

Lab 1: Web Server Lab

server.py

I hard-coded the port as 12345, the host is blank for localhost.

```
from socket import *
import sys

serverSocket = socket(AF_INET, SOCK_STREAM)

#Prepare a server socket
#Fill in start
serverPort = 12345
serverSocket.bind(('', serverPort))
serverSocket.listen(1)
#Fill in end

while True:
    #Establish the connection
    print('Ready to serve...')
    connectionSocket, addr = serverSocket.accept() #Fill in start      #Fill in end
    try:
        message = connectionSocket.recv(1024)
        print(message)

        filename = message.split()[1]
        f = open(filename[1:])
        outputdata = f.read()

        #Send one HTTP header line into socket
        print("Success!\n\n")
        connectionSocket.send("\nHTTP/1.x 200 OK\n")

        #Send the content of the requested file to the client
        for i in range(0, len(outputdata)):
            connectionSocket.send(outputdata[i].encode())

        connectionSocket.send("\r\n".encode())
        connectionSocket.close()
    except IOError:
        #Send response message for file not found (404)
        print("Failure: file not found!\n\n")
        connectionSocket.send("\n404 File Not Found\n")
        connectionSocket.send("\r\n".encode())

        #Close client socket
        connectionSocket.close()

serverSocket.close()
sys.exit() #Terminate the program after sending the corresponding data
```

```
python server.py
```

First I put <http://192.168.1.14:12345/helloworld.html> in the browser, which received a 200 response, but a 404 for /favicon.ico because, well, we don't have a favicon file.

I then put <http://192.168.1.14:12345/potato.html>, which sent out 3 GET requests and received 3 404 responses. (Only the first fit in the screenshot.)

```
Documents/NYU Bridge/computer_networking_sp20
$ python server.py
Ready to serve...
GET /helloworld.html HTTP/1.1
Host: 192.168.1.14:12345
Connection: keep-alive
DNT: 1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.100 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

Success!

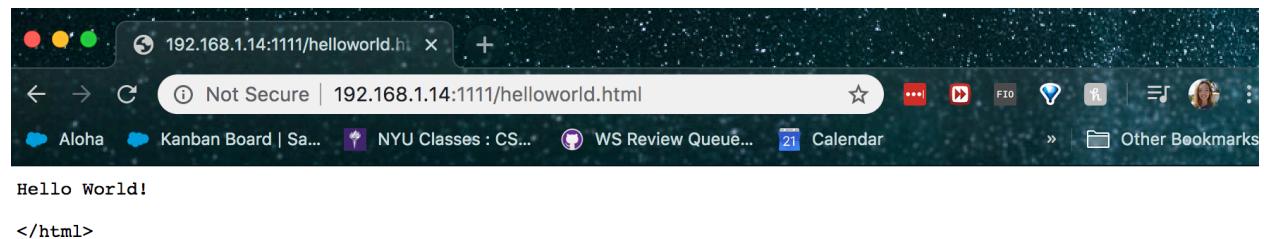
Ready to serve...
GET /favicon.ico HTTP/1.1
Host: 192.168.1.14:12345
Connection: keep-alive
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.100 Safari/537.36
DNT: 1
Accept: image/webp,image/apng,image/*/*;q=0.8
Referer: http://192.168.1.14:12345/helloworld.html
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

Failure: file not found!

Ready to serve...
GET /potato.html HTTP/1.1
Host: 192.168.1.14:12345
Connection: keep-alive
DNT: 1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.100 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

Failure: file not found!
```

Browser – successful GET Request



The server doesn't send back any data in the event of a bad request, so you receive the browser's default "This page isn't working" when the GET request response code is not 200.

Optional Exercises 1

multithread_server.py – Again I hard-coded the port as 1111, & the host is default blank for localhost.

```
1  from socket import *
2  import threading
3  import sys
4
5  class ThreadedServer(object):
6      def __init__(self):
7          self.host = ''
8          self.port = 1111
9          self.serverSocket = socket(AF_INET, SOCK_STREAM)
10
11     def run(self):
12         print("Ready to serve...")
13
14         self.serverSocket.bind((self.host, self.port))
15         self.serverSocket.listen(10)
16
17     while True:
18         clientSocket, address = self.serverSocket.accept()
19         clientSocket.settimeout(60)
20
21         threading.Thread(
22             target = self.new_thread,
23             args = (clientSocket, address)
24         ).start()
25
26     serverSocket.close()
27
28     def new_thread(self, clientSocket, address):
29         while True:
30             try:
31                 request = clientSocket.recv(1024)
32                 print(request)
33
34                 filename = request.split()[1]
35                 f = open(filename[1:])
36                 outputdata = f.read()
37                 clientSocket.send("\nHTTP/1.x 200 OK\n")
38
39                 for i in range(0, len(outputdata)):
40                     clientSocket.send(outputdata[i].encode())
41
42                 clientSocket.send("\r\n".encode())
43                 clientSocket.close()
44             except(error):
45                 print(error)
46                 clientSocket.send("\n400 Bad Request\n")
47                 clientSocket.close()
48                 sys.exit()
49
50     ThreadedServer().run()
```

```
python multithread_server.py
```

The browser page for calling <http://192.168.1.14:12345/helloworld.html> and <http://192.168.1.14:12345/potato.html> are the same as those of *server.py*.

As you can see below, there are multiple threads, with Exceptions thrown when the page does not exist. I opted to catch all errors and send a generic 400 response rather catch the specific IOError & throw a 404, even though in this lab we only encounter the 404 error to start off more generic; if I were writing a real program, I'd have an exception case for every common response code.

```
python multithread_server.py
Ready to serve...
GET /helloworld.html HTTP/1.1
Host: 192.168.1.14:1111
Connection: keep-alive
DNT: 1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.100 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

<class 'socket.error'>
Exception in thread Thread-1:
Traceback (most recent call last):
  File "/usr/local/Cellar/python@2/2.7.15_1/Frameworks/Python.framework/Versions/2.7/lib/python2.7/threading.py", line 801, in __bootstrap_inner
    self._run()
  File "/usr/local/Cellar/python@2/2.7.15_1/Frameworks/Python.framework/Versions/2.7/lib/python2.7/threading.py", line 754, in run
    self._target(*self._args, **self._kwargs)
  File "multithread_server.py", line 46, in new_thread
    clientSocket.send("\n400 Bad Request\n")
  File "/usr/local/Cellar/python@2/2.7.15_1/Frameworks/Python.framework/Versions/2.7/lib/python2.7/socket.py", line 174, in _dummy
    raise error(EBADF, 'Bad file descriptor')
error: [Errno 9] Bad file descriptor

GET /favicon.ico HTTP/1.1
Host: 192.168.1.14:1111
Connection: keep-alive
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.100 Safari/537.36
DNT: 1
Accept: image/webp,image/apng,image/*,*/*;q=0.8
Referer: http://192.168.1.14:1111/helloworld.html
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

Exception in thread Thread-2:
Traceback (most recent call last):
  File "/usr/local/Cellar/python@2/2.7.15_1/Frameworks/Python.framework/Versions/2.7/lib/python2.7/threading.py", line 801, in __bootstrap_inner
    self._run()
  File "/usr/local/Cellar/python@2/2.7.15_1/Frameworks/Python.framework/Versions/2.7/lib/python2.7/threading.py", line 754, in run
    self._target(*self._args, **self._kwargs)
  File "multithread_server.py", line 35, in new_thread
    f = open(filename[1:])
IOError: [Errno 2] No such file or directory: 'favicon.ico'
```

More threads error-ing out to show that multi-threading is working...

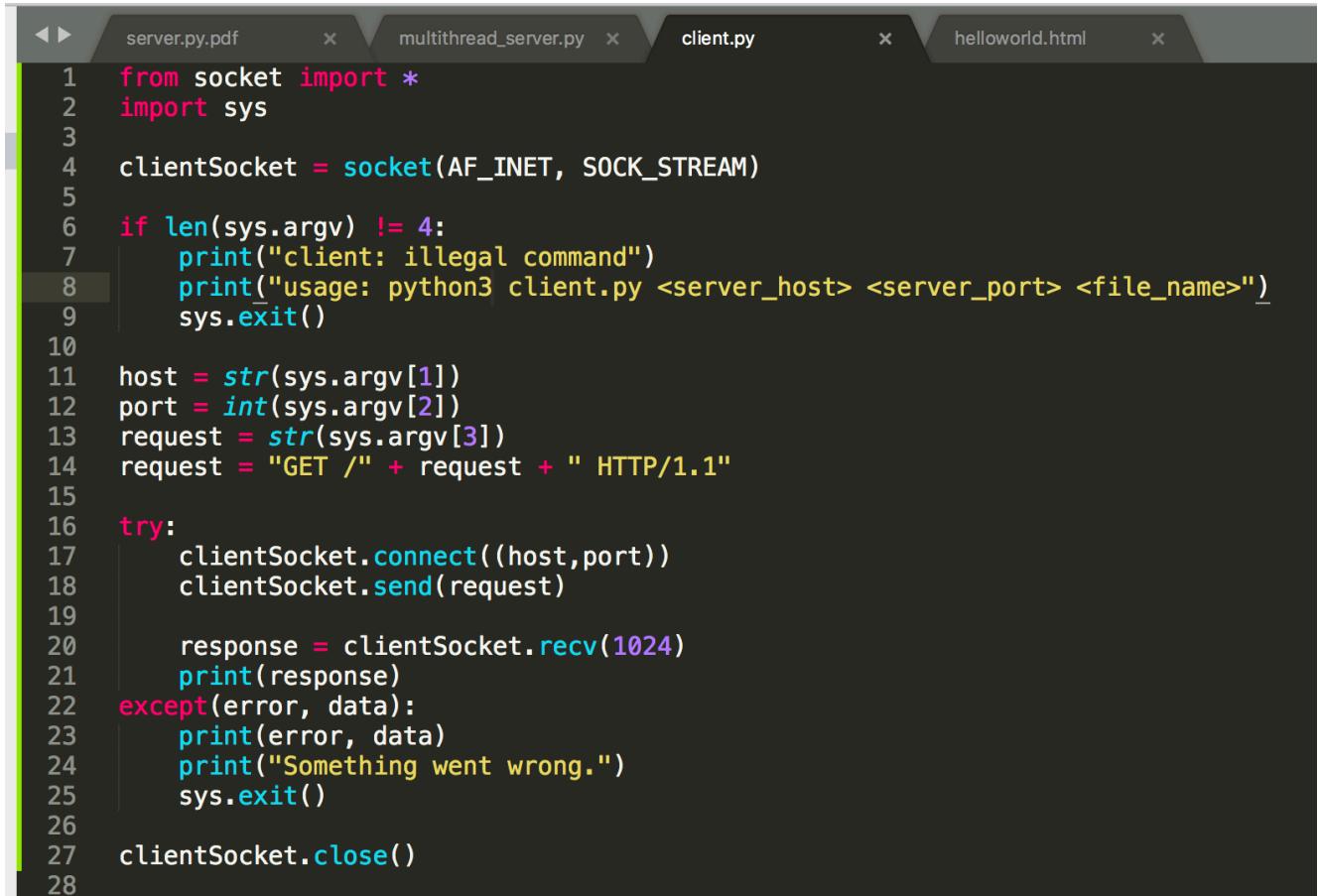
```
Exception in thread Thread-6:
Traceback (most recent call last):
  File "/usr/local/Cellar/python@2/2.7.15_1/Frameworks/Python.framework/Versions/2.7/lib/python2.7/threading.py", line 801, in __bootstrap_inner
    self._run()
  File "/usr/local/Cellar/python@2/2.7.15_1/Frameworks/Python.framework/Versions/2.7/lib/python2.7/threading.py", line 754, in run
    self._target(*self._args, **self._kwargs)
  File "multithread_server.py", line 35, in new_thread
    f = open(filename[1:])
IOError: [Errno 2] No such file or directory: 'potato.html'

GET /potato.html HTTP/1.1
Host: 192.168.1.14:1111
Connection: keep-alive
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
DNT: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.100 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

Exception in thread Thread-7:
Traceback (most recent call last):
  File "/usr/local/Cellar/python@2/2.7.15_1/Frameworks/Python.framework/Versions/2.7/lib/python2.7/threading.py", line 801, in __bootstrap_inner
    self._run()
  File "/usr/local/Cellar/python@2/2.7.15_1/Frameworks/Python.framework/Versions/2.7/lib/python2.7/threading.py", line 754, in run
    self._target(*self._args, **self._kwargs)
  File "multithread_server.py", line 35, in new_thread
    f = open(filename[1:])
IOError: [Errno 2] No such file or directory: 'potato.html'
```

Optional Exercise 2

client.py



A screenshot of a code editor showing the *client.py* script. The editor has tabs for *server.py.pdf*, *multithread_server.py*, *client.py*, and *helloworld.html*. The *client.py* tab is active. The code is a Python script that connects to a server at specified host and port, sends an HTTP GET request for a specified file, and prints the response. It includes error handling for command-line arguments and network errors.

```
1 from socket import *
2 import sys
3
4 clientSocket = socket(AF_INET, SOCK_STREAM)
5
6 if len(sys.argv) != 4:
7     print("client: illegal command")
8     print("usage: python3 client.py <server_host> <server_port> <file_name>")
9     sys.exit()
10
11 host = str(sys.argv[1])
12 port = int(sys.argv[2])
13 request = str(sys.argv[3])
14 request = "GET /" + request + " HTTP/1.1"
15
16 try:
17     clientSocket.connect((host, port))
18     clientSocket.send(request)
19
20     response = clientSocket.recv(1024)
21     print(response)
22 except(error, data):
23     print(error, data)
24     print("Something went wrong.")
25     sys.exit()
26
27 clientSocket.close()
28
```

A simple client that gets the *host*, *port*, and *file_name* from the CLI and makes requests to the server. It doesn't automatically retry requests upon 400 responses. In the event of an error, the client closes the connection with an opaque error message.

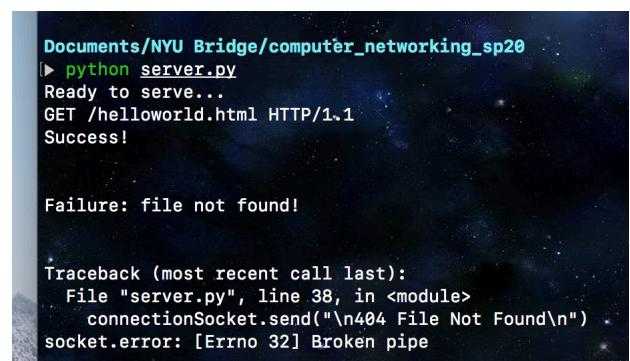
As you can see, GET /helloworld.html is a success, but the server shuts down due to the 404 error from the favicon, causing the GET /potato.html to get a socket error.



A terminal session showing the execution of *client.py*. The user runs *python client.py 192.168.1.14 12345 helloworld.html*. The response is "HTTP/1.x 200 OK". Then, the user runs *python client.py 192.168.1.14 12345 potato.html*, which results in a socket error: "Something went wrong.".

```
Documents/NYU Bridge/computer_networking_sp20
▶ python client.py 192.168.1.14 12345 helloworld.html
HTTP/1.x 200 OK

Documents/NYU Bridge/computer_networking_sp20
▶ python client.py 192.168.1.14 12345 potato.html
<class 'socket.error'> [Errno 61] Connection refused
Something went wrong.
```



A terminal session showing the execution of *server.py*. The user runs *python server.py*. The server responds with "Ready to serve...". When the user runs *python client.py 192.168.1.14 12345 helloworld.html*, it prints "Success!". However, when the user runs *python client.py 192.168.1.14 12345 potato.html*, it prints "Failure: file not found!" and shows a traceback indicating a Broken pipe error.

```
Documents/NYU Bridge/computer_networking_sp20
▶ python server.py
Ready to serve...
GET /helloworld.html HTTP/1.1
Success!

Failure: file not found!

Traceback (most recent call last):
  File "server.py", line 38, in <module>
    connectionSocket.send("\n404 File Not Found\n")
socket.error: [Errno 32] Broken pipe
```

Let's fire it up again...

```
Documents/NYU Bridge/computer_networking_sp20
▶ python server.py
Ready to serve...
GET /potato.html HTTP/1.1
Failure: file not found!

Ready to serve...
```

```
Documents/NYU Bridge/computer_networking_sp20
▶ python client.py 192.168.1.14 12345 potato.html
404 File Not Found
```

This time the server returns a 404 response and continues waiting for requests. The client program exits.