TALLER CLIENTES Y SERVICIOS

Carlos Manuel Murillo Ibañez

Luis Daniel Benavides Navarro
Arquitecturas Empresariales

Escuela Colombiana de Ingeniería Julio Garavito
3 de Septiembre del 2020
Bogotá D.C.

Índice

1. Resumen

En este artículo se expondrá la teoría y la arquitectura que se utilizó para crear un servidor web capaz de recibir múltiples solicitudes (no concurrentes), además de que se creó un framework parecido a Spark el cual permite publicar servicios web 'get' y permite acceder a recursos estáticos como páginas, javascripts, imágenes y CSSs.

2. Introducción

En este laboratorio se tiene como objetivo principal desarrollar un servidor web que sea capaz sea capaz de recibir peticiones y retornar todos los archivos solicitados, incluyendo páginas html e imágenes, tambien se desarrolló un framework parecido a Spark que permite publicar servicios web 'get' y le permita acceder a recursos estáticos como páginas, javascripts, imágenes, y CSSs. Para tener mayor claridad del tema en primera parte se realizará una explicación de la teoría que se necesitó para el desarrollar este servidor y framework, en segunda parte se explicara la arquitectura que se implementó lograr el funcionamiento del servidor con el uso del framework y para terminar en la tercera parte se presentaran las pruebas realizadas para confirmar el correcto funcionamiento.

3. Teoria

- Servidor Web: Un servidor Web es un programa que utiliza HTTP (Hypertext Transfer Protocol) para servir los archivos que forman páginas Web a los usuarios, en respuesta a sus solicitudes, que son reenviados por los clientes HTTP de sus computadoras.[1]
- Spark Framework: Es un conjunto de librerías para el desarrollo de aplicaciones web en Java inspirado en el framework Sinatra para Ruby.[2]
- Peticiones HTTP: Define un conjunto de métodos de petición para indicar la acción que se desea realizar para un recurso determinado.[3]

4. Diseño

En esta sección se explicará el diseño que se tuvo en cuenta para el desarrollo de este programa.

4.1. MySpark

Con este se busca realizar una pequeña implementación parecida al funcionamiento del framwork Spark donde se ofrecen métodos get y post de los endpoints que se tiene disponibles los cuales se encuentra almacenados en un hashmap.

4.2. HttpServer

La clase HttpServer es la implementación propia de un servidor el cual es capaz de recibir peticiones HTTP y retorna el recurso pedido.

4.3. Principal

Esta clase se encarga de inicializar el servidor para tenerlo disponible además de que agrega los endpoints con los que se va a trabajar.

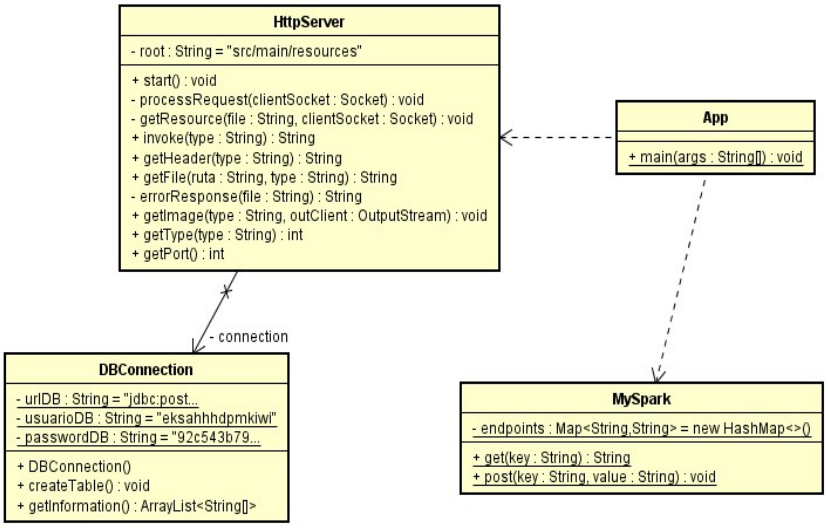

Figura 1: Diagrama de clases.

5. Arquitectura

La arquitectura que podemos identificar para el desarrollo de este proyecto es cliente-servidor en la cual el cliente que puede ser cualquier navegador con acceso a internet envía peticiones HTTP al servidor en el que son recibidas por medio de un controlador ofrecido por el framework de propio diseño el cual cuenta con el método get para obtener los recursos. Para una mayor claridad en la construcciónn de este proyecto vamos a nombrar y a identificar las tres abstracciones fundamentales del sistema:

- Como primera parte podemos identificar la abstracción de memoria en la cual se tiene en la HashMap donde se almacena todos los endpoints que se tienen disponibles además de que gracias a un método post podemos agregar más (write) y también esta misma estructura se puede recorrer y obtener en endpoint deseado gracias al método post(read).
- Como segunda parte podemos identificar la abstracción de los intérpretes, por parte del cliente tenemos cualquier tipo de navegador para consumir los servicios y por parte del servidor el cual al estar implementado en el lenguaje de programación JAVA nos ofrece la Java Virtual Machine (JVM) en donde se procesan todas las instrucciones que se le dan.
- Como tercera y última podemos identificar la abstracción de los enlaces de comunicación los cuales son las peticiones por medio del protocolo HTTP (get, post) y que se encarga de una correcta interacción del cliente con el servidor.

6. Pruebas

Para asegurar la efectividad y el correcto funcionamiento del programa desarrollado se hicieron una serie de pruebas en las cuales se le pedía al servidor una página html como podemos ver en la figura 2, también se le pidió al servidor una imagen como se puede ver en la figura 3 y por último se le pidió al recurso estático hola como se ve en la figura 4.

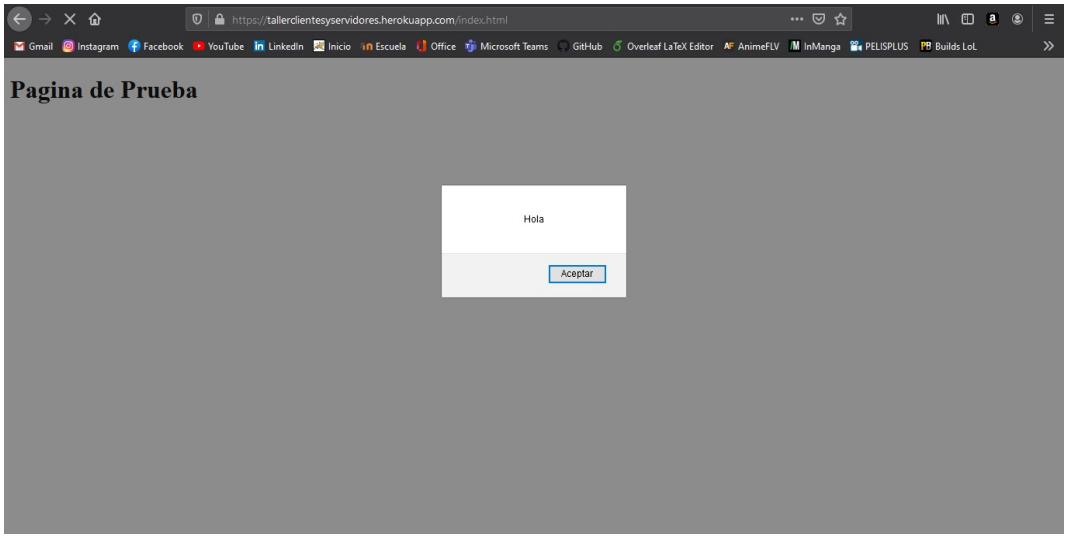

Figura 2: Pagina html con un JavaScript.

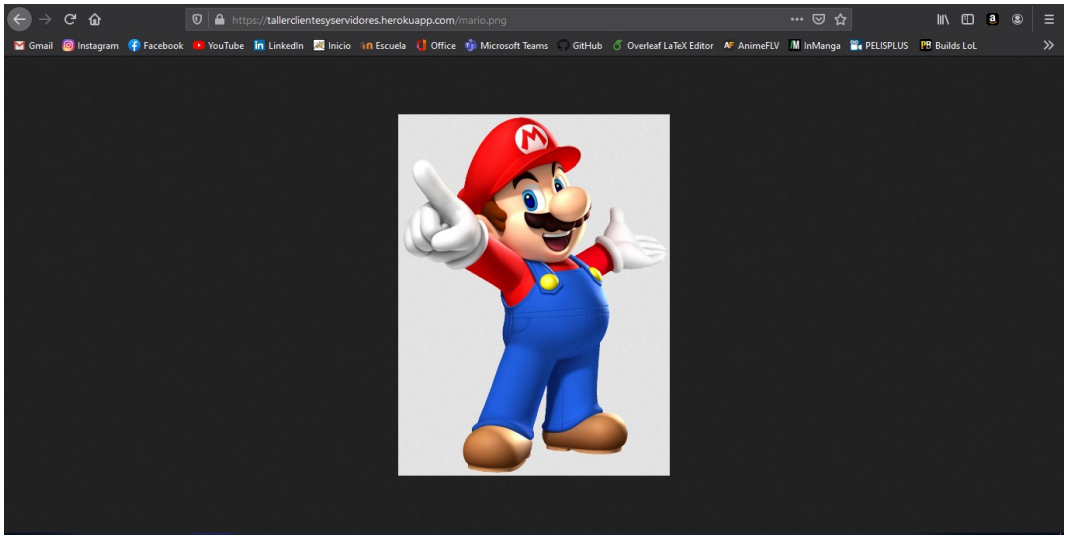

Figura 3: Imágen.

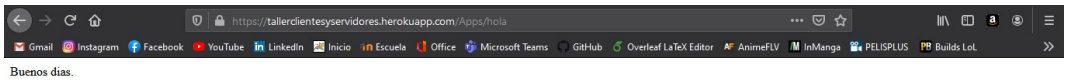

Figura 4: Recurso estatico.

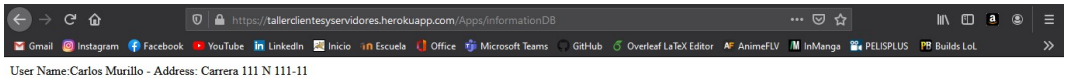

Figura 5: Información base de datos.

7. Conclusiones

Gracias a este laboratorio se logró entender el funcionamiento, mantenimiento y construcción de un servidor capaz de recibir múltiples peticiones con la integración de un framework parecido a Spark que fue construido desde cero, también se entendió como es el funcionamiento del framework por debajo.

Referencias

[1] ServidorWeb. https://www.osgroup.co/que-es-un-servidor-web/

[2] Spark. https://es.wikipedia.org/wiki/Spark_Framework

[3] HTTP. https://developer.mozilla.org/es/docs/Web/HTTP/Methods

[4] Moodle. https://campusvirtual.escuelaing.edu.co/moodle/mod/assign/view.php?id=34731