

COMPUTACION II

TP4

Fecha de entrega Final: 03/11/2020

Problema

El objetivo del práctico es desarrollar un servidor web asíncrono que pueda atender múltiples conexiones simultáneas.

Se debe especificar con la opción `-p` o `--port` el puerto donde espera conexiones nuevas. Con la opción `-d` o `--document-root` el directorio donde se encuentran los documentos web y con la opción `-s` o `--size` la cantidad máxima de bytes que se irán leyendo de los documentos web. En caso que no se solicite ningún documento, se debe responder con un archivo `index.html` de bienvenida.

Se debe crear una rutina para la atención de los clientes, y otra para registrar (loguear) las direcciones y fechas de acceso de los clientes.

Debe hacer una tabla comparativa de rendimiento respecto al mismo servidor del TP3, usando una concurrencia de 1, 10 y 100 para 1000, 5000 y 10000 requerimientos.

Requerimientos

- * La aplicación debe contener como mínimo 3 funciones.
- * Debe utilizar el módulo `asyncio`.
- * Debe implementar el método de requerimiento GET al menos.
- * Debe devolver como mínimo, tres tipos de resultados al cliente: 200 Ok, 404 Not Found y 500 Internal Server Error (Ver especificación de HTTP 1.1) con los headers de respuesta `Content-Length` y `Content-Type` correspondientes.
- * Debe soportar archivos de tipo `html`, `jpg`, `pdf` y `ppm`.
- * La lectura de los archivos se debe hacer como máximo de `a s` bytes.
- * Debe procesar las opciones con `getopt` (agregar una opción de ayuda) o con `argparse`. *
- Debe soportar sockets de tipo IPv4 o IPv6 indistintamente.
- * Debe soportar al menos diez accesos concurrentes y mil requerimientos por acceso. (Apache benchmark - "ab -c 10 -n 1000" ... man ab)

Ejemplo modo de uso

~~~~~

```
$ ./tp4.py -h
usage: tp4.py [-h] -d DIR -p PUERTO -s SIZE
```

Tp3 - servidor web y filtro de ppm

optional arguments:

|                                         |                                              |
|-----------------------------------------|----------------------------------------------|
| <code>-h, --help</code>                 | show this help message and exit              |
| <code>-d DIR, --documentroot DIR</code> | Directorio donde estan los documentos web    |
| <code>-p PORT, --port PORT</code>       | Puerto en donde espera conexiones nuevas     |
| <code>-s SIZE, --size SIZE</code>       | Bloque de lectura máxima para los documentos |

```
$ ./tp4.py --port 5000 -s 1024 -d /tmp
```

~~~~~

consultas:

```
wget http://192.168.2.2:5000/index.html
wget http://192.168.2.2:5000/imagen.jpg
wget http://192.168.2.2:5000/enunciado.pdf
wget http://192.168.2.2:5000/dog.ppm
```

Objetivos

- * Uso de mecanismos de IPC. Socket.
- * Manejo de corrutinas, tareas, awaitables y event loops.
- * Manejo de archivos (apertura, escritura y cierre).

Referencias

- * mime types en /etc/mime.types
- * RFC 1945 - Hypertext Transfer Protocol -- HTTP/1.0
- * Ver concepto de URI, URL, mime type, cabecera HTTP (request y response), HTTP status code.

Bonus Track

Utilizar el framework aiohttp y agregarlo en la tabla comparativa.

