

UD 06. PROGRAMACIÓN ASÍNCRONA

Desarrollo Web en entorno cliente CFGS DAW

Ejercicios

Álvaro Maceda Arranz

alvaro.maceda@ceedcv.es

2022/2023

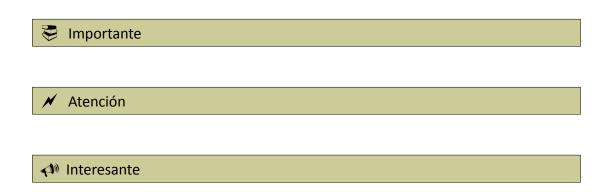
Versión:231214.1858

Licencia

Reconocimiento - NoComercial - Compartirlgual (by-nc-sa): No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

Nomenclatura

A lo largo de este tema se utilizarán distintos símbolos para distinguir elementos importantes dentro del contenido. Estos símbolos son:



ÍNDICE DE CONTENIDO

1. Ejercicio A	🗤: Información de usuario simple	3
	A2: Información de usuario completa	
3. Ejercicio A	A2: Información de múltiples usuarios	5

UD06. Programación Asíncrona

1. EJERCICIO A1: INFORMACIÓN DE USUARIO SIMPLE

En este ejercicio, tu tarea es simular la obtención de información de un usuario mediante una función asíncrona que devuelve una promesa. No utilices fetch en este caso, simplemente crea una función obtenerInformacionUsuario que simula una solicitud asíncrona.

- Crea una función llamada obtenerInformacionUsuario que toma un parámetro idUsuario (número entero) y devuelve una promesa.
- Dentro de la función, simula una demora asíncrona aleatoria entre 0.5 y 3 segundos antes de resolver o rechazar la promesa. Puedes usar setTimeout para simular esta demora.
- Si idUsuario es un número par, resuelve la promesa con un objeto que contenga la información del usuario, por ejemplo:

{ id: idUsuario, nombre: 'Usuario Par', tipo: 'Regular' }

 Si idUsuario es un número impar, rechaza la promesa con un mensaje de error, por ejemplo:

new Error('Error: Usuario Impar no permitido')

5. Implementa un ejemplo de uso de esta función. Llama a obtenerInformacionUsuario con un número par e imprime la información del usuario si la promesa se resuelve correctamente. Luego, llama a la función con un número impar e imprime el mensaje de error si la promesa es rechazada.

2. EJERCICIO A2: INFORMACIÓN DE USUARIO COMPLETA

En este ejercicio, practicaremos el uso de promesas para realizar llamadas paralelas a múltiples endpoints de una API ficticia. Supongamos que tenemos tres endpoints que proporcionan información sobre usuarios, posts y comentarios respectivamente. Tu tarea es obtener datos de estos tres endpoints en paralelo.

Crea dos funciones llamadas obtenerInfoUsuarios y obtenerInfoPosts. Cada una de ellas toma un parámetro id y devuelve una promesa.

Cada función simulará una llamada asíncrona a un endpoint diferente de una API ficticia. Usa setTimeout para simular una demora en cada llamada.

 obtenerInfoUsuarios(id): Simula obtener información sobre un usuario con el id proporcionado. Los usuarios tendrán este esquema:

```
{
  id: 1, nombre: 'Usuario 1'
}
```

 obtenerInfoPosts(id): Simula obtener información sobre los posts del usuario con el id proporcionado. Cada post tendrá los campos titulo y contenido:

```
[
    {id: 1, titulo: 'Post 1', contenido: 'Lorem ipsum dolor sit amet' },
    {id: 2, titulo: 'Post 2', contenido: 'consectetur adipisicing elit.' },
    ...
]
```

Cada una de estas funciones fallará aleatoriamente con una probabilidad del 5%.

Una vez tengas esas funciones crea una función llamada obtenerInformacionCompleta que toma un idUsuario y utiliza las funciones anteriores para obtener información de usuario, posts y comentarios en paralelo. La función debe devolver una promesa que se resuelva con un objeto que contenga la información completa. Por ejemplo:

```
{
  usuario: { id: idUsuario, nombre: 'Usuario Ejemplo' },
  posts: [...], // Array de posts
}
```

Implementa un ejemplo de uso de obtenerInformacionCompleta. Llama a esta función con un idUsuario y maneja la promesa resultante utilizando . then para imprimir la información completa si la promesa se resuelve correctamente, y utiliza .catch para manejar cualquier error.

3. EJERCICIO A2: INFORMACIÓN DE MÚLTIPLES USUARIOS

Utiliza la función obtenerInformacionCompleta del apartado anterior para obtener los datos de cinco usuarios al mismo tiempo. Cuando hayas obtenido la información de todos los usuarios muéstralos con sus datos completos por orden alfabético.