

Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики.

Факультет инфокоммуникационных технологий.

Практическая работа №1 по теме:
«Работа с сокетами» по дисциплине:
Web-программирование.

Выполнил: Кирильцев Роман
Группа: K33401

Преподаватель: Говоров Антон Игоревич

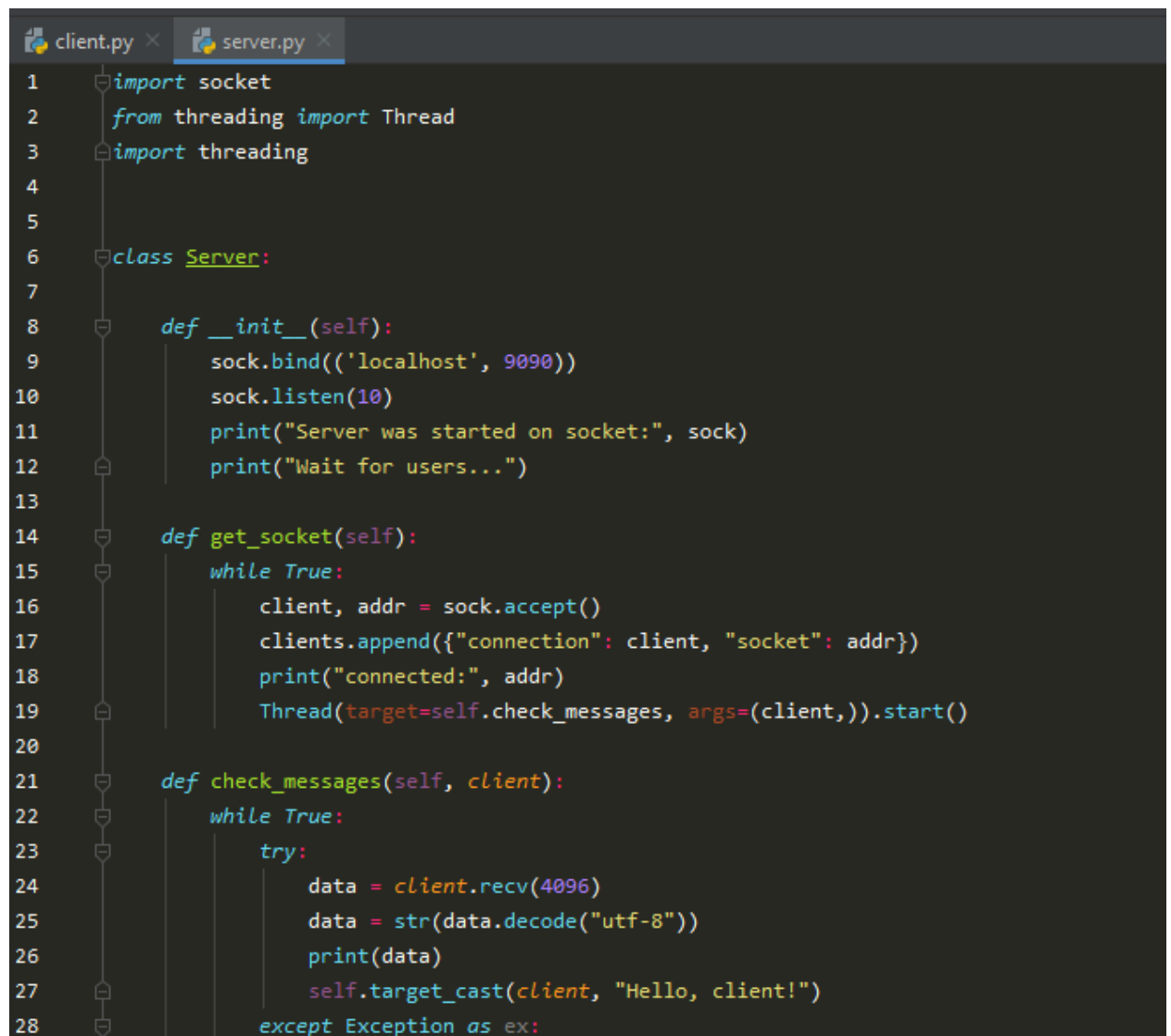
Санкт-Петербург, 2020

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

Практическое задание 1:

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

Код программы:



```
1  import socket
2  from threading import Thread
3  import threading
4
5
6  class Server:
7
8      def __init__(self):
9          sock.bind(('localhost', 9090))
10         sock.listen(10)
11         print("Server was started on socket:", sock)
12         print("Wait for users...")
13
14     def get_socket(self):
15         while True:
16             client, addr = sock.accept()
17             clients.append({"connection": client, "socket": addr})
18             print("connected:", addr)
19             Thread(target=self.check_messages, args=(client,)).start()
20
21     def check_messages(self, client):
22         while True:
23             try:
24                 data = client.recv(4096)
25                 data = str(data.decode("utf-8"))
26                 print(data)
27                 self.target_cast(client, "Hello, client!")
28             except Exception as ex:
```

```

28         except Exception as ex:
29             self.delete_user_socket(client)
30             break
31
32     def delete_user_socket(self, client):
33         for user in clients:
34             if user["connection"] == client:
35                 clients.remove(user)
36                 break
37
38     def target_cast(self, client, message):
39         message += "\n"
40         client.send(message.encode())
41
42
43 if __name__ == "__main__":
44     sock = socket.socket()
45     clients = []
46     server = Server()
47     thread = threading.Thread(target=server.get_socket())
48     thread.start()
49     thread.join()

```

```

client.py x server.py x
1 import socket
2
3 sock = socket.socket()
4 sock.connect(('localhost', 9090))
5 sock.send('Hello, server'.encode("UTF-8"))
6
7 data = sock.recv(1024)
8 sock.close()
9
10 print(data.decode("UTF-8"))

```

Результат работы программы:

```

terminal: Local x Local (2) x +
:~\Users\rk319\Documents\Study\untitled\lab1>py client.py
ello, client!

:~\Users\rk319\Documents\Study\untitled\lab1>

```

```
Terminal Local x Local (2) x +
(с) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\rk319\Documents\Study\untitled>cd lab1.py
Системе не удастся найти указанный путь.

C:\Users\rk319\Documents\Study\untitled>cd lab1

C:\Users\rk319\Documents\Study\untitled\lab1>py server.py
Server was started on socket: <socket.socket fd=524, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 9090)>
Wait for users...
connected: ('127.0.0.1', 54874)
Hello, server
```

Практическое задание 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.

Вариант: а) Теорема Пифагора

Код программы:

```
client.py x server.py x
1  import socket
2      from threading import Thread
3      import threading
4  import math
5
6
7  class Server:
8
9      def __init__(self):
10         sock.bind(('localhost', 9090))
11         sock.listen(10)
12         print("Server was started on socket:", sock)
13         print("Wait for users...")
14
15     def get_socket(self):
16         while True:
17             client, addr = sock.accept()
18             clients.append({"connection": client, "socket": addr})
19             print("connected:", addr)
20             Thread(target=self.check_messages, args=(client,)).start()
21
22     def check_messages(self, client):
23         while True:
24             try:
25                 data = client.recv(4096)
26                 data = str(data.decode("utf-8"))
27                 if not data == "Square Equations":
```

```

25         data = client.recv(4096)
26         data = str(data.decode("utf-8"))
27         if not data == "Square Equations":
28             self.target_cast(client, "Такой задачи нет")
29             continue
30         self.target_cast(client, "Введите катеты а, b через пробел")
31         data = client.recv(4096)
32         data = str(data.decode("utf-8"))
33         a, b = map(float, [i.strip() for i in data.split()])
34         self.target_cast(client, "Гипотенуза равна: %.2f" % math.sqrt(math.pow(a,2)+math.pow(b,2)))
35     except Exception as ex:
36         print(ex)
37         self.delete_user_socket(client)
38         break
39
40     def delete_user_socket(self, client):
41         print("delete user", client)
42         for user in clients:
43             if user["connection"] == client:
44                 clients.remove(user)
45                 break
46
47     def target_cast(self, client, message):
48         message += "\n"
49         client.send(message.encode())
50

```

```

45         break
46
47     def target_cast(self, client, message):
48         message += "\n"
49         client.send(message.encode())
50
51
52 ► if __name__ == "__main__":
53     sock = socket.socket()
54     clients = []
55     server = Server()
56     thread = threading.Thread(target=server.get_socket())
57     thread.start()
58     thread.join()

```

```
client.py x server.py x
1 import socket
2
3 task = "Square Equations"
4 encoding = "UTF-8"
5 sock = socket.socket()
6 sock.connect(('localhost', 9090))
7
8 sock.send(task.encode(encoding))
9 data = sock.recv(1024)
10 print(data.decode("UTF-8"))
11
12 numbers = input()
13
14 sock.send(numbers.encode(encoding))
15 data = sock.recv(1024)
16 print(data.decode("UTF-8"))
17 sock.close()
```

Результат работы программы:

```
Terminal: Local x Local (2) x +
C:\Users\rk319\Documents\Study\untitled\lab2>py server.py
Server was started on socket: <socket.socket fd=476, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 9090)>
Wait for users...
connected: ('127.0.0.1', 54883)
[WinError 10053] Программа на вашем хост-компьютере разорвала установленное подключение
delete user <socket.socket fd=488, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 9090), raddr=('127.0.0.1', 54883)>
```

```
Terminal: Local x Local (2) x +
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.
C:\Users\rk319\Documents\Study\untitled>cd lab2

C:\Users\rk319\Documents\Study\untitled\lab2>py client.py
Введите катеты a, b через пробел

3 4
Гипотенуза равна: 5.00

C:\Users\rk319\Documents\Study\untitled\lab2>
```

Практическое задание 3:

Реализовать серверную часть приложения. Клиент подключается к серверу. В ответ клиент получает http-сообщение, содержащее html-страницу, которую сервер подгружает из файла index.html.

Код программы:

```
client.py x server.py x index.html x
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Page</title>
6  </head>
7  <body>
8
9      <H1>WebPage</H1>
10
11 </body>
12 </html>
```

```
client.py x server.py x index.html x
1  import socket
2  from threading import Thread
3  import threading
4
5
6  class Server:
7
8      def __init__(self):
9          sock.bind(('localhost', 9090))
10         sock.listen(10)
11         print("Server was started on socket:", sock)
12         print("Wait for users...")
13
14     def get_socket(self):
15         while True:
16             client, addr = sock.accept()
17             clients.append({"connection": client, "socket": addr})
18             print("connected:", addr)
19             Thread(target=self.check_messages, args=(client,)).start()
20
21     def check_messages(self, client):
22         while True:
23             try:
24                 data = client.recv(1024)
25                 self.target_cast(client, open("index.html").read())
26             except Exception as ex:
27                 self.delete_user_socket(client)
28                 break
29
```

```
client.py x server.py x index.html x
25         self.target_cast(client, open("index.html").read())
26     except Exception as ex:
27         self.delete_user_socket(client)
28         break
29
30     def delete_user_socket(self, client):
31         for user in clients:
32             if user["connection"] == client:
33                 clients.remove(user)
34                 break
35
36     def target_cast(self, client, message):
37         headers = bytes(f'HTTP/1.0 200 OK\nContent-Type: text/html\n\n{message}', 'utf-8')
38         # message += "\n"
39         client.send(headers)
40
41
42 if __name__ == "__main__":
43     sock = socket.socket()
44     clients = []
45     server = Server()
46     thread = threading.Thread(target=server.get_socket())
47     thread.start()
48     thread.join()
```

```
client.py x server.py x index.html x
1  import socket
2
3  sock = socket.socket()
4  sock.connect(('localhost', 9090))
5  sock.send('Hello, server'.encode("UTF-8"))
6
7  data = sock.recv(1024)
8  sock.close()
9
10 print(data.decode("UTF-8"))
```

Результат работы программы:

```
Terminal: Local (2) x Local x +
Microsoft Windows [Version 10.0.19041.746]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\rk319\Documents\Study\untitled>cd lab3

C:\Users\rk319\Documents\Study\untitled\lab3>py server.py
Server was started on socket: <socket.socket fd=424, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 9090)>
Wait for users...
connected: ('127.0.0.1', 54936)
```



```
Terminal: Local (2) × Local × +
Microsoft Windows [Version 10.0.19041.746]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\rk319\Documents\Study\untitled>cd lab3

C:\Users\rk319\Documents\Study\untitled\lab3>py client.py
HTTP/1.0 200 OK
Content-Type: text/html

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Page</title>
</head>
<body>

<H1>WebPage</H1>

</body>
</html>

C:\Users\rk319\Documents\Study\untitled\lab3>
```

Практическое задание 4:

Реализовать двухпользовательский или многопользовательский чат. Реализация многопользовательского чата позволяет получить максимальное количество баллов.

Код программы:

```
client1.py × client2.py × client3.py × server.py ×
1  import socket
2  from threading import Thread
3  import threading
4
5
6  class Server:
7
8      def __init__(self):
9          sock.bind(('localhost', 9090))
10         sock.listen(10)
11         print("Server was started on socket:", sock)
12         print("Wait for users...")
13
14     def get_socket(self):
15         while True:
16             client, addr = sock.accept()
17             clients.append({"connection": client, "socket": addr})
18             print("connected:", addr)
19             Thread(target=self.check_messages, args=(client,)).start()
20
21     def check_messages(self, client):
22         while True:
23             try:
24                 data = client.recv(4096)
25                 message = str(data.decode("utf-8"))
26                 self.broadcast(client, message)
27             except Exception as ex:
28                 self.delete_user_socket(client)
29                 break
```

```
client1.py × client2.py × client3.py × server.py ×
25         message = str(data.decode("utf-8"))
26         self.broadcast(client, message)
27     except Exception as ex:
28         self.delete_user_socket(client)
29         break
30
31     def delete_user_socket(self, client):
32         for user in clients:
33             if user["connection"] == client:
34                 clients.remove(user)
35                 break
36
37     def broadcast(self, client, message: str):
38         message += "\n"
39         for user in clients:
40             if user["connection"] != client:
41                 user["connection"].send(message.encode())
42
43
44 ▶ if __name__ == "__main__":
45     sock = socket.socket()
46     clients = []
47     server = Server()
48     thread = threading.Thread(target=server.get_socket())
49     thread.start()
50     thread.join()
```

```
client1.py × client2.py × client3.py × server.py ×
1  import socket
2  import threading
3
4  encoding = "UTF-8"
5  user_name = "user3"
6
7
8  def get_message():
9      try:
10         while True:
11             message = sock.recv(1024).decode(encoding)
12             print(message)
13         except ConnectionResetError:
14             pass
15
16
17  def send_message():
18      try:
19         while True:
20             message = (user_name + ": " + input()).encode(encoding)
21             sock.send(message)
22         except Exception:
23             pass
24
25  if __name__ == '__main__':
26      sock = socket.socket()
27      sock.connect(('localhost', 9090))
28
```

```
27      sock.connect(('localhost', 9090))
28
29      get_message_thread = threading.Thread(target=get_message)
30      get_message_thread.start()
31
32      send_message_thread = threading.Thread(target=send_message)
33      send_message_thread.start()
34
35      get_message_thread.join()
36      send_message_thread.join()
37
38      sock.close()
```

```
client1.py × client2.py × client3.py × server.py ×
1  import socket
2  import threading
3
4  encoding = "UTF-8"
5  user_name = "user2"
6
7
8  def get_message():
9      try:
10         while True:
11             message = sock.recv(1024).decode(encoding)
12             print(message)
13         except ConnectionResetError:
14             pass
15
16
17  def send_message():
18      try:
19         while True:
20             message = (user_name + ": " + input()).encode(encoding)
21             sock.send(message)
22         except Exception:
23             pass
24
25
26  if __name__ == '__main__':
27      sock = socket.socket()
28      sock.connect(('localhost', 9090))
29
```

```
30     get_message_thread = threading.Thread(target=get_message)
31     get_message_thread.start()
32
33     send_message_thread = threading.Thread(target=send_message)
34     send_message_thread.start()
35
36     get_message_thread.join()
37     send_message_thread.join()
38
39     sock.close()
```

```
client1.py x client2.py x client3.py x server.py x
1  import socket
2  import threading
3
4  encoding = "UTF-8"
5  user_name = "user1"
6
7
8  def get_message():
9      try:
10         while True:
11             message = sock.recv(1024).decode(encoding)
12             print(message)
13         except ConnectionResetError:
14             pass
15
16
17  def send_message():
18      try:
19         while True:
20             message = (user_name + ": " + input()).encode(encoding)
21             sock.send(message)
22         except Exception:
23             pass
24
25  if __name__ == '__main__':
26      sock = socket.socket()
27      sock.connect(('localhost', 9090))
28
29      get_message_thread = threading.Thread(target=get_message)
30
31
32      send_message_thread = threading.Thread(target=send_message)
33      send_message_thread.start()
34
35      get_message_thread.join()
36      send_message_thread.join()
37
38      sock.close()
```

Результат работы программы:

```
Terminal: Local × Local (2) × Local (3) × Local (4) × +
C:\Users\rk319\Documents\Study\untitled\lab4>py server.py
Server was started on socket: <socket.socket fd=540, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 9090)>
Wait for users...
connected: ('127.0.0.1', 54952)
connected: ('127.0.0.1', 54953)
connected: ('127.0.0.1', 54955)
```

```
Terminal: Local × Local (2) × Local (3) × Local (4) × +
Microsoft Windows [Version 10.0.19041.746]
(с) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\rk319\Documents\Study\untitled>cd lab4

C:\Users\rk319\Documents\Study\untitled\lab4>py client1.py
Message1
user2: Message2

user3: Message3

█
```

```
Terminal: Local × Local (2) × Local (3) × Local (4) × +
Microsoft Windows [Version 10.0.19041.746]
(с) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\rk319\Documents\Study\untitled>cd lab4

C:\Users\rk319\Documents\Study\untitled\lab4>py client2.py
user1: Message1

Message2
user3: Message3
```

Terminal: Local × Local (2) × Local (3) × Local (4) × +

Microsoft Windows [Version 10.0.19041.746]

(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\rk319\Documents\Study\untitled>cd lab4

C:\Users\rk319\Documents\Study\untitled\lab4>py client3.py

user1: Message1

user2: Message2

Message3