CS 118 Project 1

Concurrent Web Server using BSD Sockets

Connor Fong 303991911

Kai Wang 604406395

Server Design

We used the code provided by the TA as the base for our design, which created the appropriate sockets and bindings needed for connection. Then, it would wait until a connect signal is received, in this case from the web browser, and create the connection.

We coded in the “dostuff” function which parses the request made by the client, and creates an HTTP response with the requested file. If not requested file is found, we would return a “404 Not Found” error page. The file requested is read in binary mode, so that any embedded images in the requested HTML file will be output correctly.

Since we chose to use the some CPLUSPLUS library to read files and handle strings, we had to modify the Makefile provided so that the program would compile using G++, rather than GCC.

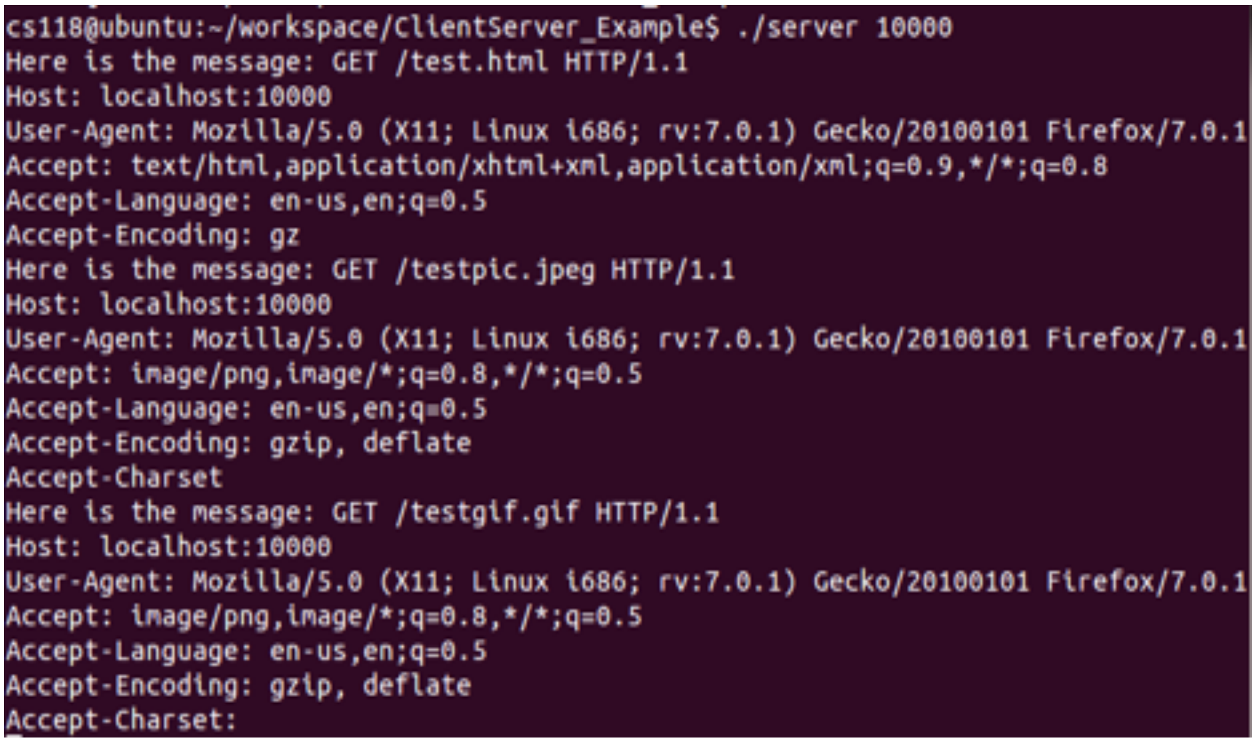
Difficulties met

The main difficulty was generating appropriate response HTTP header. In our code, we included HTTP version, Response state, Date, Content-Type and Content-Length. Generate formatted GMT time cost some effort, we looked into the definition of struct tm and outputted formatted system time in HTTP header.

Manual

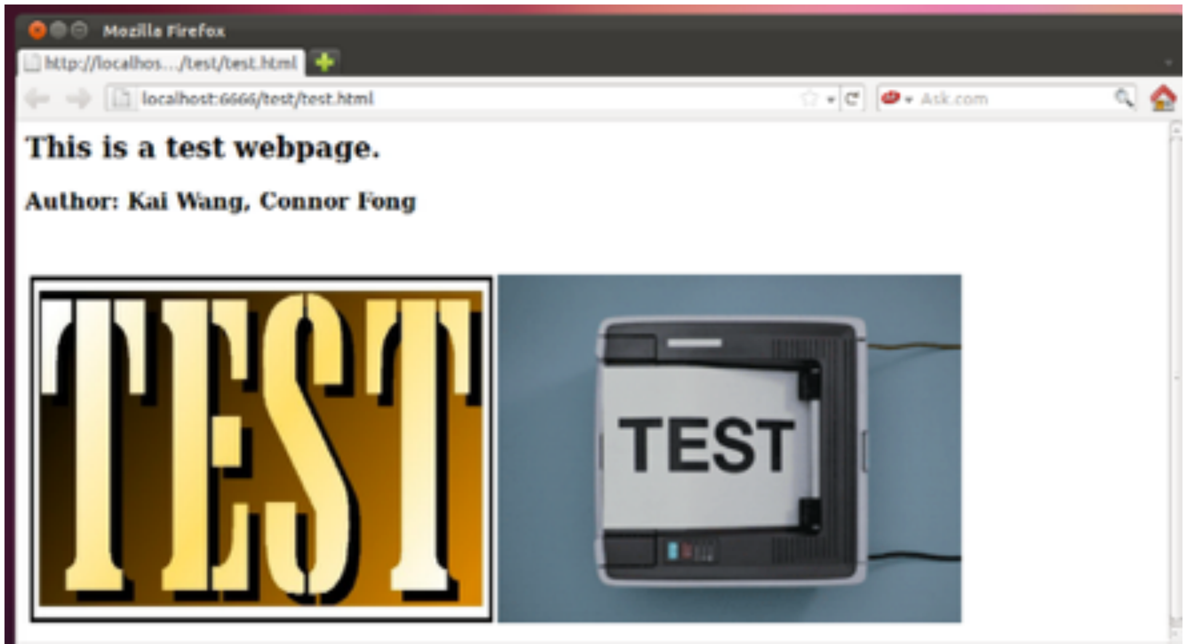
To run the server, you must type “make” in the terminal and allow the program to compile. After it has compiled, it can be run by typing “./serverFork <port number>”, which creates the server at the input port number. The port number must be defined, or the server will not run and an error will be output. Once the server is running, open a web browser and type in the URL “<machine name>:<port number>/test/test.html”. The web page should display a message, a .jpg, and a .gif. You can view a JPEG file independently by replacing “test.html” with “cloud.jpeg”.

Test Cases



```
cs118@ubuntu:~/workspace/ClientServer_Example$ ./server 10000
Here is the message: GET /test.html HTTP/1.1
Host: localhost:10000
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:7.0.1) Gecko/20100101 Firefox/7.0.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gz
Here is the message: GET /testpic.jpeg HTTP/1.1
Host: localhost:10000
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:7.0.1) Gecko/20100101 Firefox/7.0.1
Accept: image/png,image/*;q=0.8,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip, deflate
Accept-Charset
Here is the message: GET /testgif.gif HTTP/1.1
Host: localhost:10000
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:7.0.1) Gecko/20100101 Firefox/7.0.1
Accept: image/png,image/*;q=0.8,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip, deflate
Accept-Charset:
```

In this case, I set up the server in the terminal and connected to it in Mozilla using the process described in the manual. This is the HTTP request found in the console after the connection had been made. Here we can see information about the browser accessing the server, the port number, and shows the files embedded in the HTML code that the server must display.

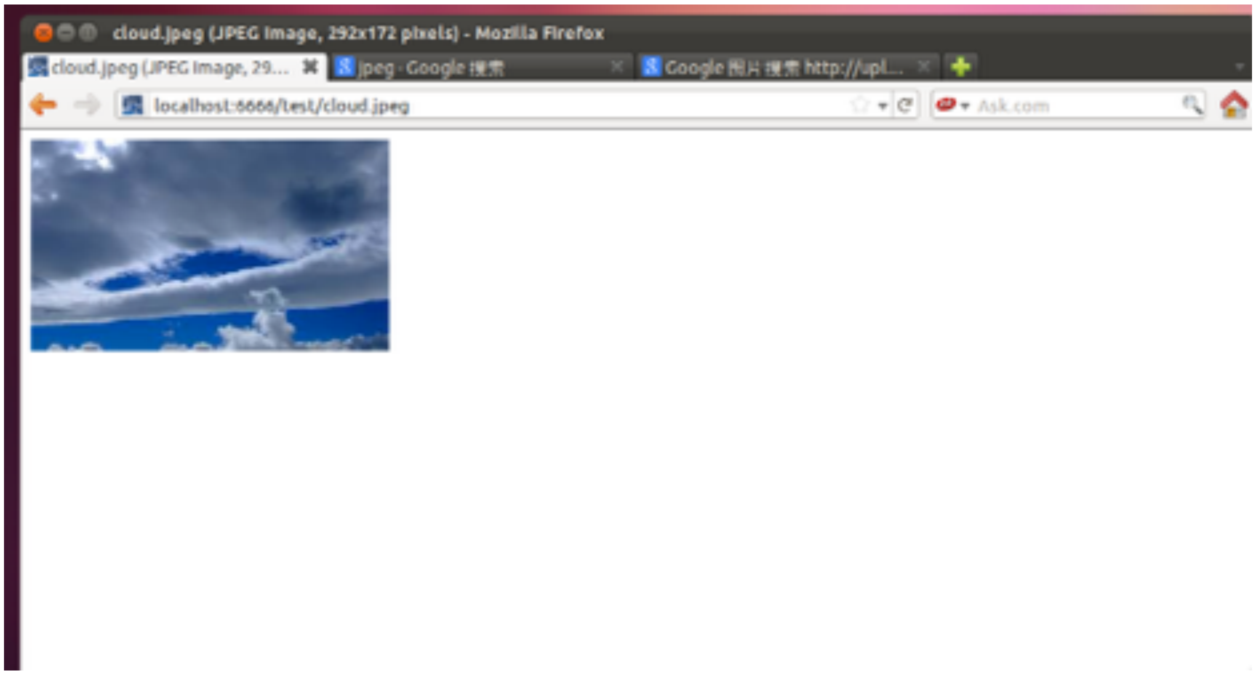


Mozilla Firefox
http://localhos.../test/test.html
localhost:6666/test/test.html
Ask.com
TEST
TEST

This is a test webpage.

Author: Kai Wang, Connor Fong

The above image shows the result of connecting to the server from the Mozilla web browser. This displayed the plain text found in the HTML file, .jpeg image, and .gif file.


cloud.jpeg (JPEG Image, 292x172 pixels) - Mozilla Firefox
cloud.jpeg (JPEG Image, 29...
jpeg - Google 搜索
Google 图片搜索 http://upl...
localhost:6666/test/cloud.jpeg
Ask.com

The above image shows connecting our server from Mozilla firefox, a single jpeg image can be correctly displayed.