

# Full Duplex Chatroom

**BIJAY REGMI**

**LAB 11**

server.py

```
import socket
import select

def runSelect():
    selectUnsuccessful = True
    while selectUnsuccessful:
        try:
            readyRecvList, readySendList, readyErrList =
select.select(recvList, sendList, [])
            selectUnsuccessful = False
        except select.error:
            for fd in recvList:
                try:
                    tempRecvList, tempSendList, tempErrList =
select.select([fd], [], [], 0)
                    except select.error:
                        if fd == serverSocket:
                            fd.close()
                            exit(1)
                        else:
                            if fd in recvList:
                                recvList.remove(fd)

                            fd.close()

    return readyRecvList, readySendList

def handleListeningSocket():
    try:
        newConnectionSocket, addr = serverSocket.accept()
    except socket.error as err:
        print("\nERROR: Something went wrong in the accept()
function call:", err)
        exit(1)

    try:
        recvList.append(newConnectionSocket)
        sendList.append(newConnectionSocket)
```

```

        print ("INFO: Connecting socket created between %s
and %s" % (newConnectionSocket.getsockname(),
newConnectionSocket.getpeername()))
        print ("* Client %s is ready to chat *" %
(str(newConnectionSocket.getpeername())))
    except (socket.error, socket.gaierror) as err:
        print ("\nERROR: Something went wrong with the new
connection socket:", err)
        if newConnectionSocket in recvList:
            recvList.remove(newConnectionSocket)
            sendList.remove(newConnectionSocket)

        newConnectionSocket.close()

def handleConnectedSocket():
    try:

        recvIsComplete = False
        rcvdStr = ""

        while not recvIsComplete:
            rcvdStr = rcvdStr + fd.recv(1024)

            if fd not in sendList:
                sendList.append(fd)

            # ~ is the delimiter used to indicate message start
and finish
            if rcvdStr.strip('~') != "":
                if (rcvdStr[0] == "~") and (rcvdStr[-1] == "~"):
                    recvIsComplete = True
                    clientMessage = rcvdStr.strip('~')
                else: # if empty string, connection has been
terminated
                    if fd in recvList:
                        recvList.remove(fd)

                    if fd in sendList:
                        sendList.remove(fd)

                    del clientMessages[fd] # Delete connection
information
                    fd.close()

            if clientMessage == "quit()":
                print ("\n* Client %s has left the chat room *\n" %
(str(fd.getpeername())))

                if fd in recvList:
                    recvList.remove(fd)
                    fd.close()

```

```

        if fd in sendList:
            sendList.remove(fd)
            fd.close()

    else:
        print ("\n%s: %s" % (fd.getpeername(), clientMessage))
        clientMessages[fd] = str(clientMessage) # add message
to dictionary, pending transmission

    except socket.error as err:
        print ("\nERROR: Connection to the client has abruptly
ended:", err)
        if fd in recvList:
            recvList.remove(fd)

        if fd in sendList:
            sendList.remove(fd)

        fd.close()
        print ("* I am ready to chat with a new client! *\n")

"""
main - Runs the Full Duplex Chat server
"""

# Global Variables
serverHost = 'localhost'
serverPort = 22222
recvList = []
sendList = []
clientMessages = {}

try:

    serverSocket = socket.socket(socket.AF_INET,
socket.SOCK_STREAM)
    serverSocket.setblocking(0)
    serverSocket.setsockopt(socket.SOL_SOCKET,
socket.SO_REUSEADDR, 1)
    serverSocket.bind((serverHost, serverPort))
    serverSocket.listen(3)

    print ("INFO: I am listening at %s" %
(str(serverSocket.getsockname())))
    print ("* I am ready to chat with a new client! *\n")

except (socket.error, socket.gaierror) as err:
    print ("\nERROR: Something went wrong in creating the
listening socket:", err)
    exit(1)

```

```

recvList = [serverSocket]

try:
    while True:
        serverSocket.setblocking(False)
        readyForRecv, readyForSend = runSelect()

        for fd in readyForRecv:
            if fd == serverSocket:
                handleListeningSocket()
            else:
                handleConnectedSocket()

        for fd in readyForSend:
            try:
                if fd in clientMessages.keys(): # See if
connection information exists
                    broadcast = str(clientMessages[fd]) # Add
message to broadcast variable

                    if broadcast: # See if a message is actually
there
                        for client in readyForSend: # Broadcast
message to every connected client
                            if broadcast != "":
                                print ("* Broadcasting message \"%s\"
to %s *" % (str(broadcast), client.getpeername()))
                                client.send(str(fd.getpeername()) + ":
" + str(broadcast))

                                clientMessages[fd] = "" # Empty pending
messages

            except:
                # print "\nERROR: Something awful happened while
broadcasting messages"
                break

except socket.error as err:
    print ("\nERROR: Something awful happened with a connected
socket:", err)

    if fd in recvList:
        recvList.remove(fd)

    if fd in sendList:
        sendList.remove(fd)

    fd.close()

except KeyboardInterrupt:
    for fd in recvList:
        fd.close()

```

```

for fd in sendList:
    fd.close()

print ("\nINFO: KeyboardInterrupt")
print ("* Closing all sockets and exiting... Goodbye! *")
exit(0)

```

## **client.py (3 client files with same code)**

```

import socket
import select
import sys

def main():
    """
    main - Runs the Full Duplex Chat Client
    """

    serverHost = 'localhost'
    serverPort = 22222

    try:
        clientSocket = socket.socket(socket.AF_INET,
socket.SOCK_STREAM)
    except socket.error as err:
        print ("ERROR: Cannot create client side socket:", err)
        exit(1)

    while True:
        try:
            clientSocket.connect((serverHost, serverPort))
        except socket.error as err:
            print ("ERROR: Cannot connect to chat server", err)
            print ("* Exiting... Goodbye! *")
            exit(1)
        except:
            print ("ERROR: Something awful happened!")
            exit(1)
        break

    recvList = [clientSocket, sys.stdin]

    print ("* You are now connected to chat server %s as %s *" %
(clientSocket.getpeername(), clientSocket.getsockname()))

    try:
        while True:
            readyRecvList, readySendList, readyErrList =
select.select(recvList, [], [])

```

```

        for fd in readyRecvList:
            if fd == sys.stdin:
                message = sys.stdin.readline().rstrip()
                clientSocket.sendall("~" + str(message) + "~")

                if (message == "quit()"):
                    print ("* Exiting chat room! *")
                    clientSocket.close()
                    exit(0)
                    break

            elif fd == clientSocket:
                clientSocket.settimeout(3)
                try:
                    message = clientSocket.recv(2048)
                except socket.timeout as err:
                    print ("ERROR: The recv() function timed
out after 3 seconds! Try again.")
                except:
                    print ("ERROR: Something awful happened!")
                else:
                    if message == "":
                        break
                    else:
                        print ("%s\n" % (message))
                        clientSocket.settimeout(None)
                        break

    except select.error as err:
        for fd in recvList:
            try:
                tempRecvList, tempSendList, tempErrList =
select.select([fd], [], [], 0)
            except select.error:
                if fd == clientSocket:
                    fd.close()
                    exit(1)
                else:
                    if fd in recvList:
                        recvList.remove(fd)
                        fd.close()

    except socket.error as err:
        print ("ERROR: Cannot connect to chat server", err)
        print ("* Exiting... Goodbye! *")
        exit(1)

    if fd in recvList:
        fd.close()

except KeyboardInterrupt:
    print ("\nINFO: KeyboardInterrupt")

```

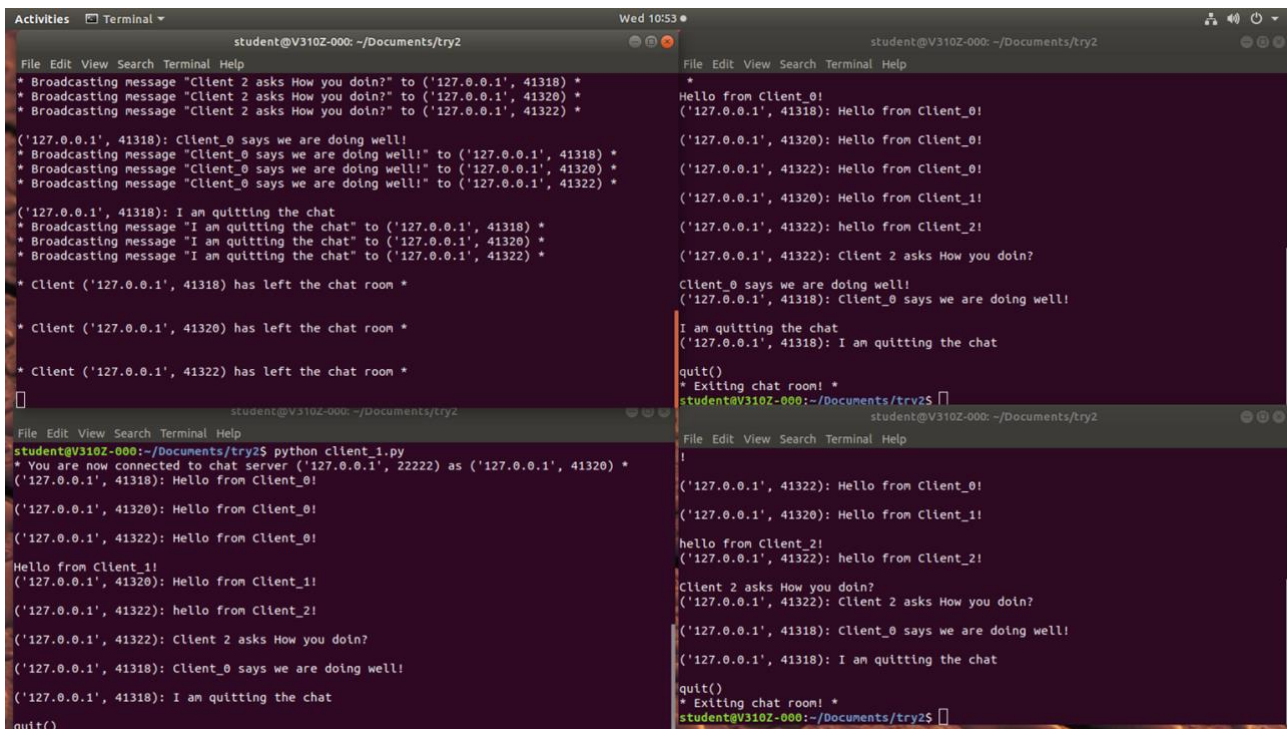
```

        print ("* Closing all sockets and exiting chat server...
Goodbye! *")
        clientSocket.close()
        exit(0)

if __name__ == '__main__':
    main()

```

## INPUT/OUTPUT



```

student@V310Z-000: ~/Documents/try2
File Edit View Search Terminal Help
* Broadcasting message "Client 2 asks How you doin?" to ('127.0.0.1', 41318) *
* Broadcasting message "Client 2 asks How you doin?" to ('127.0.0.1', 41320) *
* Broadcasting message "Client 2 asks How you doin?" to ('127.0.0.1', 41322) *
('127.0.0.1', 41318): Client_0 says we are doing well!
* Broadcasting message "Client_0 says we are doing well!" to ('127.0.0.1', 41318) *
* Broadcasting message "Client_0 says we are doing well!" to ('127.0.0.1', 41320) *
* Broadcasting message "Client_0 says we are doing well!" to ('127.0.0.1', 41322) *
('127.0.0.1', 41318): I am quitting the chat
* Broadcasting message "I am quitting the chat" to ('127.0.0.1', 41318) *
* Broadcasting message "I am quitting the chat" to ('127.0.0.1', 41320) *
* Broadcasting message "I am quitting the chat" to ('127.0.0.1', 41322) *
* Client ('127.0.0.1', 41318) has left the chat room *
* Client ('127.0.0.1', 41320) has left the chat room *
* Client ('127.0.0.1', 41322) has left the chat room *
quit()

student@V310Z-000: ~/Documents/try2
File Edit View Search Terminal Help
*
Hello from Client_0!
('127.0.0.1', 41318): Hello from Client_0!
('127.0.0.1', 41320): Hello from Client_0!
('127.0.0.1', 41322): Hello from Client_0!
('127.0.0.1', 41320): Hello from Client_1!
('127.0.0.1', 41322): Hello from Client_2!
('127.0.0.1', 41322): Client 2 asks How you doin?
Client_0 says we are doing well!
('127.0.0.1', 41318): Client_0 says we are doing well!
I am quitting the chat
('127.0.0.1', 41318): I am quitting the chat
quit()
* Exiting chat room! *
student@V310Z-000: ~/Documents/try2$

student@V310Z-000: ~/Documents/try2$ python client_1.py
* You are now connected to chat server ('127.0.0.1', 22222) as ('127.0.0.1', 41320) *
('127.0.0.1', 41318): Hello from Client_0!
('127.0.0.1', 41320): Hello from Client_0!
('127.0.0.1', 41322): Hello from Client_0!
Hello from Client_1!
('127.0.0.1', 41320): Hello from Client_1!
('127.0.0.1', 41322): Hello from Client_2!
('127.0.0.1', 41322): Client 2 asks How you doin?
('127.0.0.1', 41318): Client_0 says we are doing well!
('127.0.0.1', 41318): I am quitting the chat
quit()

student@V310Z-000: ~/Documents/try2$
File Edit View Search Terminal Help
!
('127.0.0.1', 41322): Hello from Client_0!
('127.0.0.1', 41320): Hello from Client_1!
Hello from Client_2!
('127.0.0.1', 41322): Hello from Client_2!
Client 2 asks How you doin?
('127.0.0.1', 41322): Client 2 asks How you doin?
('127.0.0.1', 41318): Client_0 says we are doing well!
('127.0.0.1', 41318): I am quitting the chat
quit()
* Exiting chat room! *
student@V310Z-000: ~/Documents/try2$

```