컴퓨터 네트워크
과제6
성능측정
201402329 김수현

1. 주제
    a. 웹 성능 테스트
2. 코드 설명
    a. server
        i. createSocket

```
def createSocket(self):

        server_socket = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
        try:
            server_socket.bind((self.ip_address,self.port_number))
        except Exception as e:
            print("bind error",e)
            server_socket.close()
            return

        print("server socekt open...")
        print("listening...")
        server_socket.listen(5)

        print("Connected with client")
        while True:
           client_socket,address = server_socket.accept()
           threading.Thread(target=self.data_send_recv, args=(client_socket,address)).start()
```

- 소켓을 열어 클라이언트를 받는다. 서버는 소켓을 열어 스레드를 통하여 데이터를 주고받는다.

        ii. parsing_request

```
def parsing_request_header(self,data):

    data_str = str(data.decode('utf-8'))
    split_data = data_str.split('\r\n')
    get_url = split_data[0].split()
    print("headerrequest",get_url)
    try:
        url = get_url[1]
        """
        if len(url[22:]) > 0:
            info = self.get_mail_info(url)
            result = self.send_mail(info)
            return 'mail_send_ok.html'
        """
        url = url.replace('?','')
       # print("url",url)
       # print(url)
        return url
    except Exception as e:
        print(e)
```

- 클라이언트로부터 얻은 데이터를 파싱한다.

- \r\n을 통하여 데이터를 헤더와 데이터로 나눈다.
- split_data[0]을 통하여 클라이언트가 원하는 url을 받고 그값을 반환해준다..

iii. data_send_recv

```
def data_send_recv(self,socket,address):

    data = socket.recv(5000)
    request_url = self.parsing_request_header(data)
    if request_url is not None:

        self.make_response_header(request_url,socket)
    else:
        pass
    socket.close()
    time.sleep(5)
```

- 파싱한 헤더를 통하여 url을 받는다.
- 만약에 얻은 url이 None이 아니면 요청한 데이터를 클라이언트에게 보낼 준비를 한다.

iv. make_response_header

```
def make_response_header(self,url,socket):

    if url[0] == '/':
        url = '.'+url
    else:
        url = './'+url
    print(url)
    if os.path.isfile(url):
        print("there is fileE",url)
        is_first = True
        f = open(url, "rb")

        while True:
            file_data = f.read(1024)

            if len(file_data) == 0:
                break
            content_length = str(len(file_data))
            if is_first:
                request_msg = self.FILE_OK+"\n"
                request_msg = request_msg.encode()+file_data
                is_first = False
                socket.send(request_msg)
            else:
                request_msg=file_data
                socket.send(request_msg)

        f.close()

    else:
        print('nofile')
        socket.send(self.FILE_NO.encode())
```

- 요청한 url을 소켓에다 보내준다.

- 만약에 is_first이면 헤더와 함께 데이터 1024바이트 만큼 보내준다.
- is_first가 아니면 데이터 1024만 파일로부터 읽어서 보내준다.
- 모두 보내면 socket send를 한다.

b. client

i. tcp_connect

```
def tcp_connect(self,execute_num):
    client_socket = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    client_socket.connect((self.server_IP,self.server_port))

    print("Connet to Sever...")
    self.start_time = time.time()
    http_header = self.make_http_header(self.request_url,self.http_method)
    client_socket.send(http_header.encode())

    print("Send Message to Server : ",http_header)
    recv_data = client_socket.recv(1024)

    if recv_data in b'HTTP/1.1 404 Not Found\r\n':
        print("404 Not Found  ERROR : Not Found File:",self.request_url)
    else:
        file_name = self.download_file(client_socket,recv_data)

        if file_name is not None:
            img_list = self.parseHTML(file_name)
            self.send_data(img_list)
    self.end_time = time.time()

    result = self.end_time - self.start_time
    self.write_performance(execute_num,result)
```

- 먼저 서버에 소켓을 연결하고, 시작 시간을 잰다.
- request_header를 만들어 서버에 보낸다.
- 서버에. 응답을받고, 받은파일에 대하여 download를한다.
- filename에대한 요청이 제대로 왔으면
- Html을 파싱하여 다운로드에 필요한 이미지 파일들을 파싱하여 리스트에 놓는다
- 받은 데이터에대한 리스트를 다시 서버에 요청한다..

ii. send_data

```
def send_data(self,lists):
    while len(lists) is not 0:
        url = lists.pop()
        url = url['src']
        print('url',url[0:2])
        if url[0:2] =='./':
            url = url[2:]
        print('url change',url)
        header = self.http_method+' '+url+' '+'HTTP/1.1'
        socket = self.make_connect_with_server()
        socket.send(header.encode())


        receive_data = socket.recv(1024)
        print(receive_data)
        self.download_file(socket,receive_data)
    self.end_time = time.time()
```

- 이미지 리스트를 받으면 이미지에대한것을 요청하는 함수이다.
- 저장된 Url에서 하나씩 꺼내서 만약에 ./라는것이 붙어있으면 떼고, 요청 헤더를 만들어 보낸다.
- 보난후 바로 데이터를 받고, 받은 데이터를 다운로드한다.

iii. make_connect_with_server

```
def make_connect_with_server(self):
    client_socket = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    client_socket.connect((self.server_IP,self.server_port))
    print('img server connect ok')
    return client_socket
```

- 이미지파일 하나를 요청할때마다 클라이언트 소켓을 하나씩 여는 함수이다.

iv. parseHTML

```
def parseHTML(self,url):
    img_list = []
    with open(url,'r') as fp:
        soup = BeautifulSoup(fp,features='lxml')
        img_list = soup.find_all('img')

    return img_list
```

- 다운로드 받은 HTML중 이미지 파일을 스크래핑하는함수이다.

v. downlaod_files

```
def download_file(self,socket,data):
    data_piece =data
    file_name_split = self.request_url.split('.')
    file_name = file_name_split[0]+'_download.'+file_name_split[1]
    count = 0;
    with open(file_name,'wb') as f:
        try:
            while len(data)>0:
                data = socket.recv(1024)
                data_piece += data

            receive_data = data_piece
            data_split = receive_data.split(b'\r\n')
            f.write(data_split[1])
            f.close()
            return file_name

        except Exception as e:
            print("error while download the ",self.request_url,"error - ",e)
            return None
```

- 파일을 다운로드하는 함수이다.
- 더이성 소켓이 받는 데이터가 없을때까지 반복한다.

vi. write_performance

```
def write_performance(self,execute_num,result):
    if execute_num is 10:
        f = open('execute_10process/process_result.csv','a')
    elif execute_num is 20:
        f = open('execute_20process/process_result.csv','a')
    elif execute_num is 30:
        f = open('execute_30process/process_result.csv','a')
    f.write(str(result)+'\n')
    print(result)
```

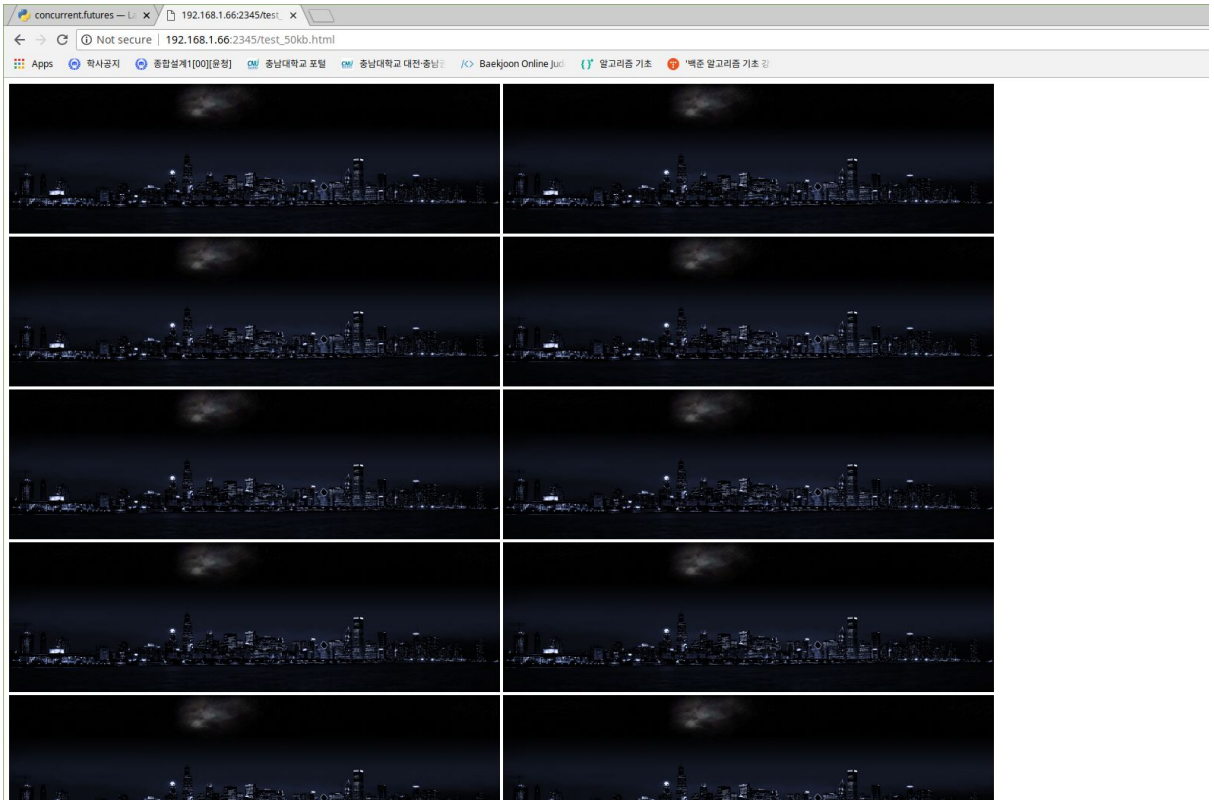
- 이미지 전송이 끝나고난후, processful이 프로세스 몇개를 실행시킬것인가에 따라 다른 파일에다가 쓰도록 하는 함수이다.

vii. main

```
def main(self):
    with concurrent.futures.ProcessPoolExecutor(max_workers=20) as executor:
        for i in range(0,10):
            executor.submit(self.tcp_connect(10))
        for i in range(0,20):
            executor.submit(self.tcp_connect(20))
        for i in range(0,30):
            executor.submit(self.tcp_connect(30))
```

- 프로세스 풀 함수이다. 프로세스를 10개로 했을때와 20개로 했을때, 30개로 했을때 다르게하여 tcp_connect를 호출하여 실행하도록한다.

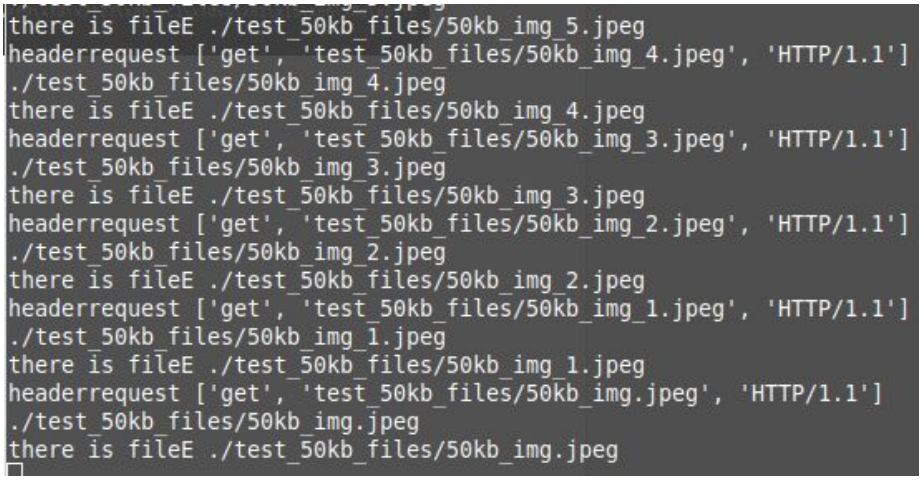
3. 결과 화면
    a. 브라우저



b. 클라이언트

- 클라이언트가 요청을 보냈을때 이미지 파일을 받은 모습이다.

- 서버가 클라이언트에게 받은 request메세지



4.

a. process Pool

i. 프로세스 풀에의해 시간이 기록되어 저장된 파일들

```
kimsoohyun@kimsoohyun-900X5N:~/Desktop/network/project6/client/execute_20process$ cat process_result.csv
0.8864619731903076
0.7596344947814941
0.7717897891998291
0.7721846103668213
0.7966454029083252
0.8187963962554932
0.7683272361755371
0.8194494247436523
0.8152384757995605
0.9362630844116211
0.765042781829834
0.7178633213043213
0.7565581798553467
0.658055305480957
0.683107852935791
0.6764483451843262
0.7432947158813477
0.7238814830780029
0.7892487049102783
0.7658183574676514
```

```
kimsoohyun@kimsoohyun-900X5N:~/Desktop/network/project6/client/execute_30process$ cat process_result.csv
0.7417044639587402
0.763308048248291
0.8887431621551514
0.829714298248291
0.8139421939849854
0.8054678440093994
0.7487084865570068
0.7745394706726074
0.7614829540252686
0.8210930824279785
0.7759079933166504
0.8196582794189453
0.8881635665893555
0.9044351577758789
0.7273166179656982
0.6494574546813965
0.6929600238800049
0.6944215297698975
0.6724526882171631
0.7126684188842773
0.7240169048309326
0.8253467082977295
0.7748572826385498
0.8821620941162109
0.8583462238311768
0.7895455360412598
0.7824764251708984
0.8251338005065918
0.7632114887237549
0.9309647083282471
kimsoohyun@kimsoohyun-900X5N:~/Desktop/network/project6/client/execute_30process$
```

b. pyplot

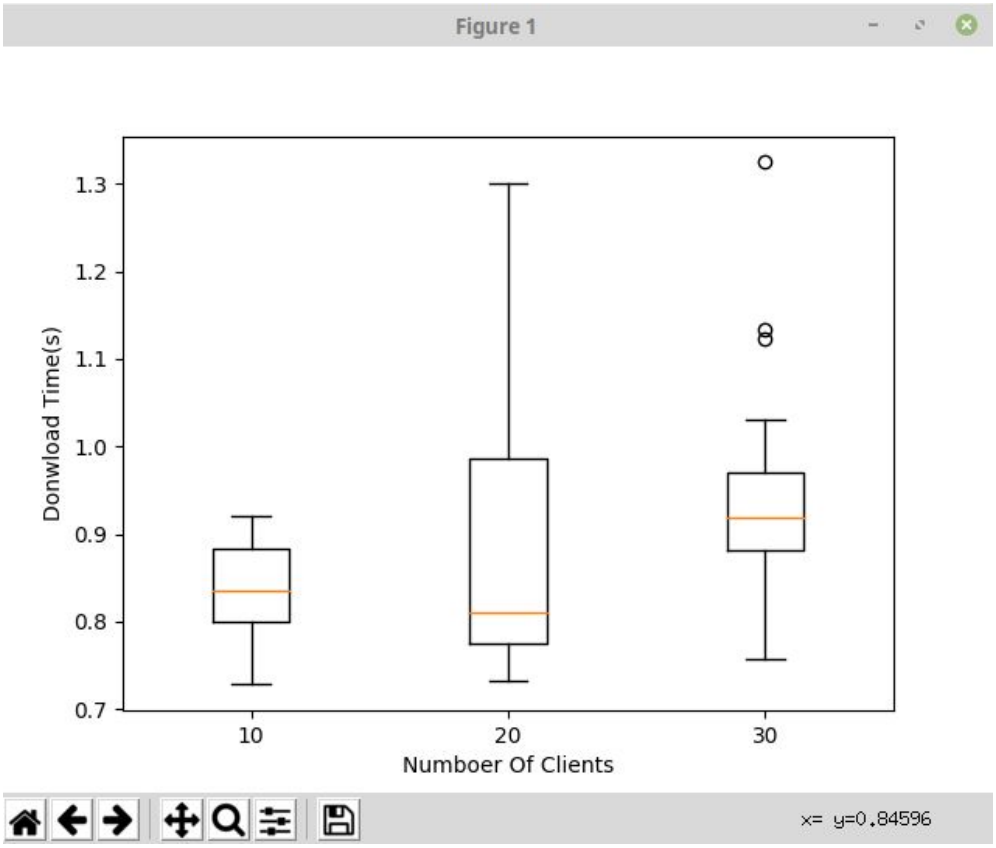


Figure 1
Donwload Time(s)
1.3
1.2
1.1
1.0
0.9
0.8
0.7
10
20
30
Numboer Of Clients
x= y=0.84596