

Q-2) Parsing with SAX

Execution Result:-

```

C:\hs6\SAX>java -jar hs6.jar
=====Movie=====
Title: A History of Violence
Year: 2005
Country: USA
Genre: Crime
Summary: Tom Stall, a humble family man and member of a popular neighborhood post
squad, lives a quiet but fulfilling existence in the Midwest. One night Tom has
to witness at his place of business one, he his character is placed all over t
he man for his heroic. Following this, mysterious people follow the Stall's ev
ery move, considering Tom more than anyone else. As this situation is confronted,
more truths are uncovered where all these occurrences have stemmed from compromising
all marriage, family relationships and the more characters' former relations in a
he process.
=====Director=====
Last Name: Cronenberg
First Name: David
Birth Date: 1943
=====Actor=====
First Name: Viggo
Last Name: Mortensen
Birth Date: 1958
Role: Tom Stall
=====Actress=====
First Name: Maria
Last Name: Bello
Birth Date: 1980
Role: Linda Stall
=====Actor=====
First Name: Ed
Last Name: Harris
Birth Date: 1950
Role: Carl Fugarty
=====Actress=====
First Name: Rachel
Last Name: Nichols
Birth Date: 1960
Role: Rachel Nichols
=====Movie=====
Title: Heat
Year: 1995
Country: USA
Genre: Crime
Summary: Heaters and their grays-shed and his professional criminal crew hunt to
score big money targets (banks, vaults, armored cars) and are, in turn, hunted
by LA Vincent Hanna and his team of cops in the Authority/Justice police division
. A hotbed job sets Hanna onto their trail while they regroup and try to put it
together and last his retirement. Jerry, bull and Vincent are similar in many wa
ys, including their criminal personal lives, as a crucial moment in his life, he
'll discover the drastic taught to his long ago by his criminal mentor--"have the
a decision on your first then you can't walk me up in their streets. If, if you
e just the heat coming around the corner--as he falls in love. Thus the stage i
s set for the commercial working...
=====Director=====
Last Name: Sam
First Name: Michael
Birth Date: 1943
=====Actor=====
First Name: Al
Last Name: Pacino
Birth Date: 1940
Role: Lt. Vincent Hanna
=====Movie=====
Title: Infamous
Year: 1996
Country: USA
Genre: Western
Summary: The time of big whiskey to fall of normal people trying to lead quiet li
ves. Cowboys try to make a living. Sheriff "Little Bill" tries to build a home
and keep a peace-keeping order. The town where just try to get by. Then a couple
of cowboys cut up a whore, identified with Bill's justice, the prostitutes and
a sheriff and the cowboys. The bounty attracts a young son killing himself as "The
Infamous Old" and more killing William Munny. Munny returned for his young son
Fe, and he has having raising traps and his children in peace. But his wife is gone.
His life is hard. And Munny is no good at the job he tells his old partner Mac.
tadles his angry nag, and rides off to kill one more time, blurring the lines
between justice and villainy, man and myth.
=====Director=====
Last Name: Eastwood
First Name: Clint
Birth Date: 1930
=====Actor=====
First Name: Clint
Last Name: Eastwood
Birth Date: 1930
Role: William "Bill" Munny
=====Actor=====
First Name: Gene
Last Name: Hackman
Birth Date: 1930
Role: Little Bill Daggett
=====Actor=====
First Name: Morgan
Last Name: Freeman
Birth Date: 1937
Role: Ned Pepper
=====Movie=====
Title: Match Point
Year: 2005
Country: USA
Genre: Crime
Summary: Chris is written to a former tennis pro, looking to find work as an instruo
r. He meets Tom Hewitt, a well-off pretty boy, Tom's sister Chloe falls in love
with Chris but Chris has his eyes on Tom's fiance, the lesbian Nola. Nola Ch
ris and Nola know it's wrong but what could be more right than love? Chris tries
to supply both women but at some point, he must choose between them...
=====Director=====
Last Name: Allen
First Name: Woody
Birth Date: 1935
```

```

C:\hs6\SAX>java -jar hs6.jar
=====Movie=====
Title: Heat
Year: 1995
Country: USA
Genre: Crime
Summary: Heaters and their grays-shed and his professional criminal crew hunt to
score big money targets (banks, vaults, armored cars) and are, in turn, hunted
by LA Vincent Hanna and his team of cops in the Authority/Justice police division
. A hotbed job sets Hanna onto their trail while they regroup and try to put it
together and last his retirement. Jerry, bull and Vincent are similar in many wa
ys, including their criminal personal lives, as a crucial moment in his life, he
'll discover the drastic taught to his long ago by his criminal mentor--"have the
a decision on your first then you can't walk me up in their streets. If, if you
e just the heat coming around the corner--as he falls in love. Thus the stage i
s set for the commercial working...
=====Director=====
Last Name: Sam
First Name: Michael
Birth Date: 1943
=====Actor=====
First Name: Al
Last Name: Pacino
Birth Date: 1940
Role: Lt. Vincent Hanna
=====Movie=====
Title: Infamous
Year: 1996
Country: USA
Genre: Western
Summary: The time of big whiskey to fall of normal people trying to lead quiet li
ves. Cowboys try to make a living. Sheriff "Little Bill" tries to build a home
and keep a peace-keeping order. The town where just try to get by. Then a couple
of cowboys cut up a whore, identified with Bill's justice, the prostitutes and
a sheriff and the cowboys. The bounty attracts a young son killing himself as "The
Infamous Old" and more killing William Munny. Munny returned for his young son
Fe, and he has having raising traps and his children in peace. But his wife is gone.
His life is hard. And Munny is no good at the job he tells his old partner Mac.
tadles his angry nag, and rides off to kill one more time, blurring the lines
between justice and villainy, man and myth.
=====Director=====
Last Name: Eastwood
First Name: Clint
Birth Date: 1930
=====Actor=====
First Name: Clint
Last Name: Eastwood
Birth Date: 1930
Role: William "Bill" Munny
=====Actor=====
First Name: Gene
Last Name: Hackman
Birth Date: 1930
Role: Little Bill Daggett
=====Actor=====
First Name: Morgan
Last Name: Freeman
Birth Date: 1937
Role: Ned Pepper
=====Movie=====
Title: Match Point
Year: 2005
Country: USA
Genre: Crime
Summary: Chris is written to a former tennis pro, looking to find work as an instruo
r. He meets Tom Hewitt, a well-off pretty boy, Tom's sister Chloe falls in love
with Chris but Chris has his eyes on Tom's fiance, the lesbian Nola. Nola Ch
ris and Nola know it's wrong but what could be more right than love? Chris tries
to supply both women but at some point, he must choose between them...
=====Director=====
Last Name: Allen
First Name: Woody
Birth Date: 1935
```

```
~\hdsax
#####
First Name: Morgan
Last Name: Freeman
Birth Date: 1957
Role: Mel Gibson
#####Movie#####
Title: Match Point
Year: 2005
Country: USA
Genre: Crime
Summary: Chris Wilton is a former tennis pro, looking to find work as an instructor. He meets Tom Hewitt, a well-off pretty boy, who's sister Chloe falls in love with Chris but Chris has his eyes on Tom's sister. The tension boils. Both Chris and Tom have to bring his wife home so he must choose between them...
#####Director#####
Last Name: Allen
First Name: Woody
Birth Date: 1935
#####Actor#####
First Name: Jonathan
Last Name: Rhys Meyers
Birth Date: 1977
Role: Chris Wilton
#####Actor#####
First Name: Scarlett
Last Name: Johansson
Birth Date: 1984
Role: Julia Rice
#####Movie#####
Title: Lost in Translation
Year: 2003
Country: USA
Genre: Drama
#####Director#####
Last Name: Coppola
First Name: Sofia
Birth Date: 1971
#####Actor#####
First Name: Scarlett
Last Name: Johansson
Birth Date: 1984
Role: Charlotte
#####Actor#####
First Name: Bill
Last Name: Murray
Birth Date: 1950
Role: Bob Harris
#####Movie#####
Title: Marie Antoinette
Year: 2006
Country: USA
Genre: Drama
Summary: Based on Antonia Fraser's book about the ill-fated Archduchess of Austria and later Queen of France, Marie Antoinette tells the story of the most misunderstood and abused woman in history, from her birth in imperial Austria to her later life in France.
#####Director#####
Last Name: Coppola
First Name: Sofia
Birth Date: 1971
#####Actor#####
First Name: Kirsten
Last Name: Dunst
Birth Date: 1982
Role: Marie Antoinette
#####Actor#####
First Name: Jason
Last Name: Schwartzman
Birth Date: 1980
Role: Louis XVI
#####Movie#####
Title: Spider-Man
Year: 2002
Country: USA
Genre: Action
Summary:
#####Director#####
Last Name: Raimi
First Name: Sam
Birth Date: 1965
#####Actor#####
First Name: Kirsten
Last Name: Dunst
Birth Date: 1982
Role: Mary Jane Watson
#####Actor#####
First Name: Tobey
Last Name: Maguire
Birth Date: 1975
Role: Spider-Man / Peter Parker
#####Actor#####
First Name: Willem
Last Name: Dafoe
Birth Date: 1955
Role: Green Goblin / Norman Osborn
#####
~\hdsax>
1 |
```

MovieHandler.py:-

#!/usr/bin/python

import xml.sax

class MovieHandler(xml.sax.ContentHandler):

def __init__(self):

self.CurrentData = ""

```
self.title = ""  
self.year = ""  
self.country = ""  
self.genre = ""  
self.summary = ""  
self.director = ""  
self.last_name = ""  
self.first_name = ""  
self.birth_date = ""  
self.role = ""
```

Call when an element starts

```
def startElement(self, tag, attributes):
```

```
    self.CurrentData = tag
```

```
    if tag == "movie":
```

```
        print ("\n*****Movie*****")
```

```
    elif tag == "director":
```

```
        print ("\n*****Directors*****")
```

```
    elif tag == "actor":
```

```
        print ("\n*****Actor*****")
```

Call when an elements ends

```
def endElement(self, tag):
```

```
if self.CurrentData == "title":
    print ("Title:", self.title)
elif self.CurrentData == "year":
    print ("Year:", self.year)
elif self.CurrentData == "country":
    print ("Country:", self.country)
elif self.CurrentData == "genre":
    print ("Genre:", self.genre)
elif self.CurrentData == "summary":
    print ("Summary:", self.summary)
elif self.CurrentData == "director":
    print ("Director:", self.director)
elif self.CurrentData == "last_name":
    print ("Last Name:", self.last_name)
elif self.CurrentData == "first_name":
    print ("First Name:", self.first_name)
elif self.CurrentData == "birth_date":
    print ("Birth Date:", self.birth_date)
elif self.CurrentData == "role":
    print ("Role:", self.role)
self.CurrentData = ""
```

```
# Call when a character is read
def characters(self, content):
    if self.CurrentData == "title":
        self.title = content
    elif self.CurrentData == "year":
        self.year = content
    elif self.CurrentData == "country":
        self.country = content
    elif self.CurrentData == "genre":
        self.genre = content
    elif self.CurrentData == "summary":
        self.summary = content
    elif self.CurrentData == "director":
        self.director = content
    elif self.CurrentData == "last_name":
        self.last_name = content
    elif self.CurrentData == "first_name":
        self.first_name = content
    elif self.CurrentData == "birth_date":
        self.birth_date = content
    elif self.CurrentData == "role":
        self.role = content
```

```
if ( __name__ == "__main__"):
```

```
# create an XMLReader
```

```
parser = xml.sax.make_parser()
```

```
# turn off namespaces
```

```
parser.setFeature(xml.sax.handler.feature_namespaces, 0)
```

```
# override the default ContextHandler
```

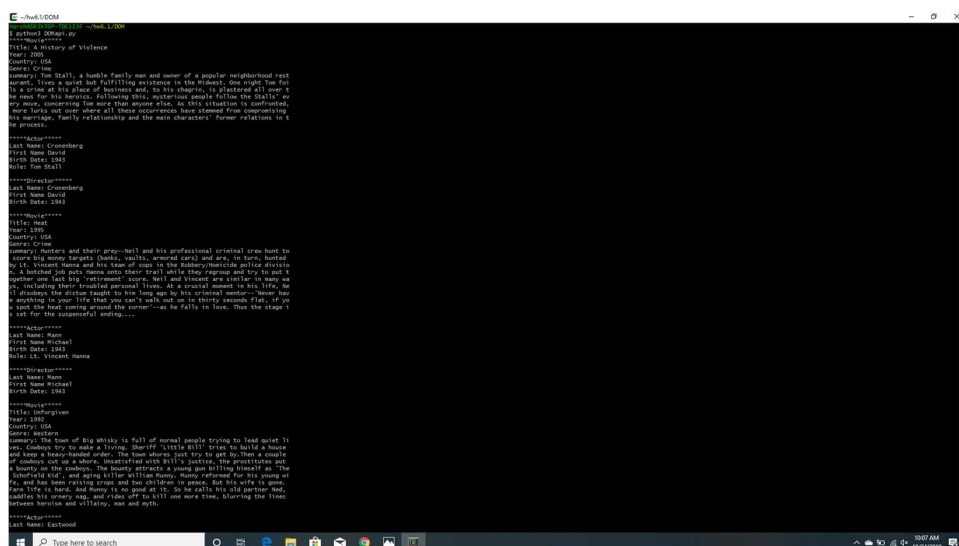
```
Handler = MovieHandler()
```

```
parser.setContentHandler( Handler )
```

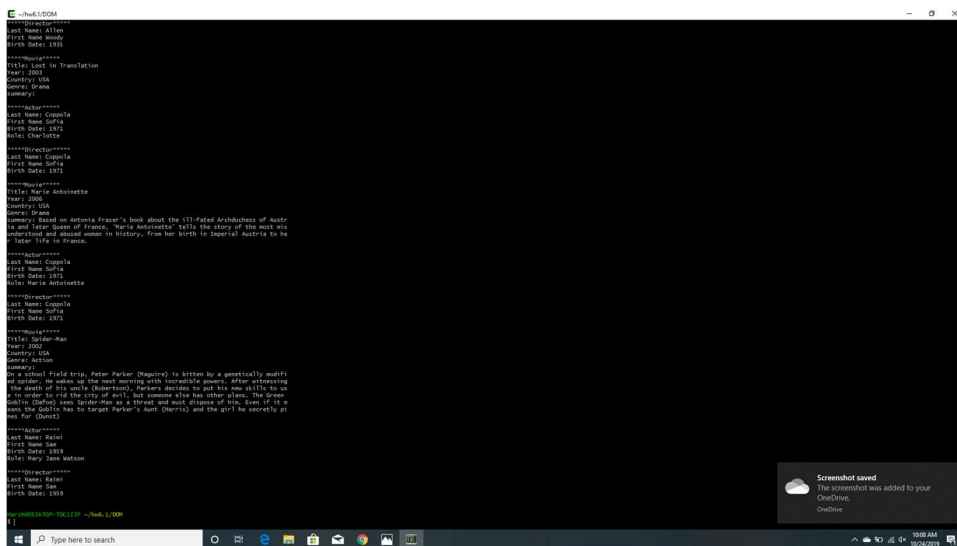
```
parser.parse("movies_long.xml")
```

Q-3) Parsing with DOMApi.py

Execution Result:-



```
C:\Users\DOM>python DOMApi.py
1 python DOMApi.py
2 movies_long.xml
3 Title: A History of Violence
4 Year: 2005
5 Country: USA
6 Genre: Crime
7 Summary: Tom Stall, a humble family man and member of a popular neighborhood rest
8aurant, lives a quiet and fulfilling existence in the Midwest, one night Tom has
9to a crime at his place of business and, to his chagrin, is charged with murder.
10He goes for his lawyer, following this, mother-in-law people follow the trails' re
11try force, uncovering the one that seemed right. As this situation is unfolded,
12more facts are seen where all these occurrences have stemmed from compensating
13his marriage, family relationships and the main characters' former relations in t
14he process.
15
16=====
17Actor:
18Last Name: Cronenberg
19First Name: David
20Birth Date: 1945
21Role: Tom Stall
22
23=====
24Director:
25Last Name: Cronenberg
26First Name: David
27Birth Date: 1945
28
29=====
30Movie:
31Title: Heat
32Year: 1995
33Country: USA
34Genre: Crime
35Summary: Hunters and their prey--hell and his professional criminal crew hunt to
36score big money targets (banks, jewelry, armored cars) and are, in turn, hunted
37by Lt. Vincent Hanna and his team of cops in the money/battle police division.
38A holdout job puts Hanna onto their trail while they regroup and try to put a
39quadruple on just the retirement home. But and discover he's really as much as
40a hit, including their troubled personal lives. At a crucial moment in his life, he
41finds the director caught in the long run to his criminal career--there's no
42anything in your life that you can't walk out on in thirty seconds flat, if you
43lose the heat coming around the corner--as he falls in love. Thus the stage is
44a set for the suspenseful ending....
45
46=====
47Actor:
48Last Name: Name
49First Name: Michael
50Birth Date: 1963
51Role: Lt. Vincent Hanna
52
53=====
54Director:
55Last Name: Name
56First Name: Michael
57Birth Date: 1963
58
59=====
60Movie:
61Title: Unforgotten
62Year: 1995
63Country: USA
64Genre: Mystery
65Summary: The town of Big Whiskey is full of normal people trying to lead quiet li
66ves. Cowboys try to make a living. Sheriff 'Little Bill' tries to build a house
67and keep a beer-hound away. The town sheriff just try to get the town a couple
68of cowboys out of a whore. Unintentionally with Bill's justice, the prostitute put
69a bounty on the cowboy. The bounty attracts a young gun killing himself as 'The
70Sheffield Kid', and young killer William Munny referred for his young wi
71fe, and has been raising cows and has children in prison. But his wife is gone.
72The life is hard, and Munny is in good of it. So he kills his old partner, who
73labeled his money bag, and rides off to kill one more time. Sharing the time
74between justice and villainy, man and myth.
75
76=====
77Last Name: Sanford
```



DOMApi.py:-

#!/usr/bin/python

```
from xml.dom.minidom import parse
```

```
import xml.dom.minidom
```

```
# Open XML document using minidom parser
```

```
DOMTree = xml.dom.minidom.parse("movies_long.xml")
```

```
collection = DOMTree.documentElement
```

```
# Get all the movies in the collection
```

```
movies = collection.getElementsByTagName("movie")
```

```
# Print detail of each movie.
```

```
for movie in movies:
```

```
    print ("*****Movie*****")
```

```
    title = movie.getElementsByTagName('title')[0]
```

```
    print ("Title: %s" % title.childNodes[0].data)
```

```
    year = movie.getElementsByTagName('year')[0]
```

```
    print ("Year: %s" % year.childNodes[0].data)
```

```
    country = movie.getElementsByTagName('country')[0]
```

```
    print ("Country: %s" % country.childNodes[0].data)
```

```
    genre = movie.getElementsByTagName('genre')[0]
```

```
    print ("Genre: %s" % genre.childNodes[0].data)
```

```
    summary = movie.getElementsByTagName('summary')[0]
```

```
    print ("summary: %s" % summary.childNodes[0].data + "\n")
```

```
    if movie.getElementsByTagName("actor"):
```

```
        print ("*****Actor*****")
```

```
        last_name = movie.getElementsByTagName('last_name')[0]
```

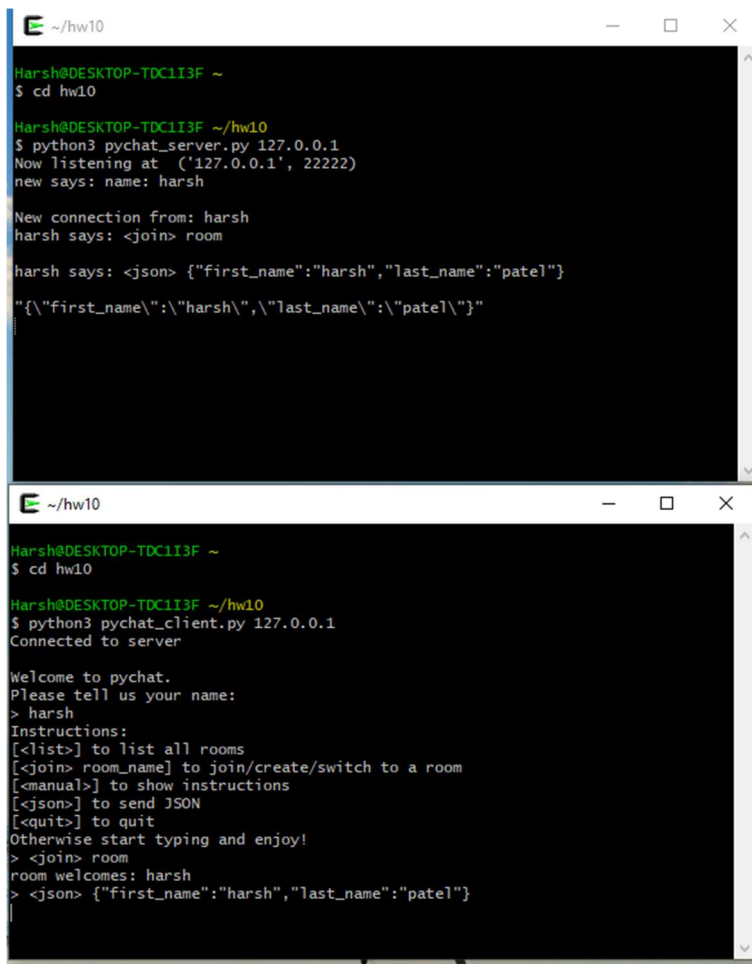


```
print ("Last Name: %s" % last_name.childNodes[0].data)
first_name = movie.getElementsByTagName('first_name')[0]
print ("First Name %s" % first_name.childNodes[0].data)
birth_date = movie.getElementsByTagName('birth_date')[0]
print ("Birth Date: %s" % birth_date.childNodes[0].data)
role = movie.getElementsByTagName('role')[0]
print ("Role: %s" % role.childNodes[0].data + "\n")
```

```
if movie.getElementsByTagName("director"):
    print ("*****Director*****")
    last_name = movie.getElementsByTagName('last_name')[0]
    print ("Last Name: %s" % last_name.childNodes[0].data)
    first_name = movie.getElementsByTagName('first_name')[0]
    print ("First Name %s" % first_name.childNodes[0].data)
    birth_date = movie.getElementsByTagName('birth_date')[0]
    print ("Birth Date: %s" % birth_date.childNodes[0].data + "\n")
```

Q-4) pychat + json

Execution Result:-



The image shows two terminal windows side-by-side. The top window is the server terminal, and the bottom window is the client terminal. Both are running on a system named 'Harsh@DESKTOP-TDC1I3F' in the directory '~/hw10'.

Top Terminal (Server):

```
Harsh@DESKTOP-TDC1I3F ~  
$ cd hw10  
Harsh@DESKTOP-TDC1I3F ~/hw10  
$ python3 pychat_server.py 127.0.0.1  
Now listening at ('127.0.0.1', 22222)  
new says: name: harsh  
  
New connection from: harsh  
harsh says: <join> room  
  
harsh says: <json> {"first_name":"harsh","last_name":"patel"}  
{"first_name":"harsh","last_name":"patel"}
```

Bottom Terminal (Client):

```
Harsh@DESKTOP-TDC1I3F ~  
$ cd hw10  
Harsh@DESKTOP-TDC1I3F ~/hw10  
$ python3 pychat_client.py 127.0.0.1  
Connected to server  
  
Welcome to pychat.  
Please tell us your name:  
> harsh  
Instructions:  
[<list>] to list all rooms  
[<join> room_name] to join/create/switch to a room  
[<manual>] to show instructions  
[<json>] to send JSON  
[<quit>] to quit  
Otherwise start typing and enjoy!  
> <join> room  
room welcomes: harsh  
> <json> {"first_name":"harsh","last_name":"patel"}  
|
```

Pychat_client.py:-

import select, socket, sys

from pychat_util import Room, Hall, Player

import pychat_util

READ_BUFFER = 4096

```
if len(sys.argv) < 2:
```

```
    print("Usage: Python3 client.py [hostname]", file = sys.stderr)
```

```
    sys.exit(1)
```

```
else:
```

```
    server_connection = socket.socket(socket.AF_INET,  
    socket.SOCK_STREAM)
```

```
    server_connection.setsockopt(socket.SOL_SOCKET,  
    socket.SO_REUSEADDR, 1)
```

```
    server_connection.connect((sys.argv[1], pychat_util.PORT))
```

```
def prompt():
```

```
    print('>', end=' ', flush = True)
```

```
print("Connected to server\n")
```

```
msg_prefix = "
```

```
socket_list = [sys.stdin, server_connection]
```

```
while True:
```

```
    read_sockets, write_sockets, error_sockets =  
    select.select(socket_list, [], [])
```

```
    for s in read_sockets:
```

```
        if s is server_connection: # incoming message
```

```
            msg = s.recv(READ_BUFFER)
```

```
            if not msg:
```

```
                print("Server down!")
```

```
                sys.exit(2)
```

else:

if msg == pychat_util.QUIT_STRING.encode():

sys.stdout.write('Bye\n')

sys.exit(2)

else:

sys.stdout.write(msg.decode())

if 'Please tell us your name' in msg.decode():

msg_prefix = 'name: ' # identifier for name

else:

msg_prefix = "

prompt()

else:

msg = msg_prefix + sys.stdin.readline()

server_connection.sendall(msg.encode())

Pychat_server.py:-

implementing 3-tier structure: Hall --> Room --> Clients;

14-Jun-2013

import select, socket, sys, pdb

from pychat_util import Hall, Room, Player

import pychat_util

READ_BUFFER = 4096

host = sys.argv[1] if len(sys.argv) >= 2 else "

```
listen_sock = pychat_util.create_socket((host, pychat_util.PORT))
```

```
hall = Hall()
```

```
connection_list = []
```

```
connection_list.append(listen_sock)
```

```
while True:
```

```
    # Player.fileno()
```

```
    read_players, write_players, error_sockets =  
    select.select(connection_list, [], [])
```

```
    for player in read_players:
```

```
        if player is listen_sock: # new connection, player is a socket
```

```
            new_socket, add = player.accept()
```

```
new_player = Player(new_socket)
```

```
connection_list.append(new_player)
```

```
hall.welcome_new(new_player)
```

```
else: # new message
```

```
msg = player.socket.recv(READ_BUFFER)
```

```
if msg:
```

```
    msg = msg.decode().lower()
```

```
    hall.handle_msg(player, msg)
```

```
else:
```

```
    player.socket.close()
```

```
    connection_list.remove(player)
```



```
for sock in error_sockets: # close error sockets
```

```
    sock.close()
```

```
    connection_list.remove(sock)
```

Pychat_util.py:-

```
#####
```

```
##
```

```
# implementing 3-tier structure:
```

```
#     Hall --> Room --> Clients;
```

```
# 14-Jun-2013
```

```
#####
```

```
##
```

```
import socket, pdb, json
```

```
MAX_CLIENTS = 30
```

```
PORT = 22222
```

```
QUIT_STRING = '<$quit$>'
```

```
def create_socket(address):
```

```
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
s.setblocking(0)
s.bind(address)
s.listen(MAX_CLIENTS)
print("Now listening at ", address)
return s
```

```
class Hall:
```

```
    def __init__(self):
```

```
        self.rooms = {} # {room_name: Room}
```

```
        self.room_player_map = {} # {playerName: roomName}
```

```
    def welcome_new(self, new_player):
```

```
        new_player.socket.sendall(b'Welcome to pychat.\nPlease tell us  
your name:\n')
```

```
    def list_rooms(self, player):
```

```
        if len(self.rooms) == 0:
```

```
            msg = 'Oops, no active rooms currently. Create your own!\n' \
```

```
                + 'Use [<join> room_name] to create a room.\n'
```

```
            player.socket.sendall(msg.encode())
```

```
else:
    msg = 'Listing current rooms...\n'
    for room in self.rooms:
        msg += room + ": " + str(len(self.rooms[room].players)) + "
player(s)\n"
    player.socket.sendall(msg.encode())
```

```
def handle_msg(self, player, msg):
    instructions = b'Instructions:\n\
    + b' [<list>] to list all rooms\n\
    + b' [<join> room_name] to join/create/switch to a room\n\
    + b' [<manual>] to show instructions\n\
    + b' [<json>] to send JSON\n\
    + b' [<quit>] to quit\n\
    + b' Otherwise start typing and enjoy!\
    + b'\n'
```

```
print(player.name + " says: " + msg)
if "name:" in msg:
    name = msg.split()[1]
    player.name = name
    print("New connection from:", player.name)
    player.socket.sendall(instructions)
```

```
elif "<join>" in msg:
    same_room = False
    if len(msg.split()) >= 2: # error check
        room_name = msg.split()[1]
        if player.name in self.room_player_map: # switching?
            if self.room_player_map[player.name] == room_name:
                player.socket.sendall(b'You are already in room: ' +
room_name.encode())
                same_room = True
            else: # switch
                old_room = self.room_player_map[player.name]
                self.rooms[old_room].remove_player(player)
        if not same_room:
            if not room_name in self.rooms: # new room:
                new_room = Room(room_name)
                self.rooms[room_name] = new_room
            self.rooms[room_name].players.append(player)
            self.rooms[room_name].welcome_new(player)
            self.room_player_map[player.name] = room_name
        else:
            player.socket.sendall(instructions)
```

```
elif "<list>" in msg:
```

```
    self.list_rooms(player)
```

```
elif "<manual>" in msg:
```

```
    player.socket.sendall(instructions)
```

```
elif "<json>" in msg:
```

```
    if len(msg.split()) >= 2: # error check
```

```
        json_string = msg.split()[1]
```

```
        print(json.dumps(json_string))
```

```
    else:
```

```
        player.socket.sendall(instructions)
```

```
elif "<quit>" in msg:
```

```
    player.socket.sendall(QUIT_STRING.encode())
```

```
    self.remove_player(player)
```

```
else:
```

```
    #####
```

```
    # check if in a room or not first
```

```
    #####
```

```
    if player.name in self.room_player_map:
```

```
self.rooms[self.room_player_map[player.name]].broadcast(player,
msg.encode())
```

```
    else:
```

```
        msg = 'You are currently not in any room! \n' \
              + 'Use [<list>] to see available rooms! \n' \
              + 'Use [<join> room_name] to join a room! \n'
        player.socket.sendall(msg.encode())
```

```
def remove_player(self, player):
```

```
    if player.name in self.room_player_map:
```

```
self.rooms[self.room_player_map[player.name]].remove_player(player
)
```

```
    del self.room_player_map[player.name]
```

```
    print("Player: " + player.name + " has left\n")
```

```
class Room:
```

```
    def __init__(self, name):
```

```
        self.players = [] # a list of sockets
```

```
        self.name = name
```

```
    def welcome_new(self, from_player):
```

```
        msg = self.name + " welcomes: " + from_player.name + '\n'
```

```
        for player in self.players:
```

```
player.socket.sendall(msg.encode())
```

```
def broadcast(self, from_player, msg):
```

```
    msg = from_player.name.encode() + b": " + msg
```

```
    for player in self.players:
```

```
        player.socket.sendall(msg)
```

```
def remove_player(self, player):
```

```
    self.players.remove(player)
```

```
    leave_msg = player.name.encode() + b"has left the room\n"
```

```
    self.broadcast(player, leave_msg)
```

```
class Player:
```

```
    def __init__(self, socket, name = "new"):
```

```
        socket.setblocking(0)
```

```
        self.socket = socket
```

```
        self.name = name
```

```
    def fileno(self):
```

```
        return self.socket.fileno()
```