

Microframeworks Web

Requisitos de Finalización

Project Statement: Web Framework Development for REST Services and Static File Management

Objective

Este proyecto busca mejorar un servidor web existente —el cual actualmente soporta archivos HTML, JavaScript, CSS e imágenes— para convertirlo en un **microframework web** completamente funcional.

El framework permitirá crear aplicaciones web con servicios REST en el backend. Además, ofrecerá herramientas para:

- Definir servicios REST usando **funciones lambda**.
- Gestionar valores de consulta (query parameters).
- Especificar la ubicación de archivos estáticos.

Project Scope and Features

1. GET Static Method for REST Services

Implementar un método `get()` que permita definir servicios REST utilizando funciones lambda.

Ejemplo:

```
get("/hello", (req, res) -> "hello world!");
```

Esto permitirá a los desarrolladores definir rutas claras dentro de sus aplicaciones, mapeando URLs a expresiones lambda que gestionan solicitudes y respuestas.

2. Query Value Extraction Mechanism

Crear un mecanismo para extraer parámetros de consulta desde solicitudes entrantes y hacerlos accesibles dentro de los servicios REST.

Ejemplo:

```
get("/hello", (req, res) -> "hello " + req.getValues("name"));
```

Esto facilitará la creación de servicios dinámicos y parametrizados.

3. Static File Location Specification

Implementar el método `staticfiles()` para definir la carpeta donde residen los archivos estáticos.

Ejemplo:

```
staticfiles("webroot/public");
```

El framework deberá buscar archivos estáticos en dicha carpeta (por ejemplo, `target/classes/webroot/public`), permitiendo una mejor organización del proyecto.

4. Additional Tasks

Build an Example Application

Crear un ejemplo que demuestre cómo los desarrolladores usarían el framework.

La aplicación debe:

- Iniciar el servidor web.
- Servir archivos estáticos ubicados en `target/classes/webroot`.
- Exponer servicios REST GET.

Rutas esperadas:

- `http://localhost:8080/App/hello?name=Pedro`
- `http://localhost:8080/App/pi`

El prefijo `/App` es solo una sugerencia.

Example Code

```
public static void main(String[] args) {
    staticfiles("/webroot");

    get("/hello", (req, resp) ->
        "Hello " + req.getValues("name")
    );

    get("/pi", (req, resp) -> {
        return String.valueOf(Math.PI);
    });
}
```

El código debe también responder a solicitudes como:

```
http://localhost:8080/index.html
```

Deliverables

- Código fuente completo en un repositorio público de GitHub.
- Proyecto construido usando **Maven** y **Git**.
- README que incluya:
 - Descripción del proyecto
 - Arquitectura
 - Cómo ejecutarlo
 - Ejemplos de pruebas realizadas
- Estructura profesional del repositorio.

Outcome

Al finalizar el proyecto, los desarrolladores contarán con un microframework que permite crear aplicaciones web escalables y mantenibles.

El proceso fortalece la comprensión de:

- Arquitectura del protocolo HTTP
- Arquitectura de internet
- Aplicaciones distribuidas

La experiencia práctica permitirá diseñar e implementar servicios web modernos, eficientes y bien estructurados.