MINOR- II
NETWORKING – PROJECT

“ALPHA DOWNLOADER && FILE SHARING ”

BATCH : B -5

VRISHTI GAHLAUT	*(12103528)*
DHRUV SINGH	*(12103533)*
SUMIT BANSAL	*(12103530)*
HAMZA KHAN	*(12103540)*

ACKNOWLEDGEMENT

We take this opportunity to express our profound sense of gratitude and respect to all those who helped us throughout the duration of this project. Preparing a project report is never a unilateral effort no matter ultimate credit may go to the author. We wish to acknowledge the guidance and support of the professors and academics in bringing up a real picture of the concept for which the report is prepared.

Our special thanks to our supervisor Ms. ARPITA JADHAV who was so critical in this project and without her we would not have been able to do this.

We also thank all the staff members of JIIT for extending full support and making this whole experience enriching, informative and facilitating the project to reach its current state.

ABSTRACT

With the dependence of today's world on downloading data from the World Wide Web is reaching new heights, new and effective techniques must be developed to satisfy these downloading demands. This project is a step towards that direction.

We present the "Alpha downloader", a new download manager that effectively boosts the downloading speed and provides multi-client cache facilities.

The project is built solely on the python platform and uses the concept of multithreading to increase the downloading speed. Threading is a feature that enables parallel processing i.e. multiple threads execute at the same time. Thus, several packets of data can be downloaded at the same time in contrast to the orthodox downloaders which allow only a single stream of packets at a time. The "Alpha downloader" needs to be fed with the URL of the intended download page and a destination location on the server memory to begin downloading.

Along with speedy downloading, "Alpha downloader" allows multiple clients to be connected to the main server which is also the downloader using IP addresses and port numbers and handles them at the same time. Thus a page needs to be downloaded only once by the server and then any connected client can have access to it, saving the highly expensive Data usage and also saving time for multiple downloads of the same data. Also, if a particular page is already downloaded by the server, then any client requesting the same data gets it in no time as the page only needs to the sent to the client by the server.

This project also contains gaming section for server and client . Chatting option is also available for multi-clients .

Youtube downloader is one of the feature of this project . Youtube downloader is a part of downloader in which we provides a link to download a video , then downloader finds the hign quality of that video and download that high quality video .

OBJECTIVE

The purpose of the project is to build a new age downloader cum server which provides high speed downloading and handles multiple clients at the same time using python as the coding platform and the concept of multithreading and server client communication.

This software is highly useful for local area networks for example a college computer lab where downloading tends to get slow due to high load on the Internet. In such a scenario, “Alpha downloader” will provide high downloading speed. Also, all students can connect to this server, therefore since most of the download requests would of the same page or data on a particular day, hence “Alpha downloader” will have to simple forward the already downloaded page to the requesting client.

Youtube downloader is one of the feature of this project . Youtube downloader is a part of downloader in which we provides a link to download a video , then downloader finds the hign quality of that video and download that high quality video .

Division of project work

Dhruv - client.py, client side commumication and functioning, gaming part , debug and testing.

Sumit - server.py, server side commumication and functioning , debug and testing , youtube downloader .

Vrishti - GUI related complete designing (server and client side) , help in server and client side communication .

Hamza - server.py , Debuging related work , ideas , code testing ,gaming part .

DESIGNING

Language: **Python 2.7 language used for create downloader application and design .**

Platform : **Windows 7**

(Socket based application)

There are 3 python files .

- Server.py
- Client.py
- Game.py

To Run :

On cmd-

```
python server.py
python client.py
```

Background Study and Findings

A **download manager** is a computer program dedicated to the task of downloading (and sometimes uploading) possibly unrelated stand-alone files from (and sometimes to) theInternet for storage. Some download managers can also be used to accelerate download speeds by downloading from multiple sources at once. Although web browsers may have download managers incorporated as a feature, they are differentiated by the fact that they do not prioritize accurate, complete and unbroken downloads of information. While some download managers are fully fledged programs dedicated to downloading any information over one or more protocols (e.g. http), many are integrated into installers or update managers and used to download parts of a specific program (or set of programs).

Download acceleration, also known, as multipart download, is a term for the method employed by software such as download managers to download a single file by splitting it in segments and using several simultaneous connections to download these segments from a single server.

The reason for doing so is to circumvent server side limitations of bandwidth per connection. Because in normal networking situations all individual connections are treated equally, rather than actual file transfers, multiple connections yields an advantage on saturated links over simple connections, both in terms of total bandwidth allocation and resilience. Many servers, however, implement a maximum number of simultaneous connections per client in order to mitigate this.

This is not to be confused with segmented downloading, which allows a client to download segments of a file simultaneously from multiple servers.

Code Implémentation and Testing

Server.py

```
no_of_connections=0

no_of_acceptance=0

start_flag = False;

connect_flag = False;


def fetch():

        os.system("start cmd /C pause")

        os.system("youtube-dl "+ent.get())


root = Tk()

root.title('Alpha-Youtube')

ent = Entry(root,bd =5,width=50)

ent.insert(0, 'Youtube-downloader')

ent.pack(side=TOP, fill=X)

ent.focus()

ent.bind('<Return>', (lambda event: fetch()))

btn = Button(root, text='Fetch Video', command=fetch)

btn.pack(side=RIGHT)
```

```
win = Tk()

win.title('Alpha ++')


def run_thread(a) :

    asyncore.loop()


class GuiPart:

    def __init__(self, master, queue, endCommand):

        self.queue = queue


    def askdirectory(self):


        #Returns a selected directoryname.

        #print (self.E1).get()

        (self.v).set(tkFileDialog.askdirectory(**self.dir_opt))


    def start_download(self):

        #print "here in start"

        global start_flag, should_proceed

        start_flag = True

        should_proceed=True

        #self.start()


    def start_pushed(self):
        if len(self.urlentered.get())!=0 and len(self.v.get())!=0:
            try:
                temp=urllib.urlopen(self.urlentered.get())
            except:
                #print e.code

def threader(self):
        self.thread2 = threading.Thread(target=self.start)
        self.thread2.start()

    def start(self):
        global server

        self.progress["value"] = 0
        self.maxbytes = server.clients_handler[0].sizeToDownload
        self.progress["maximum"] = server.clients_handler[0].sizeToDownload
        #self.read_bytes()

def download(i):
    if i==0:
        start=0
    else:
        start=i*packet_size_each_thread[i-1]
    start1=str(start)
    end=start+packet_size_each_thread[i]-1
    end1=str(end)
    req = urllib2.Request(url, headers={'Range':'bytes='+start1+'-'+end1})
    u = urllib2.urlopen(req)
    block_sz = 8192
    file_size_dl = start
    temp='\0'
    while True:
        data=u.read(block_sz)
        if not data:
            break
        print('%d Fetched %s from %s' % (i,len(data), url))
        temp=temp+data

    temp=temp[1:]
    print "\n\n"
    print len(temp),"\n\n"
    f.seek(start)
    f.write(temp)

def start_parallel():
    #result = Queue.Queue()
    threads = [threading.Thread(target=download, args = (i,)) for i in range(no_of_threads)]
    for t in threads:
        t.start()
    for t in threads:
        t.join()


def handle_connect(self):
        client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        client_socket.connect(('localhost', 8080))


    def write(self,cl_no,data):
        if len(EchoServer.clients_handler)>cl_no:
            EchoServer.clients_handler[cl_no].send(data)
        else:
            print "\nNo client connected\n"


    def getListOfClients(self):
        print "connected clients are : "
        for x in self.clients:
            print self.clients[x]
```

Client.py

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

client_socket.connect(('localhost', 8080))


request_to_download=client_socket.recv(8812)

print request_to_download

print "1.Yes\n0.No"


choice=raw_input()


if choice=='1':

    response="acceptance=1"

else:

    response="acceptance=0"


client_socket.send(response)


if choice=='1':

    client_no=int(client_socket.recv(8812))

no_of_threads=2

#client_no=1
```

```
if choice=='1':


    url = client_socket.recv(8812)

    start_from=int(client_socket.recv(8812))

    packet_size_each_client=int(client_socket.recv(8812))


    print "\n\n------amount to download= ",packet_size_each_client,"--------\n\n"


    packet_size_each_thread=[]

    i=0

    while i<no_of_threads:

        if i != no_of_threads-1:

            packet_size_each_thread.append(packet_size_each_client/no_of_threads)

        else:

            packet_size_each_thread.append((packet_size_each_client/no_of_threads) +
(packet_size_each_client%no_of_threads))

        i+=1


    #print packet_size_each

    u = urllib2.urlopen(url)


    main_data=[]

    main_data.append([])

    main_data.append([])
```

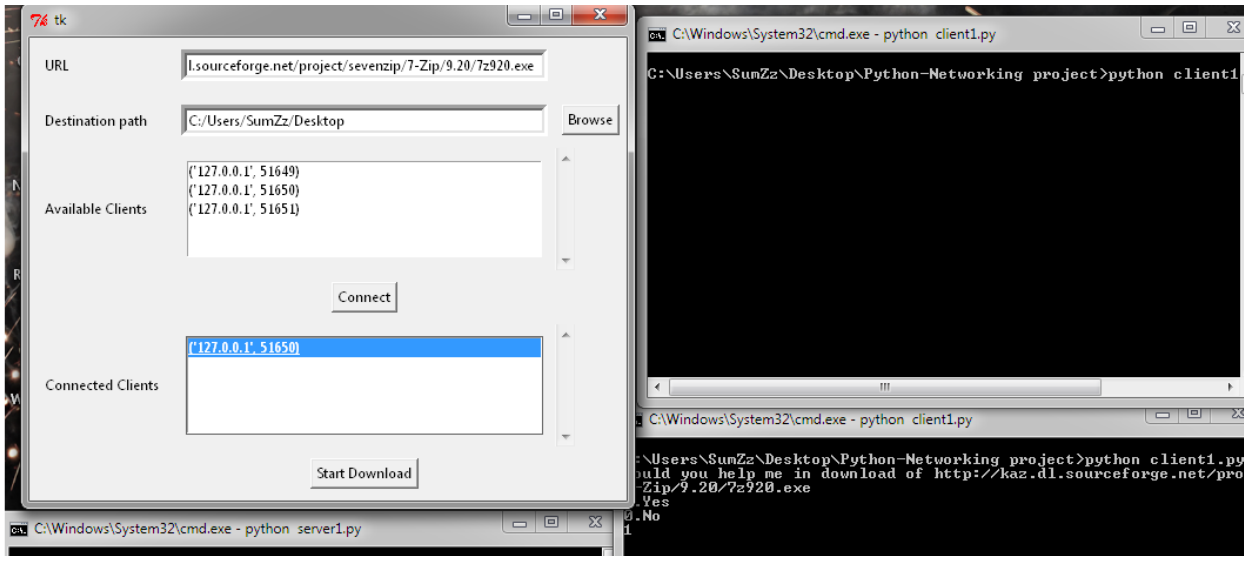
Game.py

```
import pygame

from pygame.locals import *

from sys import exit

import random


pygame.init()


screen=pygame.display.set_mode((640,480),0,32)

pygame.display.set_caption("Pong Pong!")


#Creating 2 bars, a ball and background.

back = pygame.Surface((640,480))

background = back.convert()

background.fill((0,0,0))

bar = pygame.Surface((10,50))

bar1 = bar.convert()

bar1.fill((0,0,255))

bar2 = bar.convert()

bar2.fill((255,0,0))

circ_sur = pygame.Surface((15,15))

circ = pygame.draw.circle(circ_sur,(0,255,0),(15/2,15/2),15/2)
```

```
circle = circ_sur.convert()

circle.set_colorkey((0,0,0))


# some definitions

bar1_x, bar2_x = 10. , 620.

bar1_y, bar2_y = 215. , 215.

circle_x, circle_y = 307.5, 232.5

bar1_move, bar2_move = 0. , 0.

speed_x, speed_y, speed_circ = 250., 250., 250.

bar1_score, bar2_score = 0,0

#clock and font objects

clock = pygame.time.Clock()

font = pygame.font.SysFont("calibri",40)


while True:


    for event in pygame.event.get():

        if event.type == QUIT:

            exit()

        if event.type == KEYDOWN:

            if event.key == K_UP:

                bar1_move = -ai_speed

            elif event.key == K_DOWN:

                bar1_move = ai_speed
        elif event.type == KEYUP:

            if event.key == K_UP:

                bar1_move = 0.

            elif event.key == K_DOWN:

                bar1_move = 0.


    score1 = font.render(str(bar1_score), True,(255,255,255))

    score2 = font.render(str(bar2_score), True,(255,255,255))


    screen.blit(background,(0,0))

    frame = pygame.draw.rect(screen,(255,255,255),Rect((5,5),(630,470)),2)

    middle_line = pygame.draw.aaline(screen,(255,255,255),(330,5),(330,475))

    screen.blit(bar1,(bar1_x,bar1_y))

    screen.blit(bar2,(bar2_x,bar2_y))

    screen.blit(circle,(circle_x,circle_y))

    screen.blit(score1,(250.,210.))

    screen.blit(score2,(380.,210.))


    bar1_y += bar1_move


# movement of circle

    time_passed = clock.tick(30)

    time_sec = time_passed / 1000.0
    circle_x += speed_x * time_sec
    circle_y += speed_y * time_sec
    ai_speed = speed_circ * time_sec
#AI of the computer.
    if circle_x >= 305.:
        if not bar2_y == circle_y + 7.5:
            if bar2_y < circle_y + 7.5:
                bar2_y += ai_speed
            if  bar2_y > circle_y - 42.5:
                bar2_y -= ai_speed
        else:
            bar2_y == circle_y + 7.5


    if bar1_y >= 420.: bar1_y = 420.
    elif bar1_y <= 10. : bar1_y = 10.
    if bar2_y >= 420.: bar2_y = 420.
    elif bar2_y <= 10.: bar2_y = 10.
#since i don't know anything about collision, ball hitting bars goes like this.
    if circle_x <= bar1_x + 10.:
        if circle_y >= bar1_y - 7.5 and circle_y <= bar1_y + 42.5:
            circle_x = 20.
            speed_x = -speed_x
    if circle_x >= bar2_x - 15.:
        if circle_y >= bar2_y - 7.5 and circle_y <= bar2_y + 42.5:
```

```
        circle_x = 605.

        speed_x = -speed_x

if circle_x < 5.:

    bar2_score += 1

    circle_x, circle_y = 320., 232.5

    bar1_y,bar_2_y = 215., 215.

elif circle_x > 620.:

    bar1_score += 1

    circle_x, circle_y = 307.5, 232.5

    bar1_y, bar2_y = 215., 215.

if circle_y <= 10.:

    speed_y = -speed_y

    circle_y = 10.

elif circle_y >= 457.5:

    speed_y = -speed_y

    circle_y = 457.5


pygame.display.update()
```

SNAPSHOTS

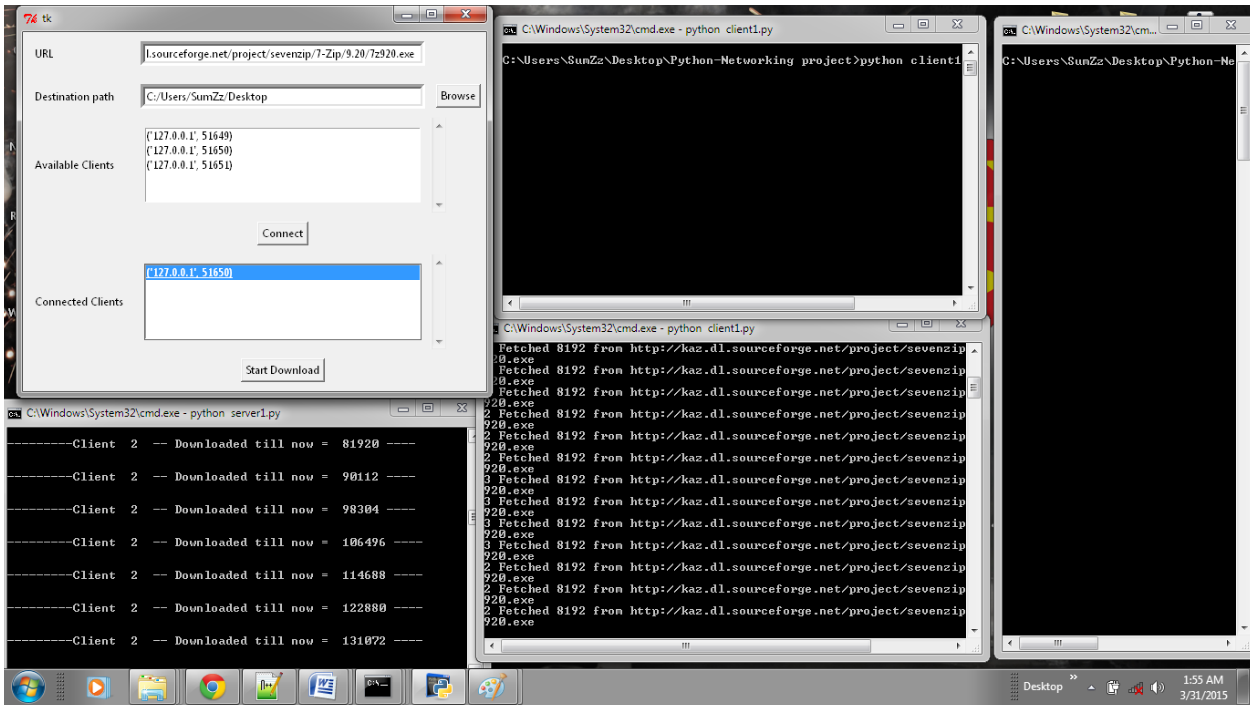


```
URL: l.sourceforge.net/project/sevenzip/7-Zip/9.20/7z920.exe
Destination path: C:/Users/SumZz/Desktop
Available Clients:
('127.0.0.1', 51649)
('127.0.0.1', 51650)
('127.0.0.1', 51651)
Connected Clients:
('127.0.0.1', 51650)
```

```
C:\Windows\System32\cmd.exe - python  client1.py
C:\Users\SumZz\Desktop\Python-Networking project>python client1
```

```
C:\Windows\System32\cmd.exe - python  client1.py
:\Users\SumZz\Desktop\Python-Networking project>python client1.py
ould you help me in download of http://kaz.dl.sourceforge.net/pro
-Zip/9.20/7z920.exe
.Yes
0.No
1
```

```
C:\Windows\System32\cmd.exe - python  server1.py
```



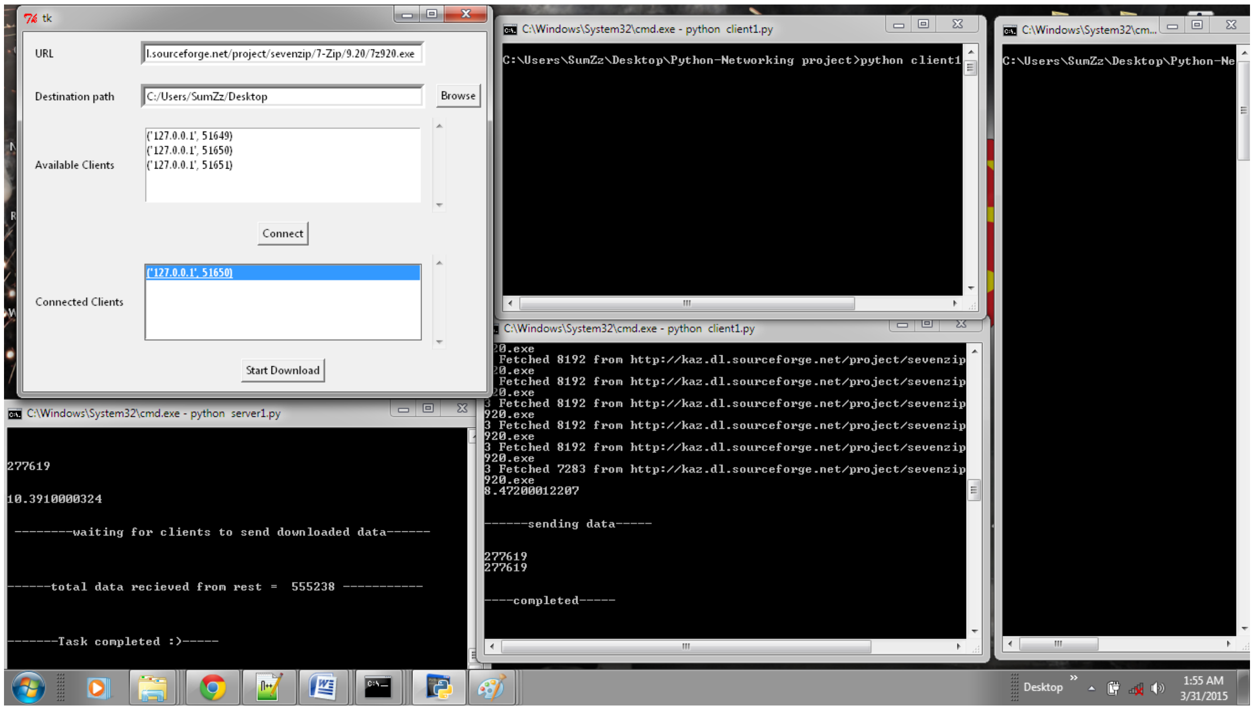
```
URL: l.sourceforge.net/project/sevenzip/7-Zip/9.20/7z920.exe
Destination path: C:/Users/SumZz/Desktop
Available Clients:
('127.0.0.1', 51649)
('127.0.0.1', 51650)
('127.0.0.1', 51651)
Connected Clients:
('127.0.0.1', 51650)
```

```
C:\Windows\System32\cmd.exe - python  server1.py
---------Client  2  -- Downloaded till now =  81920 ----

---------Client  2  -- Downloaded till now =  90112 ----

---------Client  2  -- Downloaded till now =  98304 ----

---------Client  2  -- Downloaded till now =  106496 ----

---------Client  2  -- Downloaded till now =  114688 ----

---------Client  2  -- Downloaded till now =  122880 ----

---------Client  2  -- Downloaded till now =  131072 ----
```

```
C:\Windows\System32\cmd.exe - python  client1.py
C:\Users\SumZz\Desktop\Python-Networking project>python client1
```

```
C:\Windows\System32\cm...
C:\Users\SumZz\Desktop\Python-Ne
```

```
C:\Windows\System32\cmd.exe - python  client1.py
 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
20.exe
 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
20.exe
 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
920.exe
2 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
920.exe
2 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
920.exe
2 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
920.exe
3 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
920.exe
3 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
920.exe
3 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
920.exe
3 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
920.exe
2 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
920.exe
2 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
920.exe
2 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
920.exe
```



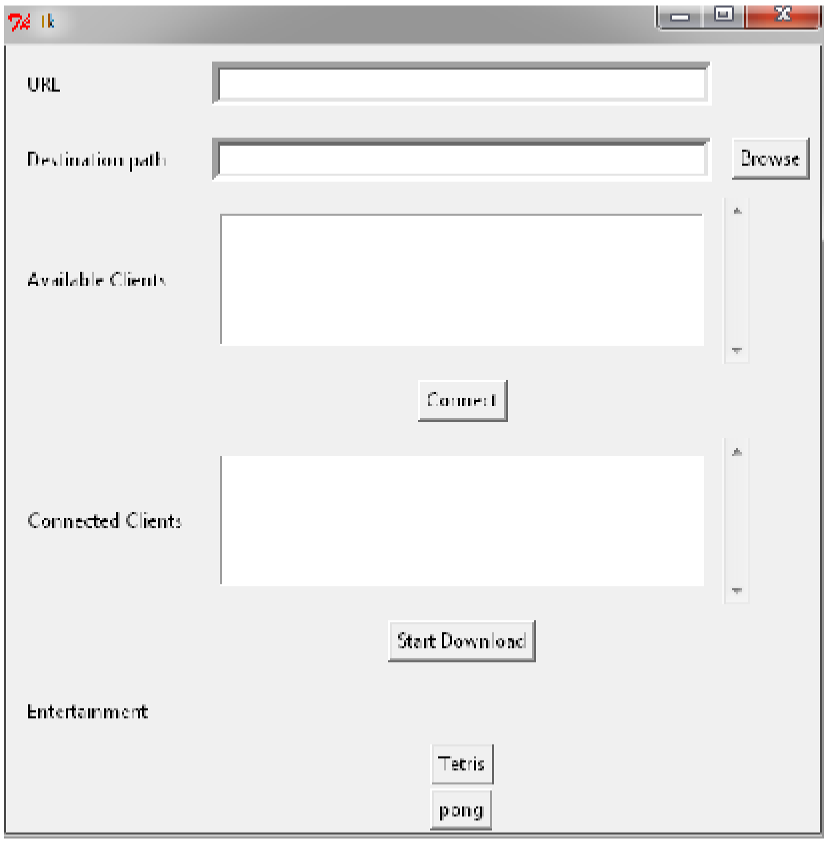
```
URL               l.sourceforge.net/project/sevenzip/7-Zip/9.20/7z920.exe
Destination path  C:/Users/SumZz/Desktop
Available Clients ('127.0.0.1', 51649)
                  ('127.0.0.1', 51650)
                  ('127.0.0.1', 51651)
Connected Clients ('127.0.0.1', 51650)
```

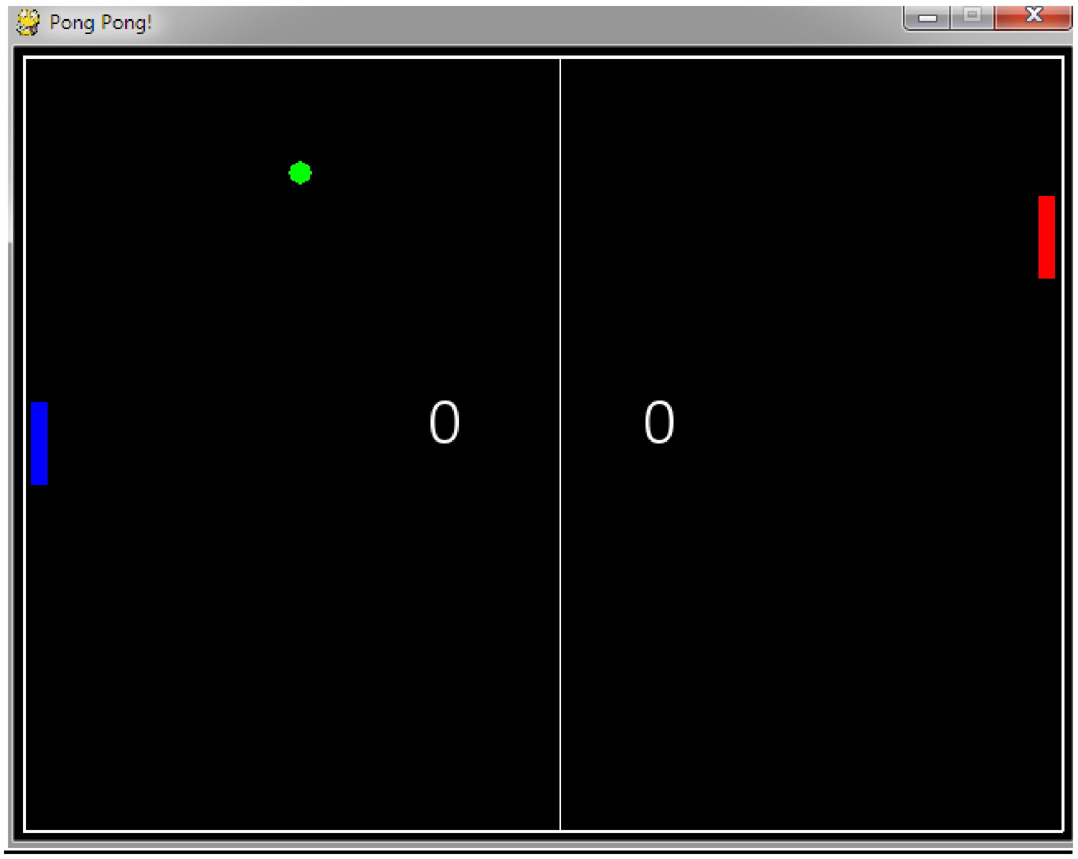
```
C:\Windows\System32\cmd.exe - python  client1.py
C:\Users\SumZz\Desktop\Python-Networking project>python client1
```

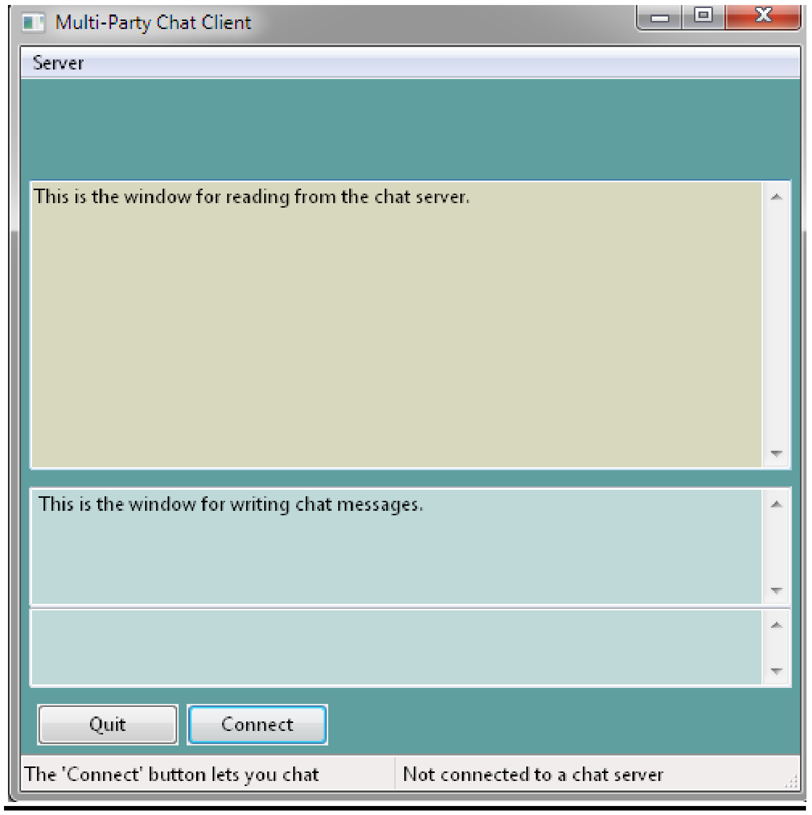
```
C:\Windows\System32\cm...
C:\Users\SumZz\Desktop\Python-Ne
```

```
C:\Windows\System32\cmd.exe - python  client1.py
20.exe
 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
20.exe
 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
20.exe
3 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
920.exe
3 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
920.exe
3 Fetched 8192 from http://kaz.dl.sourceforge.net/project/sevenzip
920.exe
3 Fetched 7283 from http://kaz.dl.sourceforge.net/project/sevenzip
920.exe
8.47200012207


------sending data-----


277619
277619


----completed-----
```

```
C:\Windows\System32\cmd.exe - python  server1.py

277619

10.3910000324

 --------waiting for clients to send downloaded data------


-----total data recieved from rest =  555238 ----------


-------Task completed :)-----
```







This is the window for reading from the chat server.

This is the window for writing chat messages.

The 'Connect' button lets you chat

Not connected to a chat server

References

- Wikipedia .
- Research paper related to downloader .
- Python documentation file and tutorials .
- Stack over flow problem discussions.
- Downloader working .
- Youtube.com