5.320 pts

Curso de Asincronismo con JavaScript

¡No te rindas!

Necesitas una **calificación mínima de 9.0** para aprobar.
Vuelve a intentarlo en 05 horas, 56 minutos, 35 segundos

8.67
Calificación

13 / 15
Aciertos

1. ¿Las promesas resuelven un principal problema de las callbacks?

callback hell ✓

2. ¿Para qué nos sirve la clase XMLHttpRequest?

Nos permite realizar solicitudes HTTP de una forma muy fácil ✓

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

COMPLETED, La operación está terminada ✓

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

Convertir una respuesta de texto en un Objecto iterable ✓

---

5. ¿La recomendación de la comunidad para anidar callbacks es?

Un máximo de 3 callbacks ✓

---

6. ¿Cuáles son los argumentos que recibe una promesa?

resolve, reject ✓

---

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

reject (new Error(`Error`)) ✓

---

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

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

---

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

Muestra el error en json de una promesa rechazada REPASAR CLASE

---

10. ¿El método then() retorna?

JSON REPASAR CLASE

---

11. ¿Nos permite ejecutar una serie de promesas secuencialmente?

Promise.all() ✓

12. ¿Nos permite definir una función asíncrona?

async ✓

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

await ✓

14. ¿Cuál es el método recomendado por la comunidad para manejar asincronismo en JavaScript?

Async/await ✓

15. ¿Cómo aseguramos manejar los errores asincrónicos correctamente?

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

REGRESAR