**SEIS752 – Advanced Web Application Development**

**[Spring 2015]**

Professor Lloyd Cledwyn

**HW #2**

**Simple HTTP Server**

Due: Feb 18th, 6pm

**Summary**

In class we built a simple socket interface. We set up a socket server, and had a client connect and communicate with the server. In this exercise we will extend the socket server and create a very basic HTTP server.

**Purpose:**

To understand basic communication details at the point of the HTTP server Build a proper HTTP response

**Deliverables**

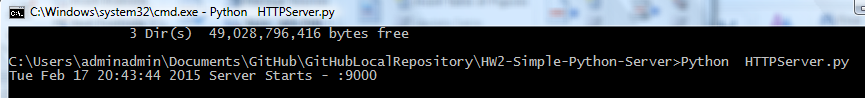
1. *Run the server as it is and notice what is written to the command window as you click on the links. What is going on here? (Short paragraph or two.)*

First, here is the ‘play-by-play’ action - the paragraphs will follow

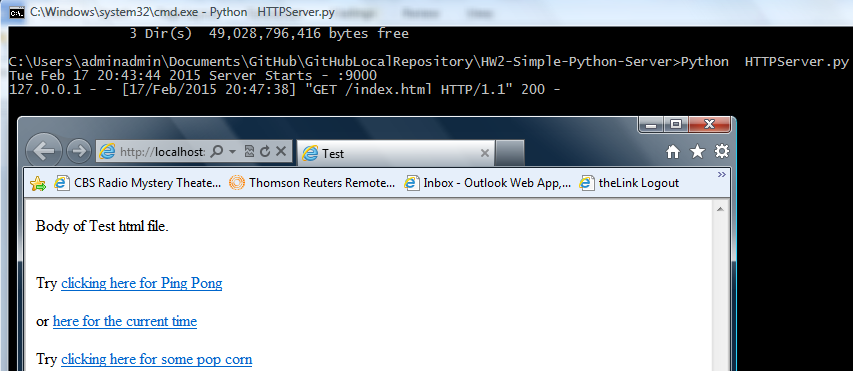
* Open a command window
* Navigate to: C:\Users\adminadmin\Documents\GitHub

\GitHubLocalRepository\HW2-Simple-Python-Server

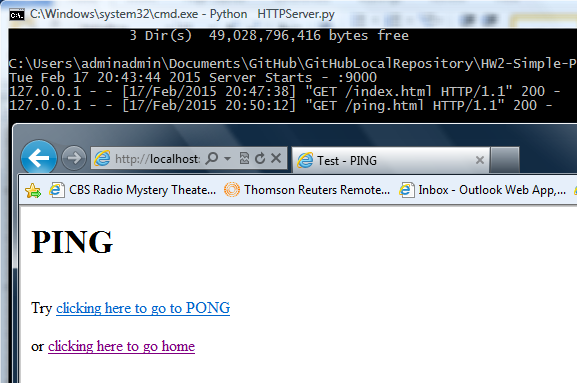
* Enter: Python HTTPServer.py



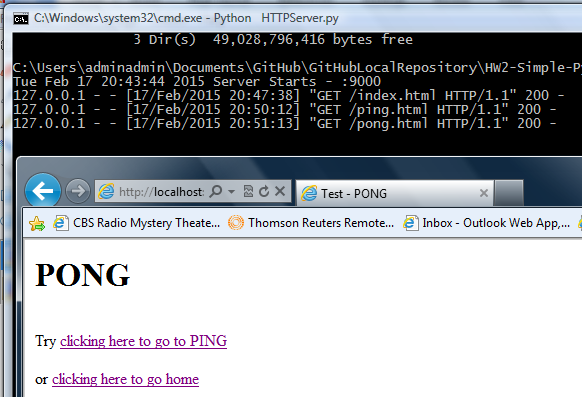
* Open a browser
* Enter: http://localhost:9000/index.html



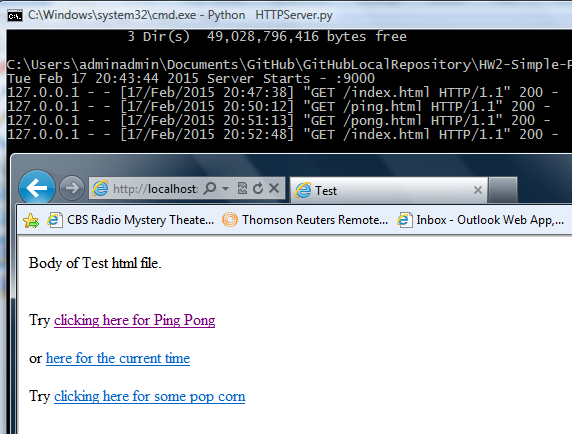
* Click on: Clicking\_here\_for\_Ping\_Pong



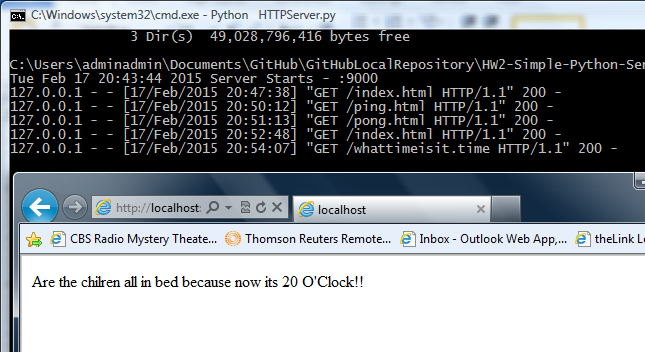
* Click on: Clicking\_here\_to\_go\_to\_Pong



* Click on: clicking\_here\_to\_home



* Click on: here\_for\_the\_current\_time



The command window seems to be echoing out each command the Python Server executes. So, each time I click on one of the links, a “GET” is performed, where the thing it ‘GET’s’ is an HTML page. Below is the server’s code that processes a GET.

def do\_GET(self): #Respond to a GET request.

try:

if self.path.endswith((".html",".htm")):

f = open(curdir + sep + self.path, 'rb') #self.path has /index.html

self.send\_response(200)

self.send\_header("Content-type", "text/html")

self.end\_headers()

# Start sending content

self.wfile.write(f.read())

f.close()

return

Here is an example of a header captured by Fiddler:

GET /ping.html HTTP/1.1

Accept: text/html, application/xhtml+xml, \*/\*

Referer: http://localhost:9000/index.html

Accept-Language: en-us

User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.0; Trident/5.0)

Accept-Encoding: gzip, deflate

Host: localhost:9000

Connection: Keep-Alive

Here is the RAW view from Fiddler:

HTTP/1.0 200 OK

Server: BaseHTTP/0.3 Python/2.7.9

Date: Wed, 18 Feb 2015 03:22:55 GMT

Content-type: text/html

<html>

<head>

<title>Test - PING</title>

</head>

<body>

<h1>PING</h1></br>

Try <a href="pong.html">clicking here to go to PONG</a>

<br />

<br />

or <a href="index.html">clicking here to go home</a>

</body>

</html>

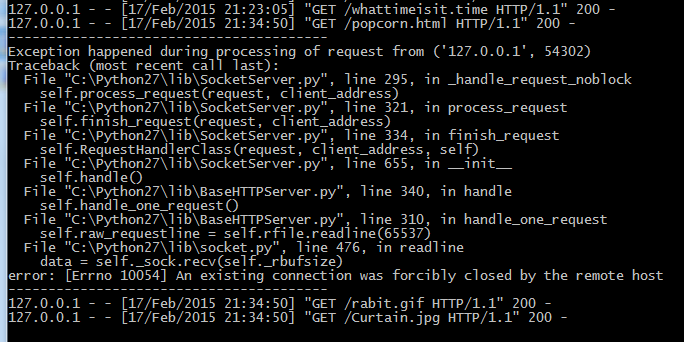
So when I click on a link, the browser forms a GET request, with the file name (see these highlighted in the Fiddler output above) included as the target.

**2 Build a simple 3-5 page website.**

* 1. *Interlink the pages so you can navigate between all of them.*
  2. *Include some (at least 2) images, at least one GIF and one JPG. Change the server code to enable you to*
     1. *render GIFs in your web pages*
     2. *render JPGs in your web pages*
     3. *allow the default request (http://localhost:9000/) return index.html*
     4. *With images working on your pages, note what is happening in the command window as you click from page to page.*

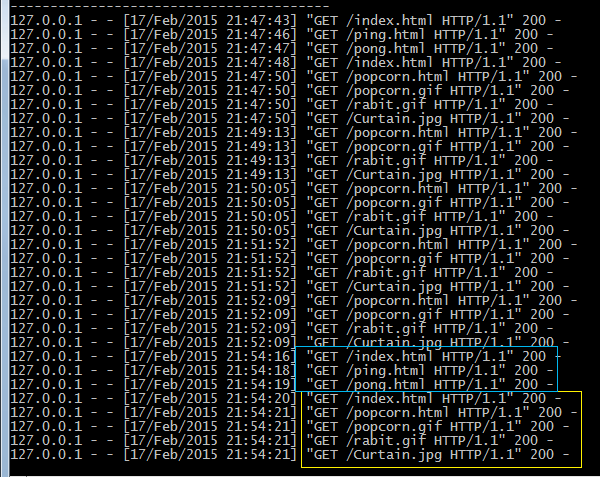
*Submit the file as a pull request to the original project.*

I got errors with IE ( I am running VISTA which is limited to an outdated version of IE).



Things went better with FireFox, even the animation worked.





Looking in the command window, I see GET’s for the ping.html and pong.html pages as before. The after the GET of the popcorn.html page, I see GET’s for the .gif and .jpg files.

**Appendix A Server listing**

#University of St. Thomas, Advanced Web Applications, SEIS752 Fall 2009

#Lloyd Cledwyn

#Basic Http Server example

#Adapted from http://wiki.python.org/moin/BaseHttpServer

import time

import BaseHTTPServer

from os import curdir, sep

#Define some variables to be used in the execution of the program

HOST\_NAME = '' # can be 'localhost' or if you change your hosts.txt file, what happens?? ;)

PORT\_NUMBER = 9000 #If you kill the server un-gracefully you may need to change this to an open socket.

class MyHandler(BaseHTTPServer.BaseHTTPRequestHandler):

#MyHandler class implements standard standard HTTP menthods

#currently HEAD and GET requests are handled.

def do\_HEAD(self):

self.send\_response(200)

self.send\_header("Content-type", "text/html")

self.end\_headers()

def do\_GET(self):

#Respond to a GET request.

try:

if self.path.endswith((".html",".htm")):

f = open(curdir + sep + self.path, 'rb') #self.path has /index.html

self.send\_response(200)

self.send\_header("Content-type", "text/html")

self.end\_headers()

# Start sending content

self.wfile.write(f.read())

f.close()

return

if self.path.endswith(".time"):

self.send\_response(200)

self.send\_header("Content-type", "text/html")

self.end\_headers()

# Send current time

self.wfile.write("Are the chilren all in bed because now its " + str(time.localtime()[3]) + " O'Clock!!")

return

if self.path.endswith(".gif"):

# IMPLEMENT THIS

f = open(curdir + sep + self.path, 'rb') #self.path has /popcorn.gif

self.send\_response(200)

self.send\_header("Content-type", "image/gif")

self.end\_headers()

self.wfile.write(f.read())

f.close()

return

if self.path.endswith(".jpg"):

# IMPLEMENT THIS

f = open(curdir + sep + self.path, 'rb') #self.path has /Curtain.jpg

self.send\_response(200)

self.send\_header("Content-type", "image/jpg")

self.end\_headers()

self.wfile.write(f.read())

f.close()

return

except IOError:

self.send\_error(404,'File not found: %self' % self.path)

if \_\_name\_\_ == '\_\_main\_\_':

server\_class = BaseHTTPServer.HTTPServer #Instantiate a server object

httpd = server\_class((HOST\_NAME, PORT\_NUMBER), MyHandler) #Tell the serever what hostname & port to run on, then what handler to handle the server requests.

print time.asctime(), "Server Starts - %s:%s" % (HOST\_NAME, PORT\_NUMBER)

try:

httpd.serve\_forever() #Run the server.

except KeyboardInterrupt:

pass

httpd.server\_close()

print time.asctime(), "Server Stops - %s:%s" % (HOST\_NAME, PORT\_NUMBER)

**Appendix B Popcorn.html**

<html>

<head>

<title>Test - POPCORN</title>

</head>

<body>

<h2>POPCORN (this is a .gif )</h2>

<img src="./popcorn.gif" width=240 height=100>

<br />

<br />

and so is this .....

<img src="./rabit.gif" width=130 height=90>

<br />

<br />

<h2>MOVIE CURTAIN (this is a .jpg) </h2>

<br />

<br />

<img src="./Curtain.jpg" width=300 height=200>

<!-- <img src="./calf.jpg" width=300 height=200> -->

<br />

<br />

Or <a href="index.html">click here to go home</a>

</body>

</html>