Tarea 2
Web Assembly y PWAs

Transformación de imágenes

IIC3585

- Nicolás Cañete
- Amanda Sandoval
- Antonio Zambrano

Objetivo

- Crear una PWA que utilice módulos WASM para aplicar filtros a imagenes.

DEMO

Metodología

Se utilizó el lenguaje Rust para crear funciones que modifican una imagen, las cuales son utilizadas luego por la PWA.

WebAssembly

Funcionamiento

- Se carga la imagen
- Se aplica la transformación
- Se retorna la nueva imagen

Cargar la imagen

Se carga la imagen mediante *load_from_memory*, arrojando un mensaje en caso de error

```rust
#[wasm_bindgen]
pub fn resize(image_data: &[u8], width: u32, height: u32) -> Vec<u8> {
    let img = image::load_from_memory(image_data).expect("No se pudo cargar la imagen");
    let resized = img.resize(width, height, image::imageops::FilterType::Lanczos3);
    let mut buf = Cursor::new(Vec::new());
    resized.write_to(&mut buf, ImageFormat::Png)
        .expect("Error al escribir la imagen");
    buf.into_inner()
}
```

Procesar la imagen

Se procesa la imagen mediante el filtro que se desee (en este ejemplo *resize*), arrojando un mensaje en caso de error

```rust
#[wasm_bindgen]
pub fn resize(image_data: &[u8], width: u32, height: u32) -> Vec<u8> {
    let img = image::load_from_memory(image_data).expect("No se pudo cargar la imagen");
    let resized = img.resize(width, height, image::imageops::FilterType::Lanczos3);
    let mut buf = Cursor::new(Vec::new());
    resized.write_to(&mut buf, ImageFormat::Png)
        .expect("Error al escribir la imagen");
    buf.into_inner()
}
```

Retornar la imagen

Para retornar la imagen, primero se crea un buffer, luego se escribe en él el contenido de la imagen y finalmente se retorna

```rust
#[wasm_bindgen]
pub fn resize(image_data: &[u8], width: u32, height: u32) -> Vec<u8> {
    let img = image::load_from_memory(image_data).expect("No se pudo cargar la imagen");
    let resized = img.resize(width, height, image::imageops::FilterType::Lanczos3);
    let mut buf = Cursor::new(Vec::new());
    resized.write_to(&mut buf, ImageFormat::Png)
        .expect("Error al escribir la imagen");
    buf.into_inner()
}
```

PWA

Funcionalidades de PWA

En el desarrollo, se implementaron las siguientes funcionalidades de entre las propuestas:

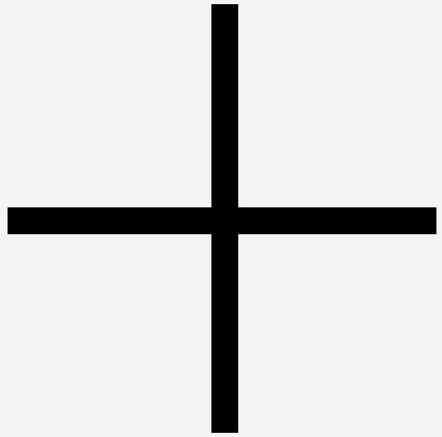
- Notificaciones push
- Guardado de imágenes usando IndexedDB
- Usar y guardar imágenes en una carpeta nativa

Notificaciones

Utilizando la API de notificaciones del navegador, primero se comprueba si se tiene permiso, y en caso de no tenerlo, se solicita para luego mostrar un mensaje

```
if (Notification.permission === 'granted') {
  new Notification('Imagen procesada', { body: 'El filtro se aplicó con éxito.' });
} else if (Notification.permission !== 'denied') {
  Notification.requestPermission().then(perm => {
    if (perm === 'granted') {
      new Notification('Imagen procesada', { body: 'El filtro se aplicó con éxito.' });
    }
  });
}
catch (err) {
console.error(err);
```

Guardado de imágenes usando IndexedDB

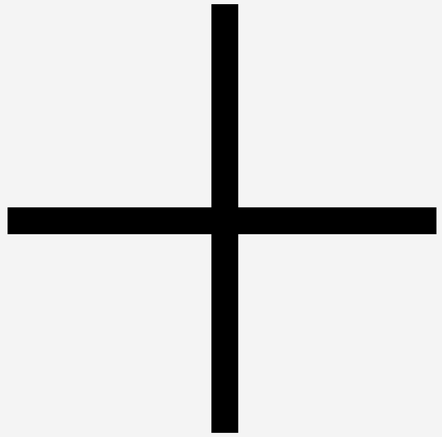


Usar y guardar imágenes en una carpeta nativa

- Se inicia la base de datos
- Se crea *images* para guardar las imágenes
- Si todo sale bien, se guarda la instancia

```
let db;
const request = window.indexedDB.open("ImageDatabase", 1);
request.onerror = function (event) {
  console.error("IndexedDB error:", event);
};
request.onupgradeneeded = function (event) {
  db = event.target.result;
  db.createObjectStore("images", { autoIncrement: true });
};
request.onsuccess = function (event) {
  db = event.target.result;
};
```

Guardado de imágenes usando IndexedDB



Usar y guardar imágenes en una carpeta nativa

- Cuando el usuario quiere guardar una imagen, se realiza una transacción en la base de datos y se agrega la imagen
- Al momento de descargar la imagen, se utiliza una referencia a la transacción previa

```
document.getElementById('saveImage').addEventListener('click', () => {
  if (!processedImageBlob) {
    alert('No hay imagen procesada para guardar.');
    return;
  }
  const transaction = db.transaction(["images"], "readwrite");
  const store = transaction.objectStore("images");
  const addRequest = store.add(processedImageBlob);
  addRequest.onsuccess = function () {
    alert('Imagen guardada en IndexedDB.');
  };
  addRequest.onerror = function (e) {
    console.error("Error al guardar:", e);
  };

  const downloadLink = document.createElement('a');
  const url = URL.createObjectURL(processedImageBlob);
  downloadLink.href = url;
  downloadLink.download = "processed_image.png";
  downloadLink.click();
});
```

Guardado de imágenes usando IndexedDB

Usar y guardar imágenes en una carpeta nativa

- En caso de querer observar las imágenes generadas previamente, se puede utilizar la opción mostrada para obtener los elementos guardados en IndexedDB

```
const transaction = db.transaction(["images"], "readonly");
const store = transaction.objectStore("images");
const getRequest = store.getAll();
getRequest.onsuccess = function (event) {
  const images = event.target.result;
  if (images.length > 0) {
    imageContainer.innerHTML = ''; // Clear previous images
    images.forEach((imageBlob, index) => {
      const imgUrl = URL.createObjectURL(imageBlob);
      const img = document.createElement('img');
      img.src = imgUrl;
      img.alt = `Image ${index + 1}`;
      img.style.margin = '10px';
      img.style.maxWidth = '200px';
      img.style.maxHeight = '200px';
      img.style.cursor = 'pointer';
      img.addEventListener('click', () => {
        const fileInput = document.getElementById('imageInput');
        const dataTransfer = new DataTransfer();
        const file = new File([imageBlob], `Image_${index + 1}.png`, { type: 'image/png' });
        dataTransfer.items.add(file);
        fileInput.files = dataTransfer.files;
        fileInput.dispatchEvent(new Event('change'));

        // Reset the resultCanvas
        const resultCanvas = document.getElementById('resultCanvas');
        const ctx = resultCanvas.getContext('2d');
        resultCanvas.width = 300; // Default width
        resultCanvas.height = 150; // Default height
        ctx.clearRect(0, 0, resultCanvas.width, resultCanvas.height);

        // Trigger the goBack event
        document.getElementById('goBack').click();
      });
      imageContainer.appendChild(img);
    });
  } else {
    alert('No hay imágenes guardadas.');
  }
};
```

¡Gracias por su atención!