NAME : NIRAJ THANKI SID : 19376 CLASS: CS531

CHAT.py

File Edit Selection Find View Goto Tools Project Preferences Help

chat.py chatserver.py

```python
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)

```

```python


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()

```

SOURCE CODE:

```python
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 | chatserver.py

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

run me with twistd -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)

```

Source Code:

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

run me with twistd -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: