**React**

**Async - Await.**

Documentación: <https://es.reactjs.org/>

<https://es.reactjs.org/docs/getting-started.html>

https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Statements/async\_function

Al momento de utilizar la palabra “async” en las funciones, lo que obtenemos es una promesa que se resuelve, pero a vista del código es más sencillo ver una función async que llamar una promesa, Ejemplo:

/\*

// Promesa opción larga

const obtenerImagen = () => {

    const promesa = new Promise((resolve, reject) =>{

        resolve('https://google.com')

    })

    return promesa;

}

\*/

// Promesa opción abreviada

const obtenerImagen = () => new Promise( resolve => resolve('https://www.google.com'));

obtenerImagen().then( console.log );

// Promesa utilizando async

const obtenerImagen2 = async () => 'https://www.google.com';

obtenerImagen2().then( console.log );

**Nota importante.**

Cabe destacar que al utilizar los “await”, siempre van de la mano de los “async”, el async puede estar por si solo sin necesitar del await… Cuando queremos transformar una función que retorne por defecto una promesa, utilizar el async es muy útil:

// Promesa utilizando async

const obtenerImagen2 = async () => 'https://www.google.com';

obtenerImagen2().then( console.log );

Entonces… ¿Qué es eso del await?

El await nos ayuda a que podamos trabajar todo el código como si fuera síncrono. En el siguiente código al momento de utilizar una función async y poner el await en la petición, podemos ver que la petición se vuelve de tipo response, es decir el await le dice a la función, espera a que esta promesa termine antes de ejecutar la siguiente línea de código, una cosa difícil de manejar con los Async – Await, es el manejo de los errores, por que se tienen que utilizar los “try – catch”. Ejemplo:

const obtenerImagenAsyncAwait = async () => {

    try {

        const apiKey = "Q02k4HZ9WGQ1wzw6uEzxgzn5tIRrHaZV";

        const apiUrl = `https://api.giphy.com/v1/gifs/random?api\_key=${apiKey}`;

        const apiRespuesta = await fetch( apiUrl );

        // Esta línea de comentario no se ejecuta hasta que se termine de ejecutar el await que esta en el fetch

        const { data } = await apiRespuesta.json();

        // Esta línea de comentario no se ejecuta hasta que se termine de ejecutar el await que esta en el apiRespuesta.json()

        const { url } = data.images.original;

        return url;

    } catch (error) {

        // Manejo del error...

        console.error( error );

    }

}

obtenerImagenAsyncAwait().then( imgUrl => {

    const img     = document.createElement("img");

          img.src = imgUrl;

    document.body.append(img);

})