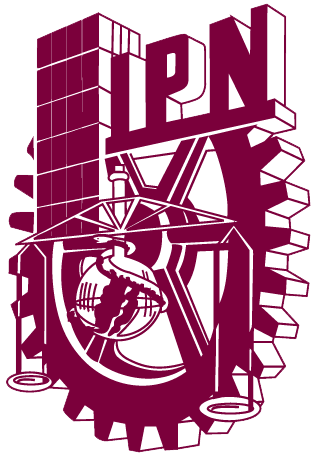
**Instituto Politécnico Nacional**

**Escuela Superior de Cómputo**

**Aplicaciones para Comunicaciones de Red**

**Práctica 4**

**Servidor Web**

**Miembros:**

Benítez Ramírez Sergio

Monteros Cervantes Miguel Angel



3CM15

**Introducción**

**Hilos**

Un hilo es un flujo de control dentro de un programa. Creando varios hilos podremos realizar varias tareas simultáneamente. Cada hilo tendrá sólo un contexto de ejecución (contador de programa, pila de ejecución). Es decir, a diferencia de los procesos UNIX, no tienen su propio espacio de memoria sino que acceden todos al mismo espacio de memoria común, por lo que será importante su sincronización cuando tengamos varios hilos accediendo a los mismos objetos.

Creación de hilos

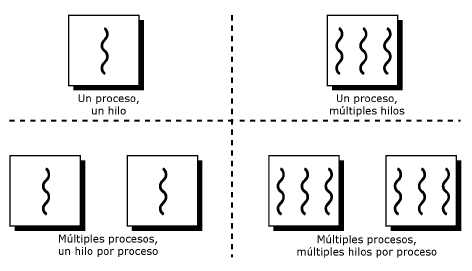
En Java los hilos están encapsulados en la clase Thread. Para crear un hilo tenemos dos posibilidades:

Heredar de Thread redefiniendo el método run().

Crear una clase que implemente la interfaz Runnable que nos obliga a definir el método run().

En ambos casos debemos definir un método run() que será el que contenga el código del hilo. Desde dentro de este método podremos llamar a cualquier otro método de cualquier objeto, pero este método run() será el método que se invoque cuando iniciemos la ejecución de un hilo. El hilo terminará su ejecución cuando termine de ejecutarse este método run()implementar Serializable.

Al llamar al método start del hilo, comenzará ejecutarse su método run. Crear un hilo heredando de Thread tiene el problema de que al no haber herencia múltiple en Java, si heredamos de Thread no podremos heredar de ninguna otra clase, y por lo tanto un hilo no podría heredar de ninguna otra clase. Este problema desaparece si utilizamos la interfaz Runnable para crear el hilo, ya que una clase puede implementar varios interfaces.



**Peticiones HTTP**

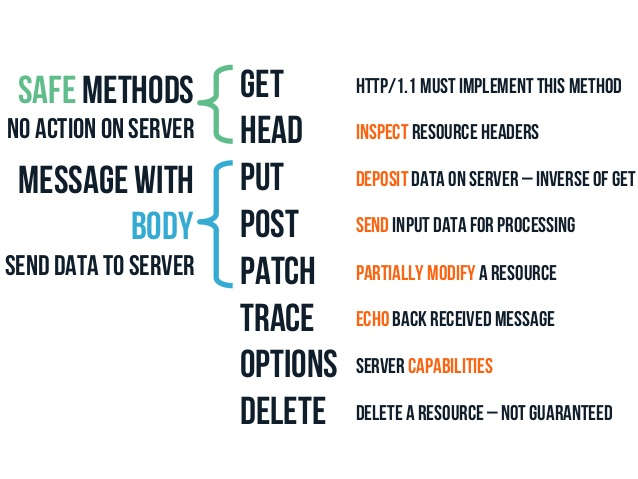
HTTP contiene un grupo de peticiones HTTP (también llamadas HTTP verbs por, el tipo de nombre que manejan casi todos ellos -pues si bien algunos son sustantivos, la gran mayoría no-) que nos ayudan a especificar la acción que se requiere realizar en un elemento determinado y aunque estas peticiones tienen distintas semánticas, también tienen muchas similitudes en las mismas que evitan que este grupo se extienda demasiado.

**Peticiones HTTP Safe**

Un método HTTP es considerado safe o seguro si no altera el estado del servidor. En otras palabras, un método es seguro si conduce a una operación de ‘sólo lectura’. Algunos de los métodos HTTP más comunes son seguros: OPTIONS, GET o HEAD. Todos los métodos seguros son también a su vez idempotent (así como también algunos, pero no todos, los métodos inseguros como DELETE o PUT).

**Peticiones HTTP Idempotent**

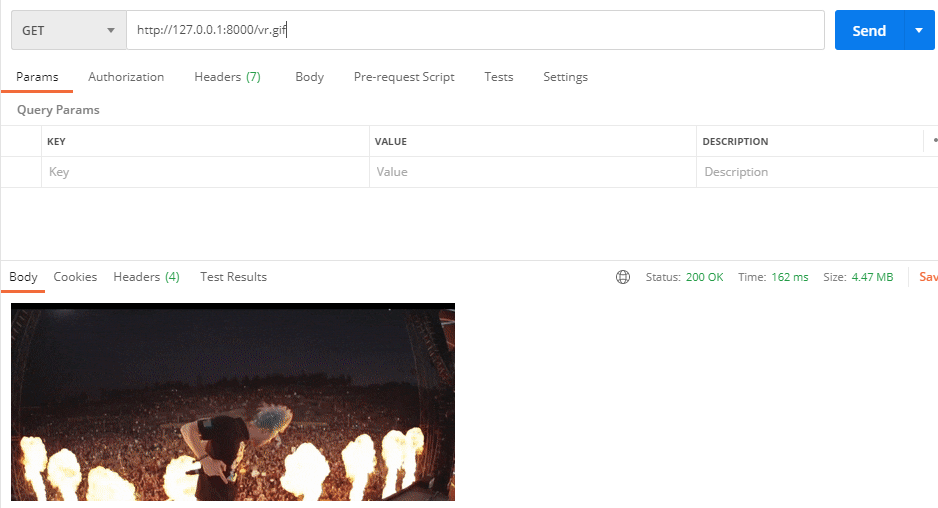
Los métodos que (implementados correctamente) son idempotentes son el GET, HEAD, PUT y DELETE y como lo explicamos en los métodos Safe, todos éstos últimos son idempotentes.



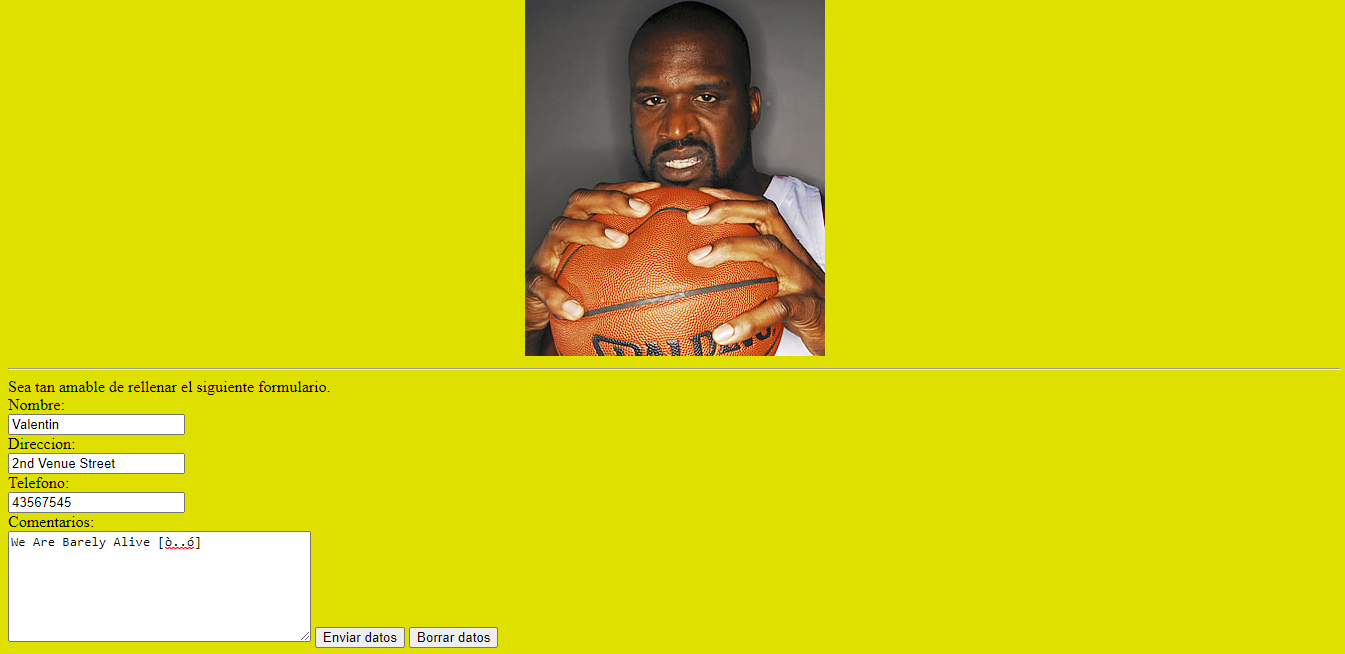
**Desarrollo**

Para observar los resultados necesitados tenemos que irnos nuestro postman y realizar las peticiones correspondientes.

**GET**

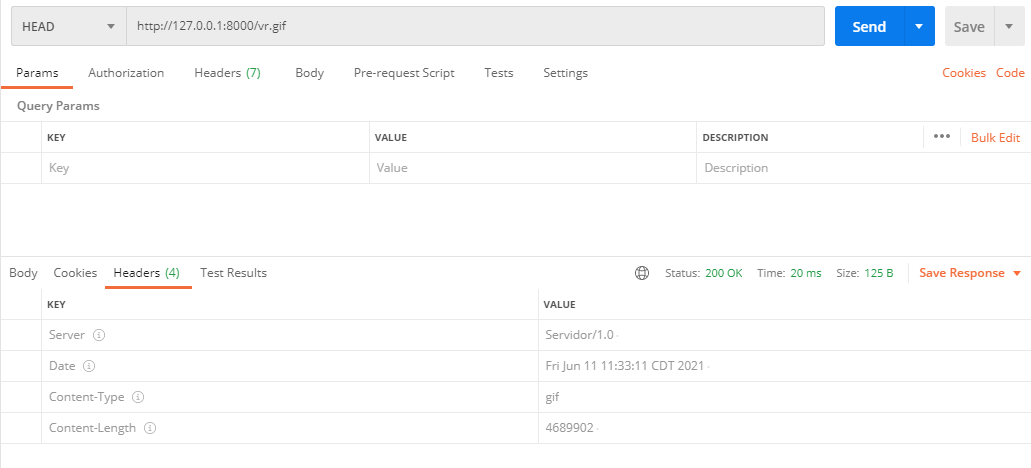
****

**POST**

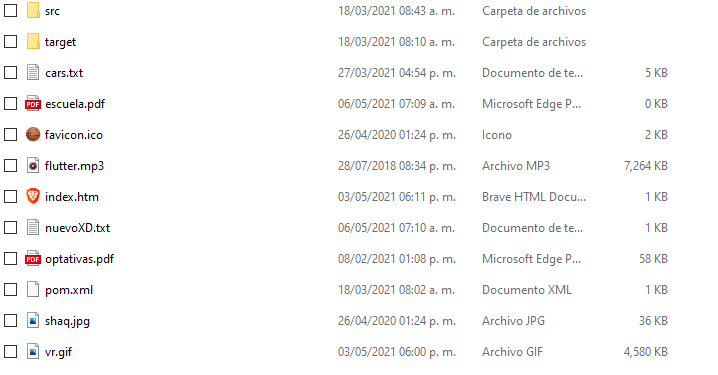
****

****

**HEAD**

****

**PUT**

****

**Conclusión**

En esta práctica pude entender mejor cómo funciona el flujo de entrada y de salida en el cliente en el servidor en un sistema que se conectan mediante sockets. Gracias a que este proceso utiliza el protocolo TCP/IP, nos asegura que los datos lleguen completa y nos notifica en caso de algún error. Solo nos preocupamos por mandar los paths de las carpetas y archivos de ambos lados y poder manejarlos. Pude identificar las ventajas y desventajas de la comunicación uno a uno en la arquitectura cliente-servidor ya que, al utilizar socket bloqueante, es muy importante abrir y cerrar los flujos de entrada, salida y los sockets para que pueda seguir sirviendo respuestas a los demás clientes.

**Referencias**

<https://yosoy.dev/peticiones-http-get-post-put-delete-etc/>

<https://developer.mozilla.org/es/docs/Web/HTTP/Methods>

<https://www.ionos.mx/digitalguide/hosting/cuestiones-tecnicas/http-request/>

<http://www.jtech.ua.es/j2ee/restringido/cw/sesion03-apuntes.html>