NanoSpark

Daniel Felipe Rincón Muñoz
Escuela Colombiana de Ingeniería Julio Garavito
Facultad de Ingeniería de Sistemas
daniel.rincon-m@mail.escuelaing.edu.co

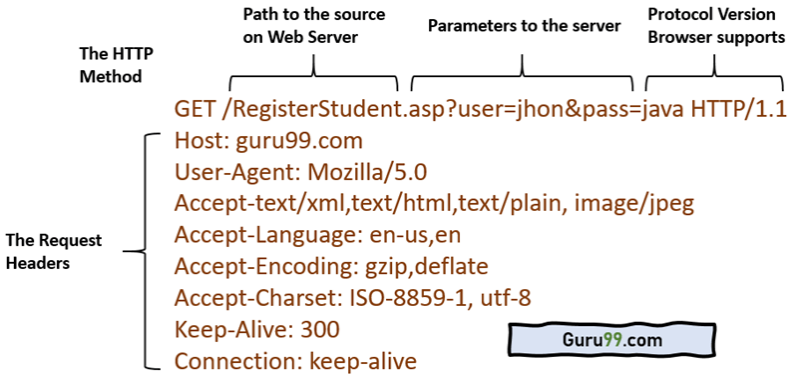
1 Introducción

El presente documento describirá el diseño que se siguió para realizar un servidor web capaz de recibir y transmitir mensajes HTTP por medio del protocolo GET, y posteriormente el diseño de un pequeño Framework web llamado NanoSpark, en el cual se pueden definir servicios web por medio de funciones lambda de una manera fácil e intuitiva.

1.1 Servidor Web

El servidor web cuenta con la capacidad de recibir y procesar peticiones GET, y de brindar una respuesta al cliente por medio de un flujo de bytes, esto nos da la flexibilidad de responder cualquier tipo de archivo que sea legible por el navegador sin la necesidad de implementar funcionalidades diferentes para cada tipo de archivo. Una petición GET luce de la siguiente forma:

Fig. 1: Formato de una petición GET



El servidor procesa la petición y accede a los recursos allí especificados, luego los codifica en un arreglo de bytes y se los devuelve al cliente.

1.2 NanoSpark

El Framework mini que se creó utiliza el servidor web antes mencionado, y crea una capa extra de abstracción, en donde el usuario final no se debe preocupar por manejar las peticiones o las respuestas, simplemente debe escribir el código que quiere que se ejecute al momento de recibir una petición a cierta URL, y automáticamente se retornará al cliente lo que el usuario desea que se retorne.

Para lograr esto, NanoSpark se encarga de procesar las peticiones y respuestas, y darles el formato adecuado para que sean entendidas por el navegador, agregando el resultado de la ejecución del código definido por el desarrollador para ser ejecutado en esa petición en específico.

2 Diseño

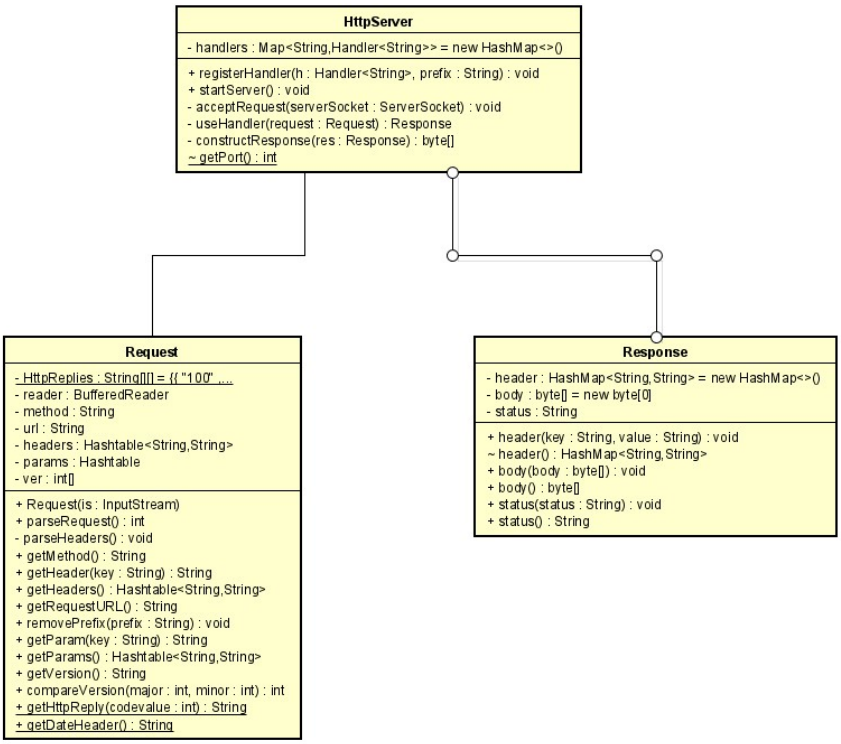
2.1 Servidor Web

Nuestro servidor web, como fue descrito en la introducción, cuenta con las siguientes características:

- ◇ Modificación del puerto de escucha por medio de las variables de entorno del sistema.
- ◇ Escucha activa de clientes de manera síncrona.
- ◇ Lectura y procesamiento de peticiones GET.
- ◇ Registro de diferentes "manejadores" para que procesen las peticiones a ciertas URL.
- ◇ Respuestas genéricas y flexibles por medio de su codificación en un arreglo de bytes, el cual es enviado al cliente para ser interpretado y mostrado en el navegador.

La arquitectura del servidor Web se muestra en la Figura 2.

Fig. 2: Arquitectura HTTP Server



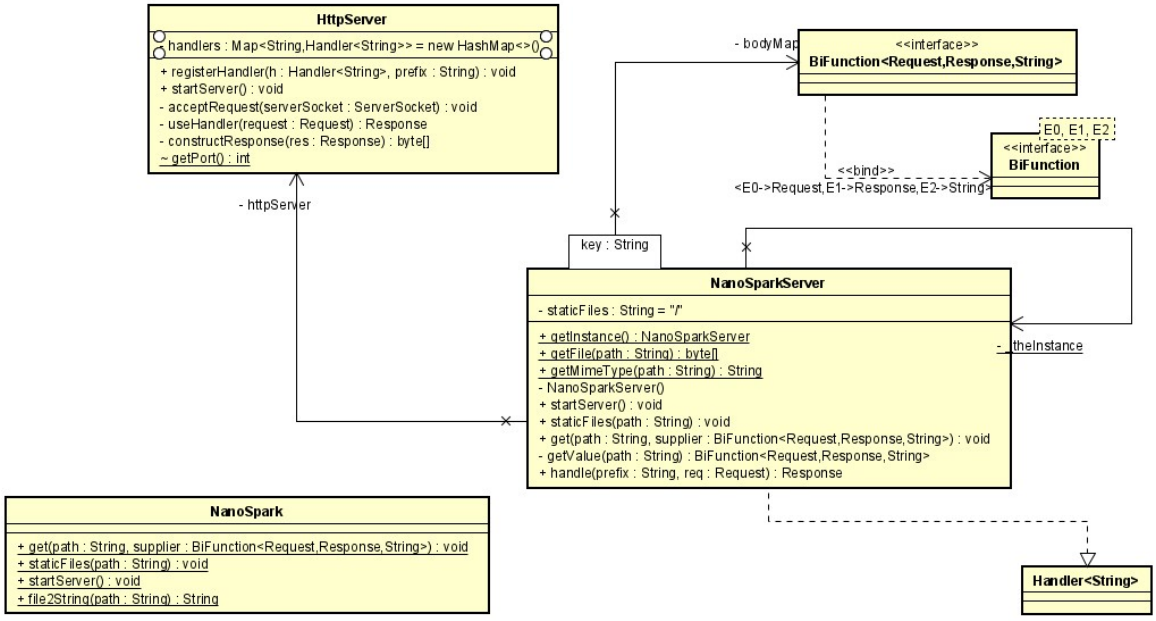
2.2 NanoSpark

El servicio de NanoSpark, descrito en la introducción, cuenta con las siguientes características:

- ◇ Búsqueda flexible de archivos dentro del servidor, teniendo en cuenta los prefijos registrados para las páginas el usuario.
- ◇ Búsqueda de MimeTypes en una base de datos local, dándonos la flexibilidad de procesar y enviar la mayoría de archivos en un formato entendible por el cliente.
- ◇ Directorio de archivos estáticos variable, para que el desarrollador tenga la libertad de elegir en donde poner sus datos.
- ◇ Manejador flexible de solicitudes, respetando los prefijos definidos por el usuario y con búsqueda de archivos estáticos en caso de error.

La arquitectura de NanoSpark se puede ver en la figura 3.

Fig. 3: Arquitectura NanoSpark



3 Conclusiones

- ◇ Realizar un servidor web que reciba y responda peticiones GET puede verse más intimidante de lo que realmente es, simplemente se trata de leer e interpretar la petición que nos llega, utilizar esta información para buscar recursos en nuestro servidor y enviarlos de vuelta al cliente con los estándares necesarios para que el lo pueda entender.
- ◇ El servidor web puede ser extendido para soportar peticiones POST, el cambio que se debe hacer es al momento de leer la petición y procesarla, decidir como se debe leer dependiendo si es un tipo de petición GET o POST, de esta forma siempre podremos leer los parámetros de manera correcta.