Computer Networking - Socket Programming Lab - 1

\*\*Identifying Lab Environment\*\*

The below image depicts my lab environment on my laptop, it uses MacOS.

A screenshot of a computer

Description automatically generated

Server.py File

Code with comment lines for explanation

from socket import \*

import sys

def webServer(port):

serverPort = port

serverSocket = socket(AF\_INET, SOCK\_STREAM)#server socket - TCP byte stream

serverSocket.bind(('',serverPort)) #binding port to server port

serverSocket.listen(1) #server begins listening for incoming requests

print('Server being set up...')

while True:

print ('Server is ready !')

connectionSocket, addr = serverSocket.accept() #server waits on accept new socket returned on request as this is TCP

print('Request accepted from:', addr)

try:

message = connectionSocket.recv(1024).decode() #decodes the received byte stream message

filename = message.split()[1]

f = open(filename[1:],'r')

outputdata = f.read()

headerLine = 'HTTP/1.1 200 OK\r\n' #header line containing response status code and status phrase

connectionSocket.send(headerLine.encode()) #encoding to send back

connectionSocket.send('\r\n'.encode()) #carriage return otherwise the headers won't be displayed

for i in range(0, len(outputdata)):

connectionSocket.send(outputdata[i].encode())

connectionSocket.send('\r\n'.encode())

connectionSocket.close()

except IOError:

errHeaderLine = 'HTTP/1.1 404 Not Found\r\n' #r for carriage return and n for new line

connectionSocket.send(errHeaderLine.encode())

connectionSocket.send('\r\n'.encode())

f2 = open('404page.html','r')

outputdata\_err = f2.read()

for i in range(0, len(outputdata\_err)):

connectionSocket.send(outputdata\_err[i].encode())

connectionSocket.send('\r\n'.encode())

connectionSocket.close()

serverSocket.close()

sys.exit()

if \_\_name\_\_ == '\_\_main\_\_':

webServer(6879)

Output:

helloworld.html 200

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

404page.html 404

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

**Optional Exercises**

1. Multi-thread.py

1. from socket import \*
2. import sys
3. import threading
4. def handle\_client(connectionSocket):
5. try:
6. message = connectionSocket.recv(1024).decode()
7. filename = message.split()[1]
8. f = open(filename[1:], 'r')
9. outputdata = f.read()
10. headerLine = 'HTTP/1.1 200 OK\r\n'
11. connectionSocket.send(headerLine.encode())
12. connectionSocket.send('\r\n'.encode())
13. for char in outputdata:
14. connectionSocket.send(char.encode())
15. connectionSocket.send('\r\n'.encode())
16. connectionSocket.close()
18. except IOError:
19. errHeaderLine = 'HTTP/1.1 404 Not Found\r\n'
20. connectionSocket.send(errHeaderLine.encode())
21. connectionSocket.send('\r\n'.encode())
22. f2 = open('404page.html', 'r')
23. outputdata\_err = f2.read()
24. for char in outputdata\_err:
25. connectionSocket.send(char.encode())
26. connectionSocket.send('\r\n'.encode())
27. connectionSocket.close()
28. def webServer(port):
29. serverPort = port
30. serverSocket = socket(AF\_INET, SOCK\_STREAM)
31. serverSocket.bind(('', serverPort))
32. serverSocket.listen(4) #this is for multiple connections - 4 connections
33. print('Server being set up...')
34. while True:
35. print('Server is ready!')
36. connectionSocket, addr = serverSocket.accept()
37. print('Request accepted from:', addr)
39. client\_thread = threading.Thread(target=handle\_client, args=(connectionSocket,))#Creating a new thread for each connection
40. client\_thread.start()
41. if \_\_name\_\_ == '\_\_main\_\_':
42. webServer(6879)

Output:

The below image indicates multi tabs open for indicating sessions with the server.

A screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

2. Client.py

1. from socket import \*
2. import sys
3. if (len(sys.argv) != 4): #For arguments to the script so that client can connect to server
4. print('Incorrect number of arguments.')
5. print('Help: To run python client.py <server\_host> <server\_port> <filename>')
6. sys.exit()
7. serverHost, serverPort, filename = sys.argv[1:] #server arguments
8. clientSocket = socket(AF\_INET, SOCK\_STREAM)
9. try:
10. clientSocket.connect((serverHost, int(serverPort))) #client connecting to server
11. except:
12. print('The server is currently inactive')
13. clientSocket.close() #close connection if inactive
14. sys.exit()
15. print('Connection OK.')
16. httpRequest = 'GET /' + filename + ' HTTP/1.1\r\n\r\n' #HTTP Request
17. clientSocket.send(httpRequest.encode())
18. print('Request message sent.')
19. print('Server HTTP Response:\r\n')
20. data = ""
21. while True:
22. clientSocket.settimeout(5)
23. newData = clientSocket.recv(1024).decode()
24. data += newData
25. if (len(newData) == 0):
26. break
27. print(data)
28. print('Closing socket') #Closing socket
29. clientSocket.close()

Output: A screenshot of a computer

Description automatically generated

Please refer to the code from my Github Repo Link : <https://github.com/KrapaKarthik09/socket-programming/tree/Prog-1>