CS118 Report

Shuo BAI(505032786), Cheng MA(105033453)

January 31, 2018

1 Description of the design

In the server design, first we create a socket and bind this socket with a specfic port number. Then we use listen() and accept() to enable this socket to get the request information from this port. And the way we used to process the request information and provide the corresponding response is in the dostuff method.

In the dostuff function, first we parse the request information and get the requested file name. And then we check the current directory to determine whether the desired file is in it. And according to the existence, we add either '404 Not Found' or '200 OK' in the response head. If not in the direcory, we simply return a webpage showing that the desired file is not in the server; Otherwise, we add different content_type to the response head according to the type of the requested file, and also the content of the file. Apart from this, we add other fields in the response head, such as server and date information.

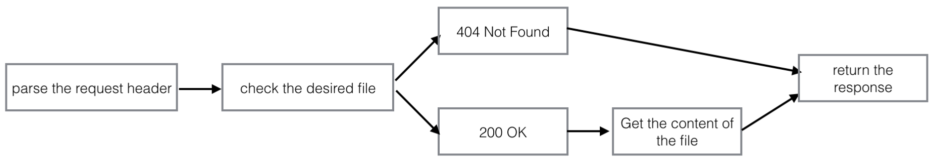


Figure 1: The flow chart of the codes

2 Difficulties

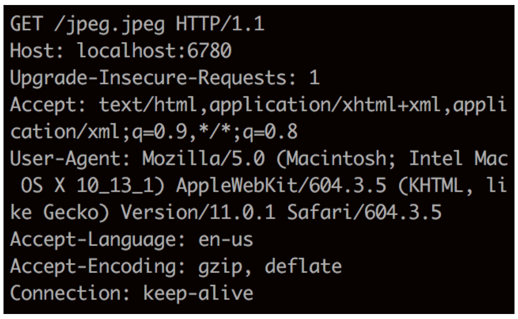
Because we are given the basic frame of this project, we don't have many difficulties. The first difficulty we encounter is that we have to be familiar with the HTTP response format. And the other one is that when we can successfully run our program in the Chrome of macOS, we find it cannot display the right file in the Ubuntu. After checking our codes, we find we do not erase the data in the char array initially so that it will contain some strange characters when running under Ubuntu system. We then use bzero() function to clean the memory so that it can also run well in Ubuntu.

3 Result

3.1 Part A result

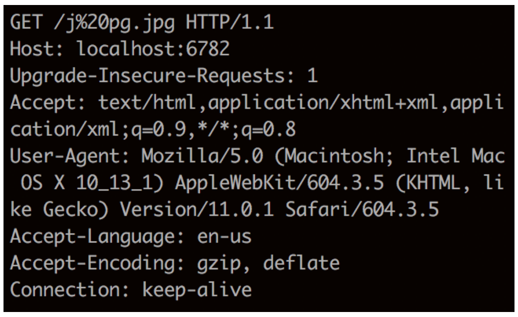
Part A's goal is mainly to display the HTTP request content from Server side. In this project, we pick up html, jpg, jpeg, gif File to send the request from client side. The result is shown as following:

From the Http request we can see that the main difference between different request is the GET URL. Client will try to get those URLs within the Server database, in this case, which is the root folder.



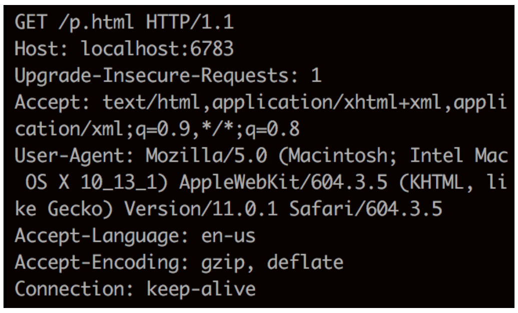
```
GET /jpeg.jpeg HTTP/1.1
Host: localhost:6780
Upgrade-Insecure-Requests: 1
Accept: text/html,application/xhtml+xml,appli
cation/xml;q=0.9,*/*;q=0.8
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac
 OS X 10_13_1) AppleWebKit/604.3.5 (KHTML, li
ke Gecko) Version/11.0.1 Safari/604.3.5
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

(a)



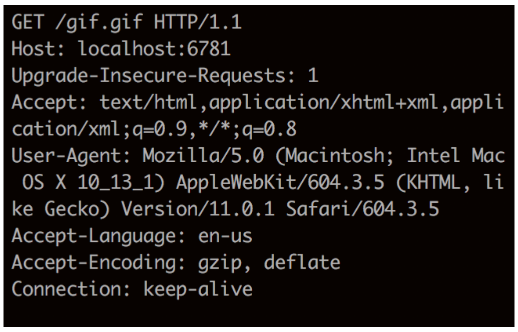
```
GET /j%20pg.jpg HTTP/1.1
Host: localhost:6782
Upgrade-Insecure-Requests: 1
Accept: text/html,application/xhtml+xml,appli
cation/xml;q=0.9,*/*;q=0.8
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac
 OS X 10_13_1) AppleWebKit/604.3.5 (KHTML, li
ke Gecko) Version/11.0.1 Safari/604.3.5
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

(b)



```
GET /p.html HTTP/1.1
Host: localhost:6783
Upgrade-Insecure-Requests: 1
Accept: text/html,application/xhtml+xml,appli
cation/xml;q=0.9,*/*;q=0.8
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac
 OS X 10_13_1) AppleWebKit/604.3.5 (KHTML, li
ke Gecko) Version/11.0.1 Safari/604.3.5
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

(c)



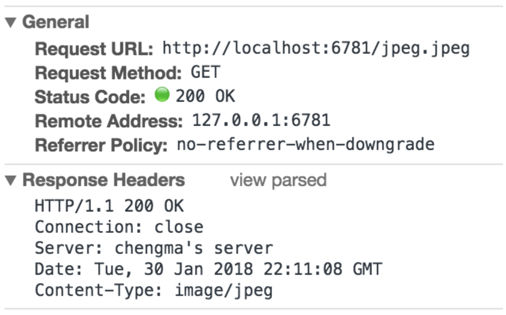
```
GET /gif.gif HTTP/1.1
Host: localhost:6781
Upgrade-Insecure-Requests: 1
Accept: text/html,application/xhtml+xml,appli
cation/xml;q=0.9,*/*;q=0.8
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac
 OS X 10_13_1) AppleWebKit/604.3.5 (KHTML, li
ke Gecko) Version/11.0.1 Safari/604.3.5
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

(d)

Figure 2: HTTP Request.

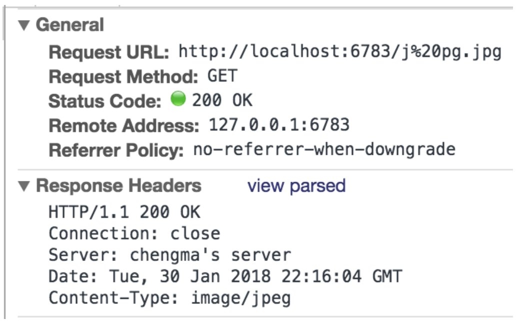
3.2 Part B result

In this part, we should analyze the request to get the URL client needs and give the response.



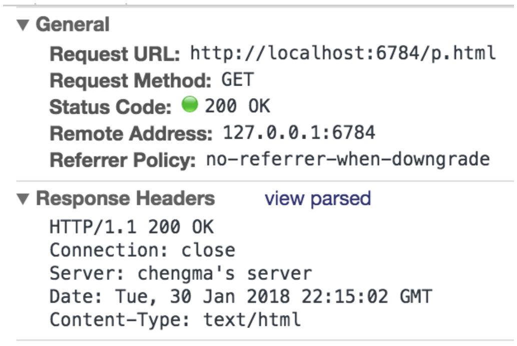
```
▼ General
    Request URL: http://localhost:6781/jpeg.jpeg
    Request Method: GET
    Status Code: ● 200 OK
    Remote Address: 127.0.0.1:6781
    Referrer Policy: no-referrer-when-downgrade
▼ Response Headers      view parsed
    HTTP/1.1 200 OK
    Connection: close
    Server: chengma's server
    Date: Tue, 30 Jan 2018 22:11:08 GMT
    Content-Type: image/jpeg
```

(a)



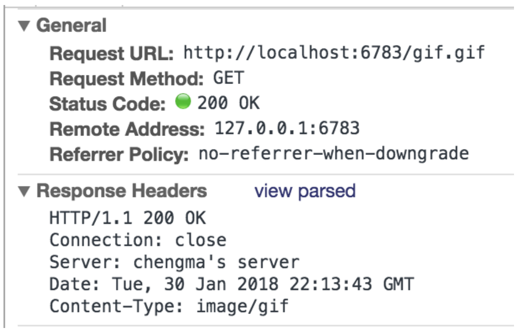
```
▼ General
    Request URL: http://localhost:6783/j%20pg.jpg
    Request Method: GET
    Status Code: ● 200 OK
    Remote Address: 127.0.0.1:6783
    Referrer Policy: no-referrer-when-downgrade
▼ Response Headers      view parsed
    HTTP/1.1 200 OK
    Connection: close
    Server: chengma's server
    Date: Tue, 30 Jan 2018 22:16:04 GMT
    Content-Type: image/jpeg
```

(b)



```
▼ General
    Request URL: http://localhost:6784/p.html
    Request Method: GET
    Status Code: ● 200 OK
    Remote Address: 127.0.0.1:6784
    Referrer Policy: no-referrer-when-downgrade
▼ Response Headers      view parsed
    HTTP/1.1 200 OK
    Connection: close
    Server: chengma's server
    Date: Tue, 30 Jan 2018 22:15:02 GMT
    Content-Type: text/html
```

(c)

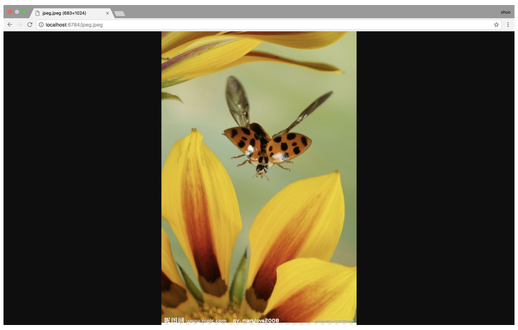


```
▼ General
    Request URL: http://localhost:6783/gif.gif
    Request Method: GET
    Status Code: ● 200 OK
    Remote Address: 127.0.0.1:6783
    Referrer Policy: no-referrer-when-downgrade
▼ Response Headers      view parsed
    HTTP/1.1 200 OK
    Connection: close
    Server: chengma's server
    Date: Tue, 30 Jan 2018 22:13:43 GMT
    Content-Type: image/gif
```

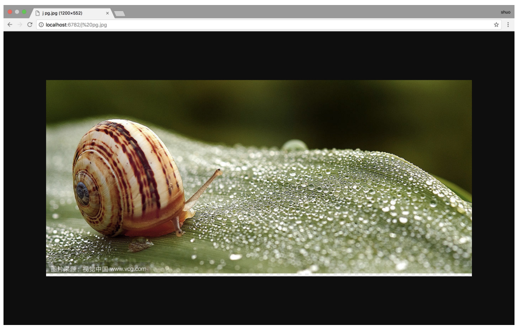
(d)

Figure 3: HTTP Response.

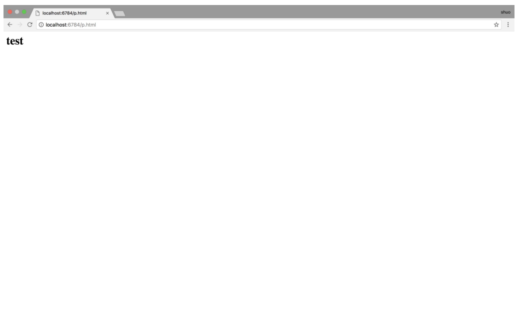
We can see from the response message. Server will resolve the request message to decide the specific content type client request. For example, it will include **jpeg, html, gif** into the response header in this case. After that, it will begin to send binary data file and our test data is shown as following:



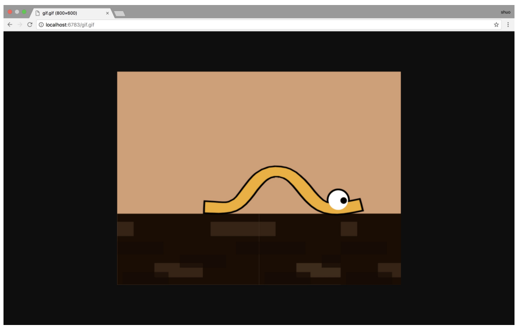
(a)



(b)



(c)



(d)

Figure 4: HTTP Response display in Chrome.

project1 Manual

Include

source_code

> server.c Makefile

test_file

> gif.gif jpeg.jpeg j pg.jpg p.html a b.html

Instrcution

Compile code

type `make` in terminal and you will get a `server` executable file

Use executable file

type `./server $port_number$` in terminal and you can choose any port you want excluding the range(0-1023), after that, you can input `localhost:$port_number$/$file_name$` in any browser you want, the file_name is one of the file in test_file category. You will see the content of files displaying on the website page.