

Actividad Tratamiento de Imágenes

Renzo Jimenez Alfaro | Computación en el Cliente Web | 05/02/2024

# Desarrollo e Implementación

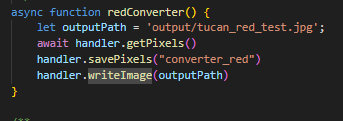
Para la implementación se descargó el material de apoyo más la instalación y configuración del IDE para programar, que en este caso fue Visual Studio Code.

Primero para entender el código, se leyó paso a paso cuales es el flujo natural del código ya implementado para poder saber que agregar y que modificar.

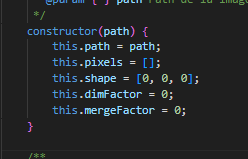
Segundo, ya leído el código base que se nos brindó se procedió a configurar los archivos actividad.js y ImageHandler.js, identificando así que hay 9 métodos las cuales se escogen para determinar un resultado dependiendo cual opción se escogió, dentro del código se identificó los siguientes métodos.

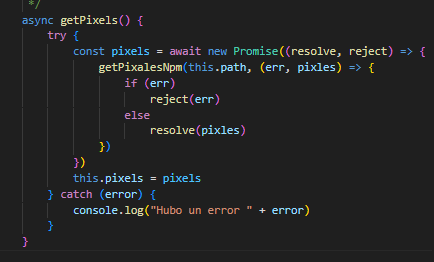
* redConverter
* greenConverter
* blueConverter
* greyConverter
* blackAndWhiteConverter
* scaleDown
* dimBrightness
* invertColors
* merge

Para las funciones desde el redConverter hasta el método blackAndWhiteConverter se mantuvo un código similar cambiando así solo las variables pedidas como la ruta del resultado, y el tipo de conversión a seguir teniendo así la siguiente estructura:

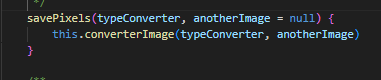


* La variable outputPath es la ruta del archivo resultante a convertir, esto varia de los metodos redConverter hasta el método merge, por ejemplo, para el redConverter use ‘output/tuca\_red\_test.jpg’, y para el siguiente greenConverter seria ‘output/tuca\_green\_test.jpg’, siguiendo así hasta el último de los métodos.
* Después tenemos el método getPixels que es un await porque para obtener los pixeles de una imagen tenemos una promesa que devolverá los pixels de la imagen pero que esta la sete en una variable propia de la clase imageHandle que se llama this.pixels, implementándolo de la siguiente manera, teniendo en cuenta que el constructor de la clase imageHandler que considere la más eficiente fue la siguiente



* La variable this.path, guardar la ruta de salida, this.pixels guardara el objeto que obtengamos de obtener la información de la imagen usando la librería get-pixels, this.shape tendrá la información de la altura, anchura y el número de filas y columnas de la imagen, el this.dimFactor es una variable que me permitirá saber cual es tipo de brillo que quiero darle a mi imagen lo cual fue mas eficiente guardarlo en el objeto y el mergeFactor se usó, para el último ejercicio en el cual se deben fusionar dos imágenes teniendo en cuenta un factor de fusión por cada imagen. 
* El método getPixels consta en una promesa que devuelve un mensaje de error en caso haya un error con obtener la información de la imagen, la función getPixalesNpm es un sobrenombre al método get-pixels de la librería, esta me permite obtener la información de una imagen pasándole la ruta donde esta la foto a obtener la información y una vez pasa exitosamente se guarda la información de los pixels en this.pixels para su posterior uso.

A continuación, se tiene el método savePixels de imageHandler que permite convertir una imagen dependiendo del tipo de conversión para estos tipos de conversión se consideró los siguientes:

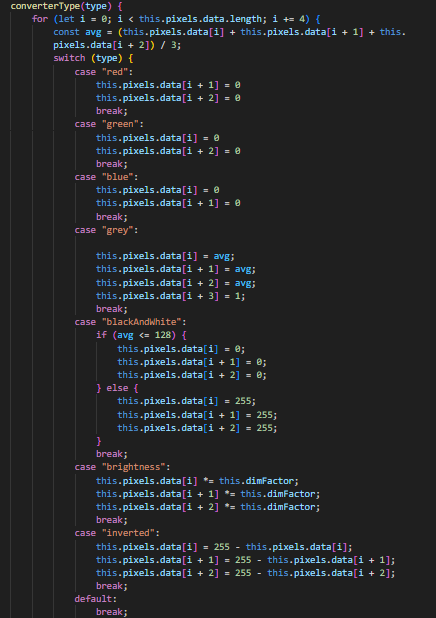


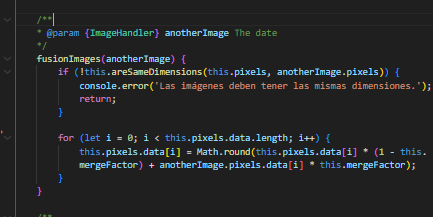
* Converter\_red
* Converter\_green
* Converter\_blue
* Converter\_grey
* Converter\_black\_white
* Converter\_scale\_down
* Converter\_brightness
* Invert\_colors
* Merger\_images

Esta función admite dos variables, pero por términos de simplicidad se pueden agregar dos, primero el tipo de conversión y el otro es la imagen la cual se quiere funcionar la cual se usa para la última opción de conversión. Dentro de esta función se tiene el método converterImage que es una función que dependiendo del tipo de conversión se ejecutara una lógica diferente dependiendo del tipo de conversión.

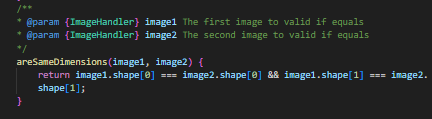


Y para seguir las buenas practicas de la programación se agregó otras funciones llamadas converterType y fusionImages que tendrán toda la lógica de conversión de las imágenes.





La función fusiónImages implementa una función adición que valida que las dimensiones de las dos imágenes sean iguales ya que solo se pueden fusionar imágenes que tengan las mismas dimensiones.



Por último, está el método writeImage que es un método que permite guardar la imagen en una ruta especifica usando las librerías savePixelsNpm que es una abreviación de save-pixels y usando métodos de la librería fs para escribir archivos.

