

Socket Programming Assignment 2: Web Server

Project Contributors:

Minh Phung and Christopher Blake Matis

CSC 138 Spring 2017

California State University, Sacramento

Objective:

The main objective of this assignment was to design and build a web server in Python that would be at least capable of processing one request for a file. The server creates a connection socket for the client browser when contacted. The HTTP request is then received and parsed to determine what file is being requested. The requested file on the server's file storage is then sent to the client over a TCP connection via a HTTP response message which consists of the requested file preceded by header lines. Of course, if there is a file that is requested which is not available on the server a "404 Not Found" message will be sent.

Design:

Step 1: setup server

- create a tcp socket using port 8000 by default.
- bind the socket to localhost, and whatever port we choose 8000 by default.
- listen for incoming connections or requests.

Step 2: setup http request/response

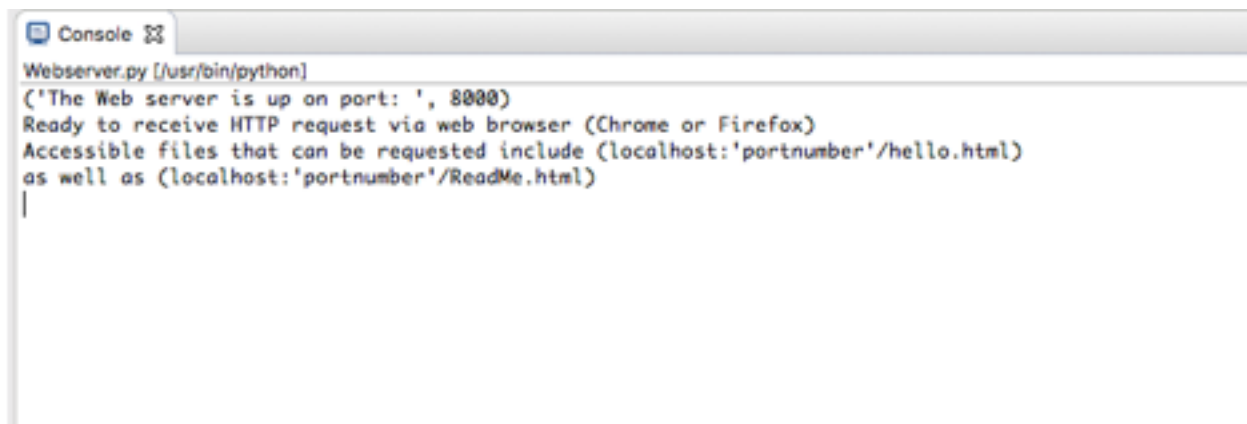
- Embed everything in an infinite while loop
 - print that the server is online and ready for requests.
 - create a connection socket once requested for data communication.
 - get a request from a client.
 - parse the request using built in request split
 - open the requested file and retrieve the data and send that data.

Sending the data:

```
-setup two different response that get encoded.  
    if(file exists on server)  
        send encoded message on the connection  
        socket('http/1.1 200 OK')  
        send the file data to the client browser as well.  
    else  
        send encoded message on the connection  
        socket('404 not found')  
  
close connection.
```

Runtime tests

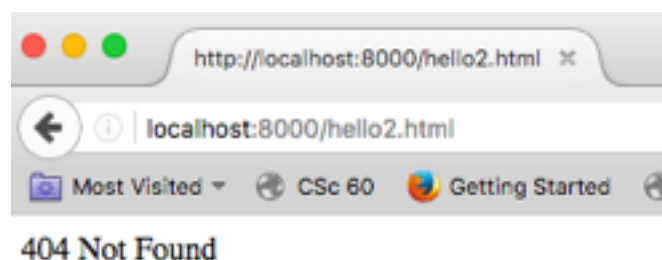
Server when its online:



```
Console [x]
Webserver.py [/usr/bin/python]
('The Web server is up on port: ', 8000)
Ready to receive HTTP request via web browser (Chrome or Firefox)
Accessible files that can be requested include (localhost:'portnumber'/hello.html)
as well as (localhost:'portnumber'/ReadMe.html)
|
```

Access of incorrect file/file not available on the server:

client side:



Server side error message:

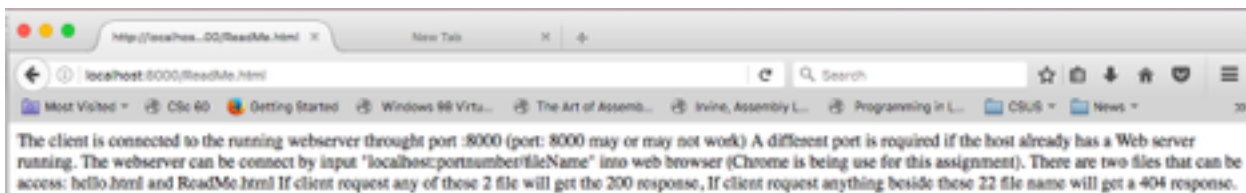
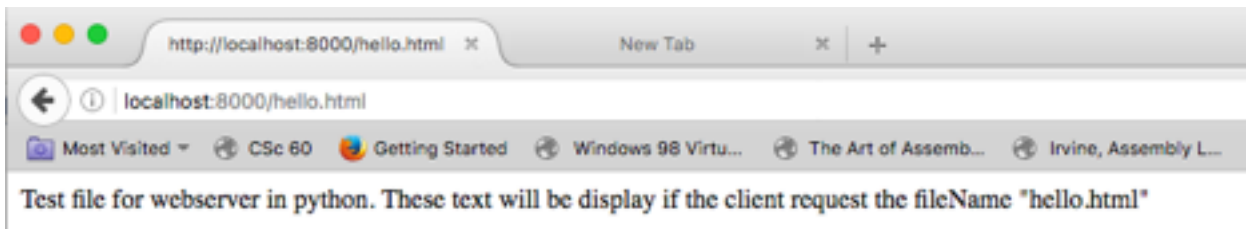
```

terminated> Webserver.py [usr/bin/python]
('The Web server is up on port: ', 8000)
Ready to receive HTTP request via web browser (Chrome or Firefox)
Accessible files that can be requested include (localhost:'portnumber'/hello.html)
as well as (localhost:'portnumber'/ReadMe.html)
['Origin form of request: -> ', 'GET /ReadMe3.html HTTP/1.1\r\nHost: localhost:8000\r\nUser-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:50.0) Gecko/20100101 Firefox/50.0\r\nAccept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\nAccept-Language: en-us\r\nAccept-Encoding: gzip, deflate\r\nConnection: keep-alive\r\nUpgrade-Insecure-Requests: 1\r\n']
404 Not Found
Webserver closed after one request

```

When file is available:

hello.html and ReadMe.html



server side when available:

```

Accessible files that can be requested include (localhost:'portnumber'/hello.html)
as well as (localhost:'portnumber'/ReadMe.html)
['Origin form of request: -> ', 'GET /ReadMe.html HTTP/1.1\r\nHost: localhost:8000\r\nUser-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:50.0) Gecko/20100101 Firefox/50.0\r\nAccept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\nAccept-Language: en-us\r\nAccept-Encoding: gzip, deflate\r\nConnection: keep-alive\r\nUpgrade-Insecure-Requests: 1\r\n']
The client is connected to the running webserver through port :8000 (port: 8000 may or may not work)
A different port is required if the host already has a Web server running.
The webserver can be connect by input "localhost:portnumber/fileName" into web browser (Chrome is being use for this assignment).
There are two files that can be access: hello.html and ReadMe.html
If client request any of these 2 file will get the 200 response,
If client request anything beside these 22 file name will get a 404 response.
Webserver closed after one request

```

