Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Факультет инфокоммуникационных технологий

Лабораторная работа №1

по дисциплине:

«Web-программирование»

Выполнил:
студентка III курса ИКТ
группы K33422
Ф.И.О Плотская Дарья
Александровна

Санкт-Петербург
2021

Работа с сокетами

Цель: овладеть практическими навыками и умениями реализации web-серверов и использования сокетов.

Практическое задание:

1. Реализовать клиентскую и серверную часть приложения. Клиент отсылает серверу сообщение «Hello, server». Сообщение должно отразиться на стороне сервера. Сервер в ответ отсылает клиенту сообщение «Hello, client». Сообщение должно отобразиться у клиента.

```python
import socket

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((socket.gethostname(), 14900))
s.listen(5)



while True:
    clientsocket, address = s.accept()
    clientsocket.send(bytes('Hello, client', 'utf-8'))
    data = clientsocket.recv(1024)
    print(data.decode("utf-8"))


s.close()
```

```python
import socket

s = socket.socket (socket.AF_INET, socket.SOCK_STREAM)
s.connect((socket.gethostname(), 14900))
msg = s.recv(1024)
s.send(bytes("Hello, server", "utf-8"))

print(msg.decode('utf-8'))
```

```
C:\Users\plots\Desktop\py\web>py -3.7 server.py
Hello, server
```

```
C:\Users\plots\Desktop\py\web>py -3.7 client.py
Hello, client
```

2. Реализовать клиентскую и серверную часть приложения. Клиент запрашивает у сервера выполнение математической операции, параметры, которые вводятся с клавиатуры. Сервер обрабатывает полученные данные и возвращает результат клиенту.

   Я искала площадь трапеции:

```python
import socket

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((socket.gethostname(), 14900))
s.listen(5)

while True:
    clientsocket, address = s.accept()
    data = clientsocket.recv(1024)
    numbers = data.decode("utf-8")
    a = int(numbers[0])
    b = int(numbers[1])
    h = int(numbers[2])
    sq = (a+b)/2*h
    clientsocket.send(bytes(str(sq), "utf-8"))
s.close()
```

```python
import socket

s = socket.socket (socket.AF_INET, socket.SOCK_STREAM)
s.connect((socket.gethostname(), 14900))
a = int(input('Введите длину верхнего основания трапеции: '))
b = int(input('Введите длину нижнего основания трапеции: '))
h = int(input('Введите длину высоты трапеции: '))
q = f'{a}{b}{h}'

s.send(bytes(q, 'utf-8'))
data = s.recv(1024)
print(f'Ваш ответ: {data.decode("utf-8")}')
```

```
C:\Users\plots\Desktop\py\web\lab1>py -3.7 client_result.py
Введите длину верхнего основания трапеции: 3
Введите длину нижнего основания трапеции: 4
Введите длину высоты трапеции: 5
Ваш ответ: 17.5
```

3. Необходимо написать простой web-сервер для обработки GET и POST http запросов средствами Python и библиотеки socket.

```python
import socket
import sys

MAX_LINE = 64*1024

class MyHTTPServer:
    '''Параметры сервера'''

    def __init__(self, host, port, name):
        self.host = host
        self.port = port
        self.name = name

    def serve_forever(self):
    '''1. Запуск сервера на сокете,
    # обработка входящих соединений'''

        serv_sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM, proto=0)

        try:
            serv_sock.bind((self.host, self.port))
            serv_sock.listen(1)

            while True:
                conn, _ = serv_sock.accept()
                try:
                    self.serve_client(conn)
                except Exception as e:
                    print('Client serving failed', e)
        finally:
            serv_sock.close()
```

```python
    def serve_client(self, conn):
        '''2. Обработка клиентского подключения'''

        try:
            req = self.parse_request(conn)
            resp = self.handle_request(req)
            self.send_response(conn, resp)
        except ConnectionResetError:
            conn = None
        except Exception as e:
            self.send_error(conn, e)

        if conn:
            conn.close()
```

```python
def parse_request(self, conn):
    '''3. функция для обработки заголовка http+запроса.
    Python, сокет предоставляет возможность создать
    вокруг него некоторую обертку, которая предоставляет
    file object интерфейс.
    Это дайте возможность построчно обработать запрос.
    Заголовок всегда - первая строка. Первую строку
    нужно разбить на 3 элемента
    (метод + url + версия протокола).
    URL необходимо разбить на адрес и параметры
    (isu.ifmo.ru/pls/apex/f?p=2143 ,
    где isu.ifmo.ru/pls/apex/f,
    а p=2143 - параметр p со значением 2143)'''

    rfile = conn.makefile('rb')
    raw = rfile.readline(MAX_LINE + 1)
    if len(raw) > MAX_LINE:
        raise Exception('Request line is too long')

    req_line = str(raw, 'iso-8859-1')
    req_line = req_line.rstrip('\r\n')
    words = req_line.split()
    if len(words) != 3:
        raise Exception('Malformed request line')

    method, url, ver = words
    if ver != 'HTTP/1.1':
        raise Exception('Unexpected HTTP version')

    url, params = url.split("?")
    params = params.split("&")
    params_dct = {}
    for i in range(len(params)):
        p_list = params[i].split("=")
        params_dct[p_list[0]] = p_list[1]

    dct = {"method": method, "url": url,
           "params": params_dct, "ver": ver}
    return dct
```

```python
    def handle_request(self, req):
        '''5. Функция для обработки url в соответствии
        с нужным методом. В случае данной работы,
        нужно будет создать набор условий,
        который обрабатывает GET или POST запрос.
        GET запрос должен возвращать данные.
        POST запрос должен записывать данные на основе
        переданных параметров'''

        if req["method"] == "GET":
            with open("index.html", "rb") as f:
                data = f.read()
            headers = ['Content-Type: text/html\r\n',
                       f'Content-Length: {len(data)}\r\n']

            response = {"code": 200, "reason": "OK", "headers": headers, "body": data}
            return response

        if req["method"] == "POST":
            params = req["params"]
            data = ""
            for param in params:
                st = f'\n\t\t\t<tr>\n\
                \t<td>{param}</td>\n\
                \t<td>{params[param]}</td>\n\
                </tr>'
                data += st
            with open("index.html", "r") as f:
                text = f.read()
            if "</tr>" in text:
                ind = text.rindex("</tr>")+len("</tr>")
                new_text = text[:ind]
                new_text += data
                new_text += text[ind:]
            with open("index.html", "wb") as f:
                f.write(new_text.encode())

            response = {"code": 204, "reason": "Grade added"}

            return response
```

```python
def send_response(self, conn, response_data):
'''6. Функция для отправки ответа.
Необходимо записать в соединение status line вида
HTTP/1.1 <status_code> <reason>.
Затем, построчно записать заголовки и пустую строку,
обозначающую конец секции заголовков. '''

    wfile = conn.makefile('wb')
    status_line = f"HTTP/1.1 {response_data['code']} {response_data['reason']}\r\n"
    wfile.write(status_line.encode('iso-8859-1'))

    if "headers" in response_data:
        for header in response_data["headers"]:
            wfile.write(header.encode('iso-8859-1'))

    wfile.write(b'\r\n')

    if "body" in response_data:
        wfile.write(response_data["body"])

    wfile.flush()
    wfile.close()
```

```
    def send_error(self, conn, e):
    '''7. Функция для отправки ошибки'''

        print("send_error", e)
        try:
            body = e.encode('utf-8')
        except:
            status = 500
            reason = b'Internal Server Error'
            body = b'Internal Server Error'

            headers = [f'Content-Length: {len(body)}\r\n']
            resp = {"code": status, "reason": reason, "headers": headers, "body": body}

        self.send_response(conn, resp)


if __name__ == '__main__':
    host = "localhost"
    port = 9090
    name = "MyServer"

    serv = MyHTTPServer(host, port, name)
    try:
        serv.serve_forever()
    except KeyboardInterrupt:
        pass
```

```
import socket

sock = socket.socket()
sock.connect(('localhost', 9090))

GET_request = ["GET isu.ifmo.ru/pls/apex/f HTTP/1.1",
        "Host: example.local",
        "Accept: text/html",
        "User-Agent: Mozilla/5.0"]

POST_request = ["POST isu.ifmo.ru/pls/apex/f?Spanish=100 HTTP/1.1",
        "Host: example.local",
        "Accept: text/html",
        "User-Agent: Mozilla/5.0"]

sock.send("\r\n".join(POST_request).encode())

print("Отправили оценочки...")
data = sock.recv(1024)
sock.close()

print(data.decode())
```

```
C:\Users\plots\Desktop\py\web\lab1>py -3.7 3_client.py
Отправили оценочки...
HTTP/1.1 204 Grade added
```

Grades

Dicipline	Grade
Physics	100
English	89
History	87
Spanish	100

4. Реализовать двухпользовательский или многопользовательский чат.

Реализовала многопользовательский:

```
import socket, threading

server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
host = '127.0.0.2'
port = 6000
server.bind((host, port))
server.listen()

clients = []
users = []


def broadcast(msg, client):
    for each in clients:
        if each != client:
            each.send(msg)


def handle(client):
    while True:
        msg = client.recv(2000)
        broadcast(msg, client)


def receive():
    while True:
        client, addr = server.accept()
        client.send('username'.encode())
        user = client.recv(2000).decode()
        clients.append(client)
        users.append(user)
        client.send('Connection established'.encode())
        thread = threading.Thread(target=handle, args=(client,))
        thread.start()


receive()
```

```python
import socket
import threading


conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server = '127.0.0.2', 6000
conn.connect(server)

username = input('Выберите псевдоним: ')


def recv_msg():
    while True:
        msg = conn.recv(2000).decode()
        if msg == 'username':
            conn.send(username.encode())
        else:
            print(msg)


def print_msg():
    while True:
        msg = '{} says: {}'.format(username, input(''))
        conn.send(msg.encode())


recv_thr = threading.Thread(target=recv_msg)
print_thr = threading.Thread(target=print_msg)
recv_thr.start()
print_thr.start()
```

```
C:\Users\plots\Desktop\py\web\lab1>py -3.7 4.py
Выберите псевдоним: olticher
Connection established
hello! I'm Dasha
masha says: i love dogs.....I'm Masha!!
pasha says: I'm allergic to pets((( my name is Pasha!
```

```
C:\Users\plots\Desktop\py\web\lab1>py -3.7 4.py
Выберите псевдоним: masha
Connection established
i love dogs.....I'm Masha!!
pasha says: I'm allergic to pets((( my name is Pasha!
```

```
C:\Users\plots\Desktop\py\web\lab1>py -3.7 4.py
Выберите псевдоним: pasha
Connection established
I'm allergic to pets((( my name is Pasha!
```

Вывод:

Я научилась работать с библиотекой sockets в python.