



Curso de Asincronismo con JavaScript 2019

¡No te rindas!

Necesitas una **calificación mínima de 9.0** para aprobar.

Vuelve a intentarlo en 05 horas, 47 minutos, 52 segundos

8

Calificación

12 / 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(xhr.responseText)`?

Convertir una respuesta de texto en un Arreglo inmutable

REPASAR CLASE

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 `xhr.status === 200`?

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



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

Registra la razón del rechazo



10. ¿El método `then()` retorna?

Promesa



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

Promise.all()



12. ¿Nos permite definir una función asíncrona?

promise

[REPASAR CLASE](#)

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?

promesas

[REPASAR CLASE](#)

15. ¿Cómo aseguramos manejar los errores asíncronicos correctamente?

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



[REGRESAR](#)