Python Proxy Server

Author: Manuel Aguilar

Requirements

- Python3
    - will throw exception if trying to run it using Python2.
- Full Source Code
    - https://github.com/maguilar15/ProxyServer

Start the server

```
~$ python ProxyServer.py 127.0.0.1
```

```
(base) user@free:~/Desktop/utep/computer-networks/ProxyServer$ python ProxyServer.py 127.0.0.1
Ready to serve...
```

Sending HTTP Requests

I will demonstrate that the proxy does take HTTP requests. I will be using a command-line utility called **httpie**. Which I can install on my host machine using Ubuntu's package manager snap.

```
~$ sudo snap install httpie --classic
```

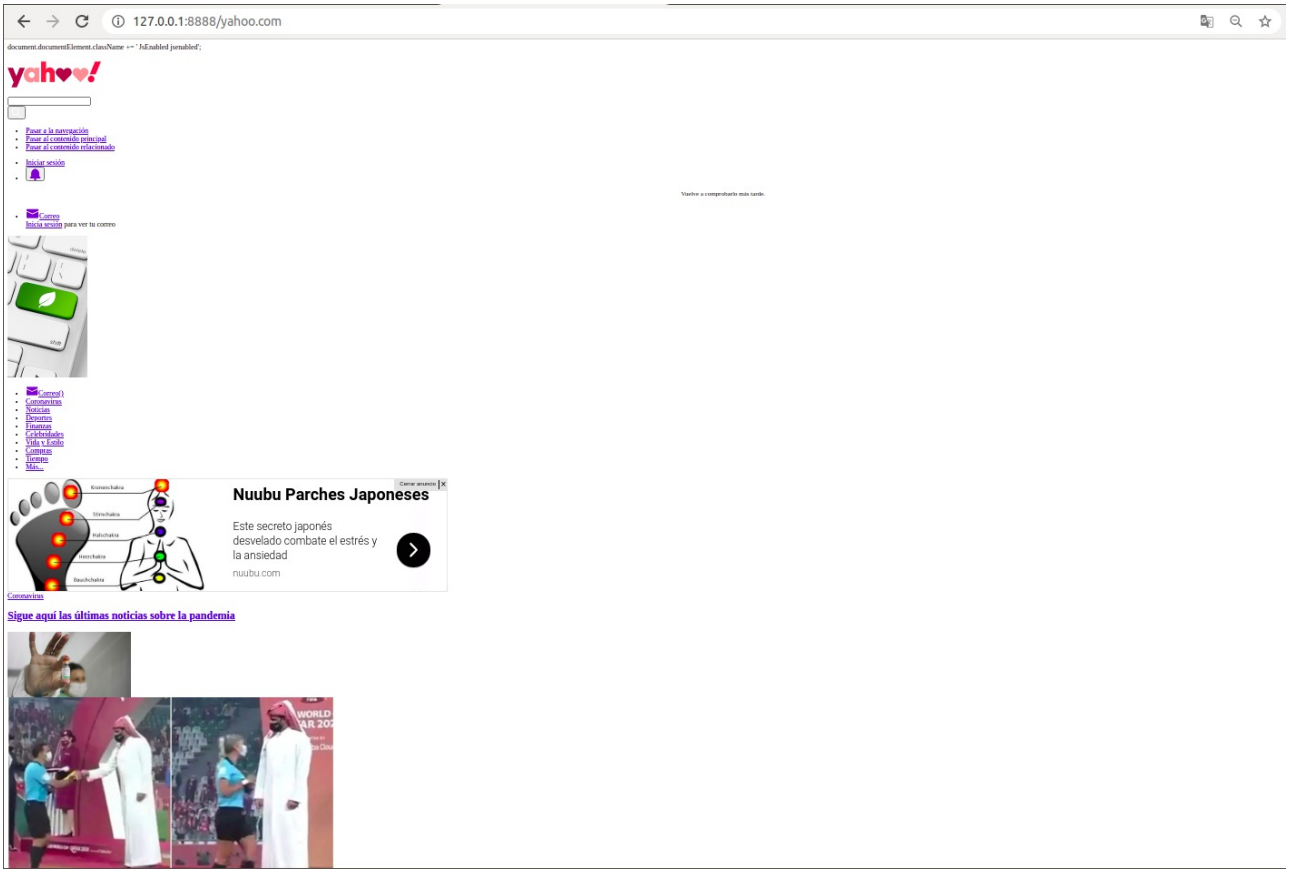
Using the command-line utility to send a GET request will give us back a 200 response code.

```
(base) user@free:~/Desktop/utep/computer-networks$ http get http://127.0.0.1:8888/google.com
HTTP/1.0 200 OK
Connection: close
Content-Type: text/html; charset=UTF-8
```

Doing the same thing ,but for a website that does not exist will give us a 404 response code.

```
(base) user@free:~/Desktop/utep/computer-networks$ http get http://127.0.0.1:8888/notgoogle.edu
HTTP/1.0 404 sendErrorErrorError
Content-Type: text/html
```

Viewing websites in the browser from data requested by the proxy server.



Searching in Cache

Searching in cache if the file exists. If the file does not exist it then raises an exception and requests the data from the origin server.

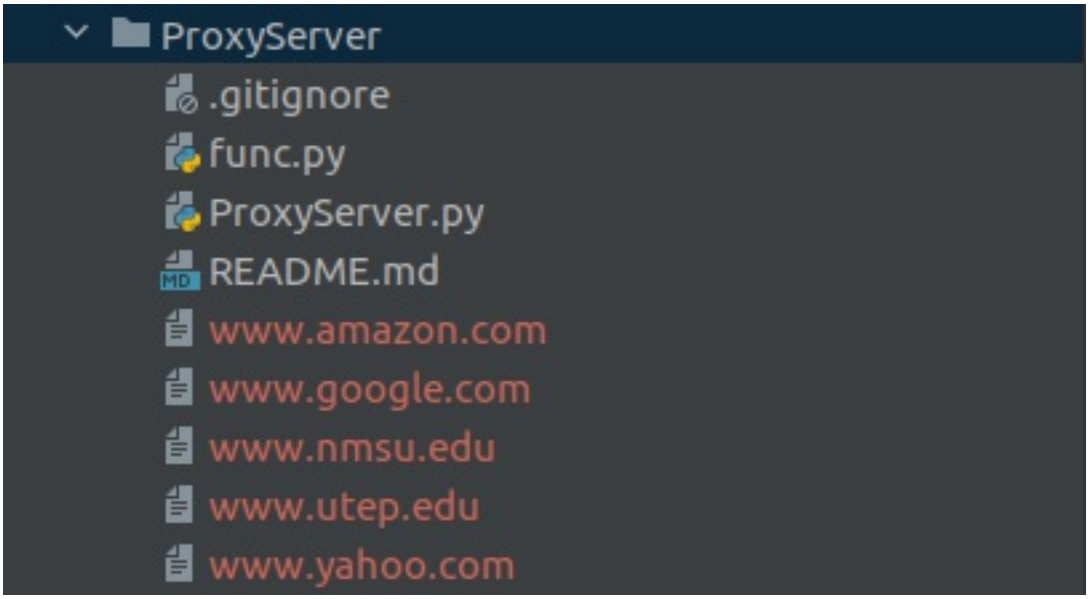
```
Searching for: yahoo.com , with hostname: www.yahoo.com
File does not exist in cache, searching for filename=www.yahoo.com
Fetching from Origin Server.....................................Request Data
```

```
Response Code: 200 for www.yahoo.com
written yahoo.com  to cache.
Ready to serve...
Received a connection from: ('127.0.0.1', 56258)
GET /yahoo.com HTTP/1.1
Host: 127.0.0.1:8888
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: clive-visitor-tid-27ô=kkz1tpf3r32rpk77s335ecsgs7ou1m54zxc08fpjsve1dsyv7lsesm2hi1s2qtvv; csm-hit=tb:P6QE0JT3N3KF6JXJDHBW+s-P6QE0JT3N3KF6JXJDHBW|161
3322657127&t:1613322657127&adb:adblk_no; _gat=1; _gat_b=1; _ga=GA1.1.550267127.1613319635; _gid=GA1.1.1939824581.1613319635
```

The next request will be stored in proxy servers' cache.

```
Searching for: yahoo.com , with hostname: www.yahoo.com
www.yahoo.com is stored in cache.
Loading the web page www.yahoo.com with 2321 number of bytes.
```

The python application will create the cache files inside the current directory.



Threads

Adding support for multiple connections using the threading library. For every request that is submitted to the server a new thread will spawn. This is indicated by the initialization of the Thread object. The thread object has the daemon property set to true. Having this property enabled will allow the thread object to background and exit from the main program. Threads can only be started once and an error can occur if you don't to handle multiple requests without rejoining to the main program. The join method on a started thread allows for the thread to finish executing before running the main thread. The programs main thread waits for any thread to complete before resuming.

```
    while 1:
        t = Thread(name="CacheService", target=start_proxy_server, args=(tcpSerSock,), daemon=True)
        t.start()
        t.join()
```

References

- Socket programming in python: https://realpython.com/python-sockets/
- Socket programming in python: https://www.geeks3d.com/hacklab/20190110/python-3-simple-http-request-with-the-socket-module/
- Threads in python: https://realpython.com/intro-to-python-threading/
- Threads in python: https://docs.python.org/3/library/threading.html