

Lab 6: Message-oriented Communication

Kivotova Evgenia, DS-01

October 2019

1 Sending process + ls -la before and after upload

```
Host
ekivo@ekivo-computer: ~/Dev/DistrSys/lab6
ls -la
total 228
drwxr-xr-x 2 ekivo ekivo 4096 Okt  2 13:09 .
drwxr-xr-x 4 ekivo ekivo 4096 Okt  2 13:09 ..
-rw-rw-r-- 1 ekivo ekivo 220557 ceH 30 13:50 pic.png
-rw-r--r-- 1 ekivo ekivo 2411 Okt  2 13:01 send_file.py
ekivo@ekivo-computer:~/Dev/DistrSys/lab6$ python3 send_file.py pic.png
Progress: [ ] 100.0% Complete
ekivo@ekivo-computer:~/Dev/DistrSys/lab6$ python3 send_file.py pic.png
Progress: [ ] 100.0% Complete
ekivo@ekivo-computer:~/Dev/DistrSys/lab6$

Server
ubuntu@ip-172-31-40-159: ~/laba
ls -la
total 12
drwxrwxr-x 2 ubuntu ubuntu 4096 Oct  2 13:07 .
drwxr-xr-x 7 ubuntu ubuntu 4096 Oct  2 12:58 ..
-rw-rw-r-- 1 ubuntu ubuntu 2529 Oct  2 13:06 receive_file.py
ubuntu@ip-172-31-40-159:~/laba$ python3 receive_file.py &
[1] 1865
ubuntu@ip-172-31-40-159:~/laba$ ('188.130.155.161', 57340) connected as u1
u1 > Receiving pic.png ...
u1 > pic.png was received.
u1 disconnected
ubuntu@ip-172-31-40-159:~/laba$ ls -la
total 228
drwxrwxr-x 2 ubuntu ubuntu 4096 Oct  2 13:12 .
drwxr-xr-x 7 ubuntu ubuntu 4096 Oct  2 12:58 ..
-rw-rw-r-- 1 ubuntu ubuntu 220557 Oct  2 13:12 pic.png
-rw-rw-r-- 1 ubuntu ubuntu 2529 Oct  2 13:06 receive_file.py
ubuntu@ip-172-31-40-159:~/laba$ ('188.130.155.161', 57386) connected as u2
u2 > Receiving copy1_pic.png ...
u2 > copy1_pic.png was received.
u2 disconnected
ubuntu@ip-172-31-40-159:~/laba$ ls -la
total 444
drwxrwxr-x 2 ubuntu ubuntu 4096 Oct  2 13:13 .
drwxr-xr-x 7 ubuntu ubuntu 4096 Oct  2 12:58 ..
-rw-rw-r-- 1 ubuntu ubuntu 220557 Oct  2 13:13 copy1_pic.png
-rw-rw-r-- 1 ubuntu ubuntu 220557 Oct  2 13:12 pic.png
-rw-rw-r-- 1 ubuntu ubuntu 2529 Oct  2 13:06 receive_file.py
ubuntu@ip-172-31-40-159:~/laba$
```

2 Server source code

```
1 import socket
2 from threading import Thread
3 import os.path
4
5 clients = []
6
7
8 # Thread to listen one particular client
9 class ClientListener(Thread):
10     def __init__(self, name: str, sock: socket.socket):
11         super().__init__(daemon=True)
12         self.sock = sock
13         self.name = name
14
15     # clean up
16     def _close(self):
17         clients.remove(self.sock)
18         self.sock.close()
19         print(self.name + ' disconnected')
20
21     def run(self):
22         # read the length of filename
23         l_filename = int(self.sock.recv(4))
```

```

24     # read the name of file
25     data = self.sock.recv(1_filename)
26     if data:
27         filename = data.decode('utf-8')
28
29         #check if file exists and add 'copyi_' to the begining of its name
30         i = 0
31         extra_part = ''
32         while os.path.isfile(extra_part+filename):
33             i += 1
34             extra_part = 'copy'+str(i)+'_'
35         filename = extra_part+filename
36         f = open(filename, 'wb')
37         print( self.name,'>','Receiving',filename,'...')
38     else:
39         # if we got no data      client has disconnected
40         self._close()
41         # finish the thread
42         return
43     while True:
44         # try to read 1024 bytes from user
45         # this is blocking call, thread will be paused here
46         data = self.sock.recv(512)
47         if data:
48             f.write(data)
49         else:
50             # if we got no data      client has disconnected
51             print(self.name,'>',filename,'was received.')
52             f.close()
53             self._close()
54             # finish the thread
55             return
56
57
58 def main():
59     next_name = 1
60
61     # AF_INET      IPv4, SOCK_STREAM      TCP
62     sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
63     # reuse address; in OS address will be reserved after app closed for a while
64     # so if we close and imidiatly start server again      we'll get error
65     sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
66     # listen to all interfaces at 8800 port
67     sock.bind(('localhost', 8800))
68     sock.listen()
69     while True:
70         # blocking call, waiting for new client to connect
71         con, addr = sock.accept()
72         clients.append(con)
73         name = 'u' + str(next_name)
74         next_name += 1
75         print(str(addr) + ' connected as ' + name)
76         # start new thread to deal with client
77         ClientListener(name, con).start()
78
79
80 if __name__ == "__main__":
81     main()

```

3 Client source code

```

1  import socket
2  import sys
3  import os
4
5  # define server
6  SERVER_DOMAIN = 'ec2-18-212-138-166.compute-1.amazonaws.com'
7  SERVER_PORT = 8800
8
9  # define filename
10 FILE_NAME = sys.argv[1]
11
12 # Print file sending progress (taken from elekt.tech)
13 def printProgressBar (iteration, total, prefix = '', suffix = '', decimals = 1, length = 100, fill = '
14     """

```

```

15     Call in a loop to create terminal progress bar
16     @params:
17         iteration    - Required    : current iteration (Int)
18         total        - Required    : total iterations (Int)
19         prefix       - Optional    : prefix string (Str)
20         suffix       - Optional    : suffix string (Str)
21         decimals     - Optional    : positive number of decimals in percent complete (Int)
22         length       - Optional    : character length of bar (Int)
23         fill         - Optional    : bar fill character (Str)
24     """
25     percent = ("{0:." + str(decimals) + "f").format(100 * (iteration / float(total)))
26     filledLength = int(length * iteration // total)
27     bar = fill * filledLength + '-' * (length - filledLength)
28     print('\r%s |%s| %s%% %s' % (prefix, bar, percent, suffix), end = '\r')
29     # Print New Line on Complete
30     if iteration == total:
31         print()
32
33     # check the file exists
34     if not os.path.isfile(FILE_NAME):
35         print('ERROR:', FILE_NAME, 'is not in the current derictory.')
36         exit()
37
38     # connect to server
39     cli_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
40     cli_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
41     cli_socket.connect((SERVER_DOMAIN, SERVER_PORT))
42
43     # define the length of the filename, extend it if needed
44     name_length = bytes(str(len(FILE_NAME)), 'utf-8')
45     name_length = (b'0')*(4-len(name_length)) + name_length
46
47     # send name length to server
48     cli_socket.send(name_length)
49
50     # send file name to server
51     cli_socket.send(bytes(FILE_NAME, 'utf-8'))
52
53     # define the length of file
54     file_length = os.path.getsize(FILE_NAME)
55
56     # send file
57     with open(FILE_NAME, 'rb') as f:
58         # print empty progress
59         printProgressBar(0, file_length, prefix = 'Progress:', suffix = 'Complete', length = 50)
60         data = f.read(512)
61         sent = 0
62         while data:
63             # send data from file
64             sent += cli_socket.send(data)
65             # print progress
66             printProgressBar(sent, file_length, prefix = 'Progress:', suffix = 'Complete', length = 50)
67             data = f.read(512)
68         # close file
69         f.close()
70     # close connection
71     cli_socket.close()

```

4 GitHub

https://github.com/Genvekt/Distributive_Systems_F19