¿Cómo aseguramos manejar los errores asincrónicos correctamente?

try { ...código } catch (error) { ...código }

2.

¿Para qué nos sirve la clase XMLHttpRequest?

Nos permite realizar solicitudes HTTP de una forma muy fácil

3.

¿Cual es el método recomendando por la comunidad para manejar asincronismo en JavaScript?

Async await

4.

¿Cuál es la expresión que pausa la ejecución de la función así­ncrona y espera la resolución de la promise?

await

5.

¿Para qué nos sirve xhttp.status === 200?

Verificamos que el estatus de la petición HTTP resuelva el estado 200

6.

¿Para qué nos sirve el método "catch()"?

Registra la razón del rechazo

7.

¿El estado 4 de xhttp.readyState a qué hace referencia?

COMPLETED, La operación está terminada

8.

¿El método then() retorna?

Promesa

9.

¿Cuáles son los argumentos que recibe una promesa?

resolve, reject

10.

¿Para qué utilizamos `JSON.parse(xhttp.responseText)`?

Convertir una respuesta de texto en un Objecto iterable

11.

¿Nos permite definir una función así­ncrona?

async

12.

¿Cuál es la forma correcta de retornar un Error en reject?

reject (new Error(`Error`))

13.

¿Las promesas resuelven un principal problema de las callbacks?

callback hell

14.

¿La recomendación de la comunidad para anidar callbacks es?

Un mí­nimo de 3callbacks

15.

¿Nos permite ejecutar una serie de promesas secuencialmente?

Promise.all()