

CHAT.py

```
File Edit Selection Find View Goto Tools Project Preferences Help
chat.py x chatserver.py x
1 from twisted.internet.protocol import Factory
2 from twisted.protocols.basic import LineReceiver
3 from twisted.internet import reactor
4
5 class Chat(LineReceiver):
6
7     def __init__(self, users):
8         self.users = users
9         self.name = None
10        self.state = "GETNAME" #initial state
11
12    def connectionMade(self):
13        self.sendLine("What's your name?")
14
15    def connectionLost(self, reason):
16        if self.name in self.users:
17            del self.users[self.name]
18
19    def lineReceived(self, line):
20        if self.state == "GETNAME":
21            self.handle_GETNAME(line)
22        else:
23            self.handle_CHAT(line)
24            if(line == "Bye"):
25                self.handle_DISCONNECT(line)
26
27    def handle_GETNAME(self, name):
28        if name in self.users:
29            self.sendLine("Name taken, please choose another.")
30            return
31        self.sendLine("Welcome, %s!" % (name,))
32        self.name = name
33        self.users[name] = self
34        self.state = "CHAT"
35
36    def handle_CHAT(self, message):
37        message = "<%s> %s" % (self.name, message)
38        for name, protocol in self.users.iteritems():
39            if protocol != self:
40                protocol.sendLine(message)
41
42    def handle_DISCONNECT(self, message):
43        print("Lost a client!")
44        self.factory.clients.remove(self)
45
```

```

45
46
47 class ChatFactory(Factory):
48
49     def __init__(self):
50         self.users = {} # maps user names to Chat instances
51
52     def buildProtocol(self, addr):
53         return Chat(self.users)
54
55
56 reactor.listenTCP(8123, ChatFactory())
57 reactor.run()
58

```

SOURCE CODE:

```

from twisted.internet.protocol import Factory
from twisted.protocols.basic import LineReceiver
from twisted.internet import reactor

```

```

class Chat(LineReceiver):

```

```

    def __init__(self, users):
        self.users = users
        self.name = None
        self.state = "GETNAME" #initial state

```

```

    def connectionMade(self):
        self.sendLine("What's your name?")

```

```

    def connectionLost(self, reason):
        if self.name in self.users:
            del self.users[self.name]

```

```

    def lineReceived(self, line):

```

```
if self.state == "GETNAME":
    self.handle_GETNAME(line)
else:
    self.handle_CHAT(line)
    if(line == "Bye"):
        self.handle_DISCONNECT(line)
```

```
def handle_GETNAME(self, name):
    if name in self.users:
        self.sendLine("Name taken, please choose another.")
        return
    self.sendLine("Welcome, %s!" % (name,))
    self.name = name
    self.users[name] = self
    self.state = "CHAT"
```

```
def handle_CHAT(self, message):
    message = "<%s> %s" % (self.name, message)
    for name, protocol in self.users.iteritems():
        if protocol != self:
            protocol.sendLine(message)
```

```
def handle_DISCONNECT(self,message):
    print("Lost a client!")
    self.factory.clients.remove(self)
```

```
class ChatFactory(Factory):
```

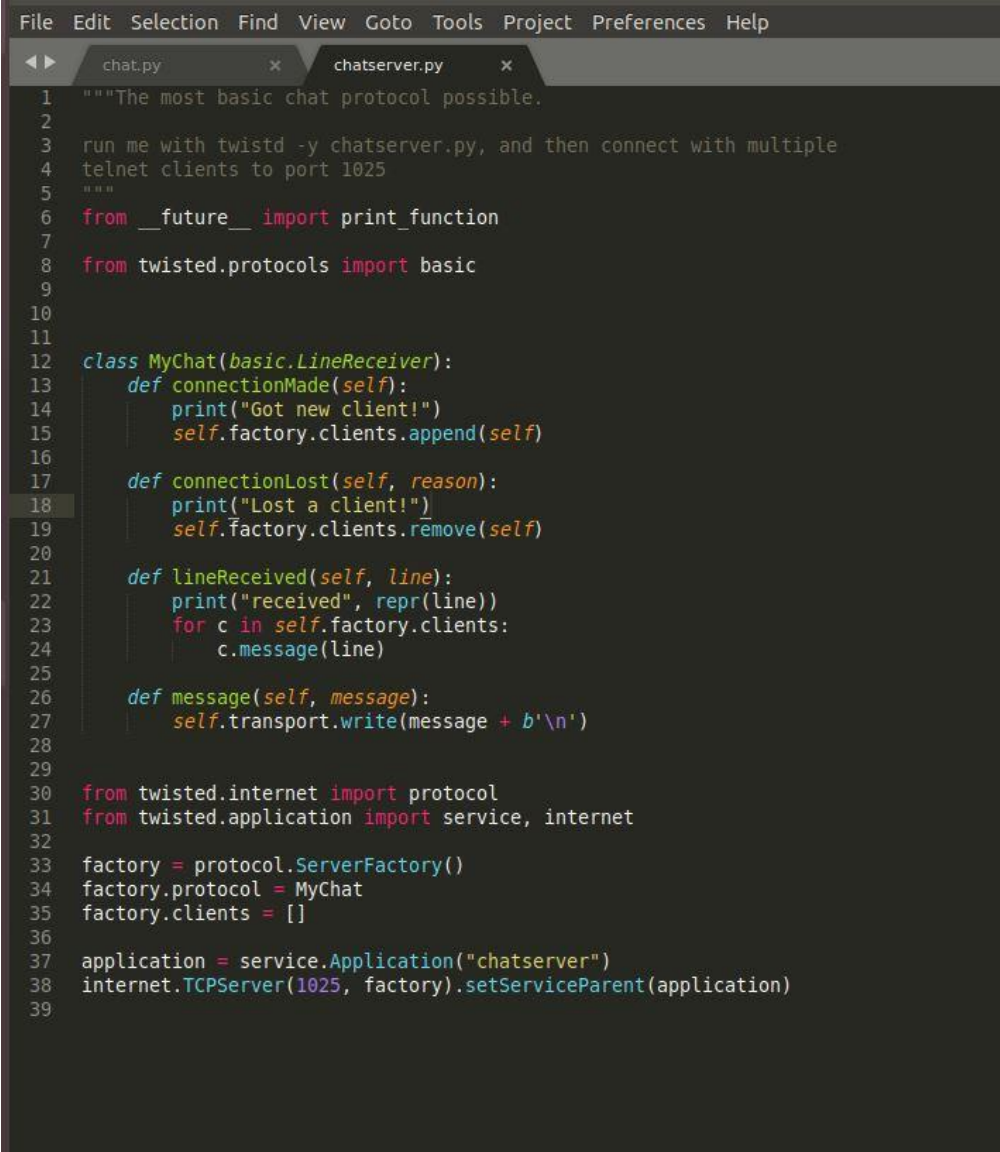
```
def __init__(self):  
    self.users = {} # maps user names to Chat instances
```

```
def buildProtocol(self, addr):  
    return Chat(self.users)
```

```
reactor.listenTCP(8123, ChatFactory())
```

```
reactor.run()
```

CHATSERVER.py



```
File Edit Selection Find View Goto Tools Project Preferences Help  
chat.py x chatserver.py x  
1 """The most basic chat protocol possible.  
2  
3 run me with twisted -y chatserver.py, and then connect with multiple  
4 telnet clients to port 1025  
5 """  
6 from __future__ import print_function  
7  
8 from twisted.protocols import basic  
9  
10  
11  
12 class MyChat(basic.LineReceiver):  
13     def connectionMade(self):  
14         print("Got new client!")  
15         self.factory.clients.append(self)  
16  
17     def connectionLost(self, reason):  
18         print("Lost a client!")  
19         self.factory.clients.remove(self)  
20  
21     def lineReceived(self, line):  
22         print("received", repr(line))  
23         for c in self.factory.clients:  
24             c.message(line)  
25  
26     def message(self, message):  
27         self.transport.write(message + b'\n')  
28  
29  
30 from twisted.internet import protocol  
31 from twisted.application import service, internet  
32  
33 factory = protocol.ServerFactory()  
34 factory.protocol = MyChat  
35 factory.clients = []  
36  
37 application = service.Application("chatserver")  
38 internet.TCPServer(1025, factory).setServiceParent(application)  
39
```

Source Code:

```
"""The most basic chat protocol possible.
```

```
run me with twisted -y chatserver.py, and then connect with multiple  
telnet clients to port 1025
```

```
"""
```

```
from __future__ import print_function
```

```
from twisted.protocols import basic
```

```
class MyChat(basic.LineReceiver):
```

```
    def connectionMade(self):
```

```
        print("Got new client!")
```

```
        self.factory.clients.append(self)
```

```
    def connectionLost(self, reason):
```

```
        print("Lost a client!")
```

```
        self.factory.clients.remove(self)
```

```
    def lineReceived(self, line):
```

```
        print("received", repr(line))
```

```
        for c in self.factory.clients:
```

```
            c.message(line)
```

```
    def message(self, message):
```

```
        self.transport.write(message + b"\n")
```

```
from twisted.internet import protocol

from twisted.application import service, internet
```

```
factory = protocol.ServerFactory()
```

```
factory.protocol = MyChat
```

```
factory.clients = []
```

```
application = service.Application("chatserver")
```

```
internet.TCPServer(1025, factory).setServiceParent(application)
```

Output:

The screenshot shows a terminal window titled 'Ubuntu [Running] - Oracle VM VirtualBox' with a date and time of 'Wed 22:10'. The terminal is divided into three panes. The top pane shows the server setup: navigating to the directory `~/cs531/Q2`, listing files, and running `python chatserver.py`, which outputs 'Lost a client!'. The bottom-left pane shows a telnet client connecting to 127.0.0.1 on port 8123, interacting with the server, and then closing the connection. The bottom-right pane shows another telnet client connecting to the same address, interacting with the server, and then closing the connection. The terminal background is orange with a dark purple terminal window overlay.

```
niraj_ubuntu@niraj-ubuntu: ~/cs531/Q2
File Edit View Search Terminal Help
niraj_ubuntu@niraj-ubuntu:~$ cd cs531/
niraj_ubuntu@niraj-ubuntu:~/cs531$ cd Q2/
niraj_ubuntu@niraj-ubuntu:~/cs531/Q2$ ls
chat.py  chatserver.py
niraj_ubuntu@niraj-ubuntu:~/cs531/Q2$ python chatserver.py
Lost a client!

niraj_ubuntu@niraj-ubuntu:~$ telnet 127.0.0.1 8123
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^J'.
What's your name?
Mom
Welcome, Mom!
Hello How are you ?
<Niraj> I am fine thank you!
<Niraj> I Miss You!
I miss you too!
Bye
Connection closed by foreign host.
niraj_ubuntu@niraj-ubuntu:~$

niraj_ubuntu@niraj-ubuntu:~$ telnet 127.0.0.1 8123
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^J'.
What's your name?
Niraj
Welcome, Niraj!
<Mom> Hello How are you ?
I am fine thank you!
I Miss You!
<Mom> I miss you too!
<Mom> Bye
bye
Connection closed by foreign host.
niraj_ubuntu@niraj-ubuntu:~$
```