Outline

- Load an Image and draw it on a canvas.
- Create and configure a new Web Worker object.
- On the click of a button, get pixel data of the 2d canvas context as an ImageData object.
- Post the ImageData object to a Web Worker as its message data.
- Receive an ImageData object in the Web Worker.
- Convert RGB data of the Uint8ClampedArray of the ImageData to grayscale.
- Post the modified ImageData object back to the main script.
- Receive the modified ImageData object and draw it on the canvas.

Description

In this lab you will make use of several of the techniques covered in lectures involving both Web Workers and Image pixel data manipulation. To complete this lab you will need to write at least one HTML file and one JavaScript file. Your HTML file should have at least one <button> element and one <canvas> element.

As soon as your HTML file opens, the page script should load an Image file and draw it on the canvas. Images are loaded asynchronously so you should do your drawing as part of the "load" event handler function of an Image object. Don't forget to size the canvas to match the dimensions of the loaded image. Also, when the HTML file opens, create one Worker object that preloads your JavaScript file. Refer to examples from the lecture notes.

When the button on your page is clicked, get an ImageData object from a 2d context obtained from your canvas element. Use the `getImageData(…)` method of the 2d context to get its ImageData object. Post a message to your Worker with the ImageData object as message data.

When the new message is received by your Worker script access the ImageData object from the message event data. Loop over all pixel data in the ImageData's Uint8ClampedArray (the ImageData object's `.data` property) and convert all pixels to grayscale. Because the human eye is not equally sensitive to red, green and blue, use the following formula to compute a single grayscale value. Reassign all three color values in the Uint8ClampedArray to this grayscale value.

```
grayscale = 0.299*red + 0.587*green + 0.114*blue
```

Return the modified ImageData object by posting it back to the main script. Refer to the examples from recent lectures on how to modify pixel data in the ImageData object.

In your main script make sure to set up a "message" event handler for your Worker object. When a response is received from the Worker, get the modified ImageData object from the event object and draw it on the canvas using the 2d context object's `putImageData(…)` method.

Requirements

- Create at least one HTML and one JavaScript file.
- Create at least one <button> and one <canvas> element in your HTML file.
- When your HTML file opens, immediately load and draw an image on the resized canvas.
- Also immediately create a new Worker object that preloads your JavaScript file.
- Handle your button's "click" event so that when the button is clicked an ImageData object is obtained from the canvas and passed as message data to your Worker.
- Handle the "message" event inside your Worker script.
- Remove the ImageData object from the event argument data passed to the "message" event handler.
- Loop over pixel data in the ImageData's embedded Uint8ClampedArray and convert all red, green and blue bytes to grayscale using the provided formula.
- Post the modified ImageData object back to the main script.
- Handle the Worker "message" event in your main script by extracting the modified ImageData object from the received event data.
- Draw the modified ImageData object back to the canvas element.
- The main script in your HTML file must be enclosed in an IIFE.
- You MUST enter header comments in your JavaScript code including (1) your name, (2) description and or purpose of the assignment.
- You MUST comment your code, explaining what you did in each section.
- Submit JavaScript and HTML files on Canvas under the appropriate assignment.