**Callback en JavaScript**

Un callback en JavaScript es una función que se pasa como argumento a otra función. Esta función interna se ejecuta en algún momento después de que la función externa haya terminado su ejecución. Esto permite un mayor control y flexibilidad en la programación asíncrona.

**¿Cuándo se usan los callbacks?**

* **Operaciones asíncronas:** Cuando se realizan tareas que no bloquean la ejecución del programa, como solicitudes de red, operaciones de entrada/salida o temporizadores.
* **Eventos:** Para manejar eventos del DOM, como clics, cambios de valor o carga de páginas.

**Ejemplo de callback:**

JavaScript

function greet(name, callback) { console.log("Hola, " + name + "!"); callback(); } function sayGoodbye() { console.log("Adiós!"); } greet("Juan", sayGoodbye);

Use code [with caution.](file:///C:/faq#coding)

En este ejemplo, greet es la función externa que toma un nombre y una función de callback. Dentro de greet, se imprime un saludo y luego se ejecuta la función de callback sayGoodbye.

**¿Por qué usar callbacks?**

* **Programación asíncrona:** Evitar bloqueos en la ejecución del programa.
* **Modularidad:** Separar la lógica de la tarea principal de la lógica de respuesta.
* **Flexibilidad:** Permitir diferentes acciones en respuesta a un evento o resultado.

**Consideraciones:**

* **Callback Hell:** Evitar anidar demasiados callbacks, lo que puede dificultar la legibilidad y el mantenimiento.
* **Promesas:** En JavaScript moderno, las promesas se utilizan a menudo como una alternativa a los callbacks para manejar operaciones asíncronas de manera más clara y con mejor manejo de errores.