Aarón Medina Rodríguez

2º DAW - Desarrollo Web en entorno cliente - DEW

2024 Q1

Enunciado

Selecciona una API publica de tu elección (puedes escoger una del repositorio: https://github.com/public-apis/public-apis) y genera una interfaz con los siguientes requisitos:

- Usa fecth para obtener los datos desde la API.
- Utilizar .map() para representar cada uno de los elementos obtenidos de la API. (Modificando el DOM, creando nuevos nodos, intentar evitar usar el innerHTML...)
- Utilizar .filter() para realizar busquedas de items.
- Utilizar Cookies, LocalStorage, sessionStorage para implementar una funcionalidad de tu interfaz.
- Añade 4 funcionalidades a tu aplicación web (a tu libre elección) usando javascript.

Resolucion

Se eligio la api de pokemon que contiene todos los pokemon *https://pokeapi.co/api/v2/pokemon/?offset=00&limit=15000*.

Utilizamos un doble fecth para obtener los datos que requerimos de la api. El primero obtiene de un objeto con el siguiente formato la url del pokemon concreto.

```
{
  "name": "bulbasaur",
  "url": "https://pokeapi.co/api/v2/pokemon/1/"
}
```

El segundo fetch obtiene el resto de datos del Pokémon principalmente extraemos los siguientes datos para trabajar en nuestra app. Los datos son almacenados en una array y en un SessionStorage

```
{
  "id": "pokemon.id",
  "sprite": "pokemon.sprites.front_default",
  "nombre": "pokemon.name",
  "tipos": "pokemon.types"
}
```

También utilizamos la api de *https://pokeapi.co/api/v2/type* para obtener los tipos de los Pokémon, de esta manera ahorramos un fetch anidado que puede dar problemas. Los datos son almacenados en una array y en un SessionStorage

Los datos que obtenemos de esta API los guardamos de la siguietne manera. Basicamente el id el la url y el nombre seria en español si se pudiera sino el default.

```
{
  "id": "tipo.url",
  "nombre": "data.names[5].name"
}
```

Durante la llamada a la API se muestra en el HTML una barra de carga respecto a los Pokémon obtenidos de la api con respecto a los que se esperaban.

Antes de imprimir los elementos se ordenan por id y se les cambian los tipos a su versión en español

Imprimimos cada elemento de la api individualmente en el HTML. Si no tienen una imagen definida en la API o ocurre un problema al cargar la imagen se aplicara una default.

Filtro por nombre del Pokémon.

Se guardan los datos en un sessionStorage para evitar llamadas repetidas a la api en la misma sesión.

Opcion para reimprimir los datos por si no cargaran las imágenes y opción para llamar a la api de nuevo, si ya se cargaron correctamente todos los datos de la API en nuestro array este botón no tendra efecto alguno.

Al realizar las llamadas a la API principal de la pokedex se realizan cada 1.5 segundos para asegurar que puedan cargar todos los Pokémon en la web, si no se hace este procedimiento pueden no guardarse en la web todos los Pokémon de la api. Además si superan mas de 30 segundos de realizando dichas llamadas a la api se finaliza dicha ejecución antes de tiempo, para evitar esperas elevadas y evitar posibles errores el en las condiciones de finalización de la llamada a la api.

Al pulsar en un Pokémon salta un mensaje con la info básica del Pokémon, su id, nombre y tipos.

Como se solicita en los requerimientos los el manejo de datos se hace principalmente con DOM el uso de innerHTML es para exclusivamente para los botones de recargar datos para limpiar rápidamente el contenedor de la pokedex.

También se añadió una flecha de navegación para volver al principio de la pagina mejorando la usabilidad de la pagina.

Aunque no fuera solicitado explícitamente en los requisitos se realizo una hoja de estilos básica para la interfaz.