UNIÓN EUROPEA
Fondo Social Europeo
El FSE invierte en tu futuro

MINISTERIO DE EDUCACIÓN Y FORMACIÓN PROFESIONAL

GENERALITAT VALENCIANA
Conselleria d'Educació, Investigació, Cultura i Esport

I.E.S. SERRA PERENXISA

DWEC 2º DAW

APIs / Fetch / Async Await

Curso 2.024-2025

BUSCADOR DE IMÁGENES

1. INTRODUCCIÓN RESUMEN

Partiendo de los ficheros html y css, daremos vida a una aplicación web que será un buscador de imágenes relacionadas con el tema concreto que indique el usuario. Añadiremos un paginador para ver las fotos de forma ordenada.

Se utilizará la siguiente API: https://pixabay.com/api/docs/

Se recomienda visitar la página y echar un vistazo a la documentación para ver los parámetros y los resultados que nos ofrece la API.

Estado inicial:

2. DESARROLLO → DATOS API

Para poder consumir datos de esta API, necesitamos una ***key*** que solo estará accesible si nos registramos (en la parte de "***Parameters***").

En la página indicada, en la parte de "***Example***", hay una url con lo que podemos inyectar para nuestra búsqueda. Nos puede servir de guía.

3. DESARROLLO → MOSTRAR INFO HTML

La **información** que queremos que nos muestre la página tendrá unos estilos que se indican a continuación. **Inyectaremos el html desde JavaScript**.

Creamos un div con class=”w-1/2 md:w-1/3 lg:w:1/4 mb-4 p-3”. Dentro de este div, creamos otro, con class=”bg-white”. Este segundo div será el que contenga la información:

- La foto en baja resolución (variable **previewURL**), class=”w-full”
- En un div con class=”p-4”:
    - Los likes de esa foto (variable **likes**). Usar class=”font-bold” y class=”font-light” según se necesite.
    - El número de veces que esa foto ha sido vista (variable **views**). Usar class=”font-bold” y class=”font-light” según se necesite.
    - Un enlace que nos llevará a la foto en alta resolución (variable **largeImageURL**). Utilizar las clases "block w-full bg-blue-800 hover:bg-blue-500 text-white uppercase font-bold text-center rounded mt-5 p-1”

4. DESARROLLO → MOSTRAR PAGINADOR

Por defecto, aparecen 20 imágenes, aunque hay muchas más disponibles. Podemos modificar esa cantidad en ***per_page***. Debemos modificar la url y añadir esta variable que valdrá, por ejemplo, 40 (puedes modificar la cantidad).

Para poder **gestionar todas las imágenes crearemos un paginador**. Para ello debemos tener en cuenta:
El número de imágenes que nos ofrece la API según el tema propuesto por el usuario (ese dato será diferente según sea la elección del usuario).
El número de imágenes que queremos que se muestren en cada página, en nuestro caso, 40.

Lo podemos hacer utilizando un **generador**. Es un proceso que consta de una función generadora que muestra un objeto iterable (Generator). Este proceso se puede detener, pausar y reanudar.
Se declaran a través de un ***delante del nombre de la función (también se puede poner detrás de la palabra reservada function).
Los resultados se obtienen con la palabra reservada **yield.**

Devuelve un objeto sobre el que podemos invocar el método **next()** para ir reanudando su ejecución. Es como si una vez que lanza un yield o resultado, el generador quedase en standby o dormido esperando despertar (lo hará con next()).
Ejemplo de generador:

```
function* counter() {
  var indice = 0;
  while(true)
    yield indice++;
}

var gen = counter();

console.log(gen.next().value); // 0
console.log(gen.next().value); // 1
console.log(gen.next().value); // 2
// ...
```

Los elementos del objeto que devuelven tienen dos propiedades: value (el valor de yield) y done (true o false).
Este último atributo será false mientras no se haya invocado el último elemento del objeto.
Para trabajar con los valores de generador, podemos hacer destructuring:
const {value, done} = counter.next();

Por cada página crearemos un botón con las siguientes clases: siguiente, bg-yellow-400, px-4, py-1, mr-2, Font-bold, mb-4, rounded.
Ejemplo de paginador:

Para redirigir a la página que el usuario clique, volveremos a hacer una petición a la API, con la ayuda del parámetro **page**.

Modifica el código para utilizar async/await.