**Event Loop**

El **event loop** es un componente central en el modelo de concurrencia de JavaScript, que permite la ejecución de código asíncrono sin bloquear el hilo principal. JavaScript es un lenguaje de programación de un solo hilo, lo que significa que solo puede ejecutar una tarea a la vez. El event loop es la clave para manejar múltiples tareas asíncronas en ese entorno de un solo hilo.

**Componentes Clave**

* **Call Stack (Pila de Llamadas):** Es donde se ejecuta el código JavaScript. Cuando se llama a una función, esta se coloca en la parte superior de la pila. Cuando se termina de ejecutar, se elimina de la pila. El stack sigue una estructura LIFO (Last In, First Out).
* **Web APIs:** Estas son APIs proporcionadas por el entorno de ejecución (como el navegador). Permiten ejecutar tareas asíncronas, como setTimeout, llamadas a la red (AJAX), y eventos del DOM. Cuando una función de estas APIs se completa, envía un callback a la cola de tareas.
* **Callback Queue (Cola de Tareas):** Aquí es donde se almacenan los callbacks de las tareas que están listas para ser ejecutadas. Si hay una tarea en esta cola, el event loop la mueve al call stack una vez que este está vacío.
* **Microtask Queue (Cola de Microtareas):** Es similar a la cola de tareas, pero tiene prioridad más alta. Las microtareas incluyen promesas (Promise), MutationObserver, etc. Las microtareas se procesan antes que las tareas normales.

**¿Para Qué Sirve el Event Loop?**

1. **Manejo de Asincronía:** Permite que JavaScript maneje múltiples operaciones de forma eficiente sin bloquear la ejecución. Esto es crucial para la experiencia del usuario en aplicaciones web.
2. **Interactividad:** Facilita la respuesta a eventos (como clics o desplazamientos) mientras otras tareas se procesan en segundo plano.
3. **Rendimiento:** Al evitar el bloqueo del hilo principal, las aplicaciones pueden ser más rápidas y reactivas.

**Diagrama

Descripción generada automáticamente**