST:PORT로 DELETE 오면 무조건 수행)

서버 메모리에 저장 중인 메시지를 비웁니다.

클라이언트 사이드 프로그램

```
METHOD >> DELETE
URL >> /
HTTP VERSION >> HTTP/1.1
HTTP/1.1 200 OK
Server: Python Socket Program Server
Date: Mon, 02 May 2022 18:09:16 KST
Content-Type: text/plain
Connection: keep-alive

HTTP DELETE Method (Success)
```

서버 사이드 프로그램

```
DELETE / HTTP/1.1
Host: 127.0.0.1:80
Content-Type: text/plain
User-Agent: Python Socket Program Client
Connection: keep-alive
```

(6) PUT Method - 405 Method Not Allowed

허가되지 않은 요청을 한 경우

✓ METHOD == PUT

(메소드가 잘못된 경우)

HTTP [메소드] Method (Failed) 메시지 반환
+ GET/POST/DELETE 외 전부

클라이언트 사이드 프로그램

```
METHOD >> PUT
URL >> /
HTTP VERSION >> HTTP/1.1
HTTP/1.1 405 Method Not Allowed
Server: Python Socket Program Server
Date: Mon, 02 May 2022 18:13:54 KST
Content-Type: text/plain
Connection: keep-alive

HTTP PUT Method (Failed)
```

서버 사이드 프로그램

```
PUT / HTTP/1.1
Host: 127.0.0.1:80
Content-Type: text/plain
User-Agent: Python Socket Program Client
Connection: keep-alive
```

(ADD) Postman으로 서버에 요청

Postman

Response Headers

KEY		VALUE
Server	③	Python Socket Program Server
Date	③	Mon, 02 May 2022 18:19:08 KST
Content-Type	③	text/plain
Connection	③	keep-alive

Response Body

```
1 HTTP GET Method (Success)
2
3
```

서버 사이드 프로그램

```
C:\Users\kimc9\Desktop\Github\ComputerNetwork\src>python server.py
GET / HTTP/1.1
User-Agent: PostmanRuntime/7.29.0
Cache-Control: no-cache
Postman-Token: 21d2472e-624a-4386-8f45-ac9bb475d846
Host: 127.0.0.1:80
Connection: keep-alive
```

(ADD) Wireshark 캡처 (1)

*Adapter for loopback traffic capture

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

No.	Time	Source	Destination	Protocol	Length	Info
12	15.289337	127.0.0.1	127.0.0.1	HTTP	166	GET /
41	29.585090	127.0.0.1	127.0.0.1	HTTP	174	GET / HTTP/1.1
45	29.588068	127.0.0.1	127.0.0.1	HTTP	44	HTTP/1.1 200 OK (text/plain)

> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
v Transmission Control Protocol, Src Port: 63061, Dst Port: 80, Seq: 1, Ack: 1, Len: 130
Source Port: 63061
Destination Port: 80
[Stream index: 3]
[Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 130]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 1115264982
[Next Sequence Number: 131 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 802183373
0101 = Header Length: 20 bytes (5)
> Flags: 0x018 (PSH, ACK)
Window: 8442
[Calculated window size: 2161152]
[Window size scaling factor: 256]
Checksum: 0x12a0 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> [Timestamps]
> [SEQ/ACK analysis]
TCP payload (130 bytes)
v Hypertext Transfer Protocol
> GET / HTTP/1.1\r\n
Host: 127.0.0.1:80\r\n
Content-Type: text/plain\r\n
User-Agent: Python Socket Program Client\r\n
Connection: keep-alive\r\n
\r\n
[Full request URI: http://127.0.0.1:80/]
[HTTP request 1/1]
[Response in frame: 45]

Hypertext Transfer Protocol: Protocol || Packets: 61 · Displayed: 3 (4,9%) || Profile: Default

*Adapter for loopback traffic capture

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

No.	Time	Source	Destination	Protocol	Length	Info
12	15.289337	127.0.0.1	127.0.0.1	HTTP	166	GET /
41	29.585090	127.0.0.1	127.0.0.1	HTTP	174	GET / HTTP/1.1
45	29.588068	127.0.0.1	127.0.0.1	HTTP	44	HTTP/1.1 200 OK (text/plain)

> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
v Transmission Control Protocol, Src Port: 80, Dst Port: 63061, Seq: 174, Ack: 131, Len: 0
Source Port: 80
Destination Port: 63061
[Stream index: 3]
[Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 0]
Sequence Number: 174 (relative sequence number)
Sequence Number (raw): 802183546
[Next Sequence Number: 175 (relative sequence number)]
Acknowledgment Number: 131 (relative ack number)
Acknowledgment number (raw): 1115265112
0101 = Header Length: 20 bytes (5)
> Flags: 0x011 (FIN, ACK)
Window: 8442
[Calculated window size: 2161152]
[Window size scaling factor: 256]
Checksum: 0x3615 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> [Timestamps]
> [2 Reassembled TCP Segments (173 bytes): #43(173), #45(0)]
v Hypertext Transfer Protocol
> HTTP/1.1 200 OK\r\n
Server: Python Socket Program Server\r\n
Date: Mon, 02 May 2022 18:44:47 KST\r\n
Content-Type: text/plain\r\n
Connection: keep-alive\r\n
\r\n
[HTTP response 1/1]
[Time since request: 0.002978000 seconds]
[Request in frame: 41]
[Request URI: http://127.0.0.1:80/]

frame (44 bytes) | reassembled TCP (173 bytes)
Hypertext Transfer Protocol: Protocol || Packets: 69 · Displayed: 3 (4,3%) || Profile: Default

(ADD) Wireshark 캡처 (2)

*Adapter for loopback traffic capture

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

No.	Time	Source	Destination	Protocol	Length	Info
45	29.588068	127.0.0.1	127.0.0.1	HTTP	44	HTTP/1.1 200 OK (text/plain)
96	222.566057	127.0.0.1	127.0.0.1	HTTP	177	DELETE / HTTP/1.1
100	222.567776	127.0.0.1	127.0.0.1	HTTP	44	HTTP/1.1 200 OK (text/plain)

> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
v Transmission Control Protocol, Src Port: 63062, Dst Port: 80, Seq: 1, Ack: 1, Len: 133
Source Port: 63062
Destination Port: 80
[Stream index: 4]
[Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 133]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 476681696
[Next Sequence Number: 134 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 3645840969
0101 = Header Length: 20 bytes (5)
> Flags: 0x018 (PSH, ACK)
Window: 8442
[Calculated window size: 2161152]
[Window size scaling factor: 256]
Checksum: 0x3569 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> [Timestamps]
> [SEQ/ACK analysis]
TCP payload (133 bytes)
v Hypertext Transfer Protocol
> DELETE / HTTP/1.1\r\n
Host: 127.0.0.1:80\r\n
Content-Type: text/plain\r\n
User-Agent: Python Socket Program Client\r\n
Connection: keep-alive\r\n
\r\n
[Full request URI: http://127.0.0.1:80/]
[HTTP request 1/1]
[Response in frame: 100]

Hypertext Transfer Protocol (http), 133 byte(s) | Packets: 106 · Displayed: 5 (4,7%) | Profile: Default

*Adapter for loopback traffic capture

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

No.	Time	Source	Destination	Protocol	Length	Info
45	29.588068	127.0.0.1	127.0.0.1	HTTP	44	HTTP/1.1 200 OK (text/plain)
96	222.566057	127.0.0.1	127.0.0.1	HTTP	177	DELETE / HTTP/1.1
100	222.567776	127.0.0.1	127.0.0.1	HTTP	44	HTTP/1.1 200 OK (text/plain)

> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
v Transmission Control Protocol, Src Port: 80, Dst Port: 63062, Seq: 175, Ack: 134, Len: 0
Source Port: 80
Destination Port: 63062
[Stream index: 4]
[Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 0]
Sequence Number: 175 (relative sequence number)
Sequence Number (raw): 3645841143
[Next Sequence Number: 176 (relative sequence number)]
Acknowledgment Number: 134 (relative ack number)
Acknowledgment number (raw): 476681829
0101 = Header Length: 20 bytes (5)
> Flags: 0x011 (FIN, ACK)
Window: 8442
[Calculated window size: 2161152]
[Window size scaling factor: 256]
Checksum: 0xef1a [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> [Timestamps]
> [2 Reassembled TCP Segments (174 bytes): #98(174), #100(0)]
v Hypertext Transfer Protocol
> HTTP/1.1 200 OK\r\n
Server: Python Socket Program Server\r\n
Date: Mon, 02 May 2022 18:48:00 KST\r\n
Content-Type: text/plain\r\n
Connection: keep-alive\r\n
\r\n
[HTTP response 1/1]
[Time since request: 0.001719000 seconds]
[Request in frame: 96]
[Request URI: http://127.0.0.1:80/]

frame (44 bytes) | reassembled TCP (174 bytes) | Packets: 122 · Displayed: 5 (4,1%) | Profile: Default