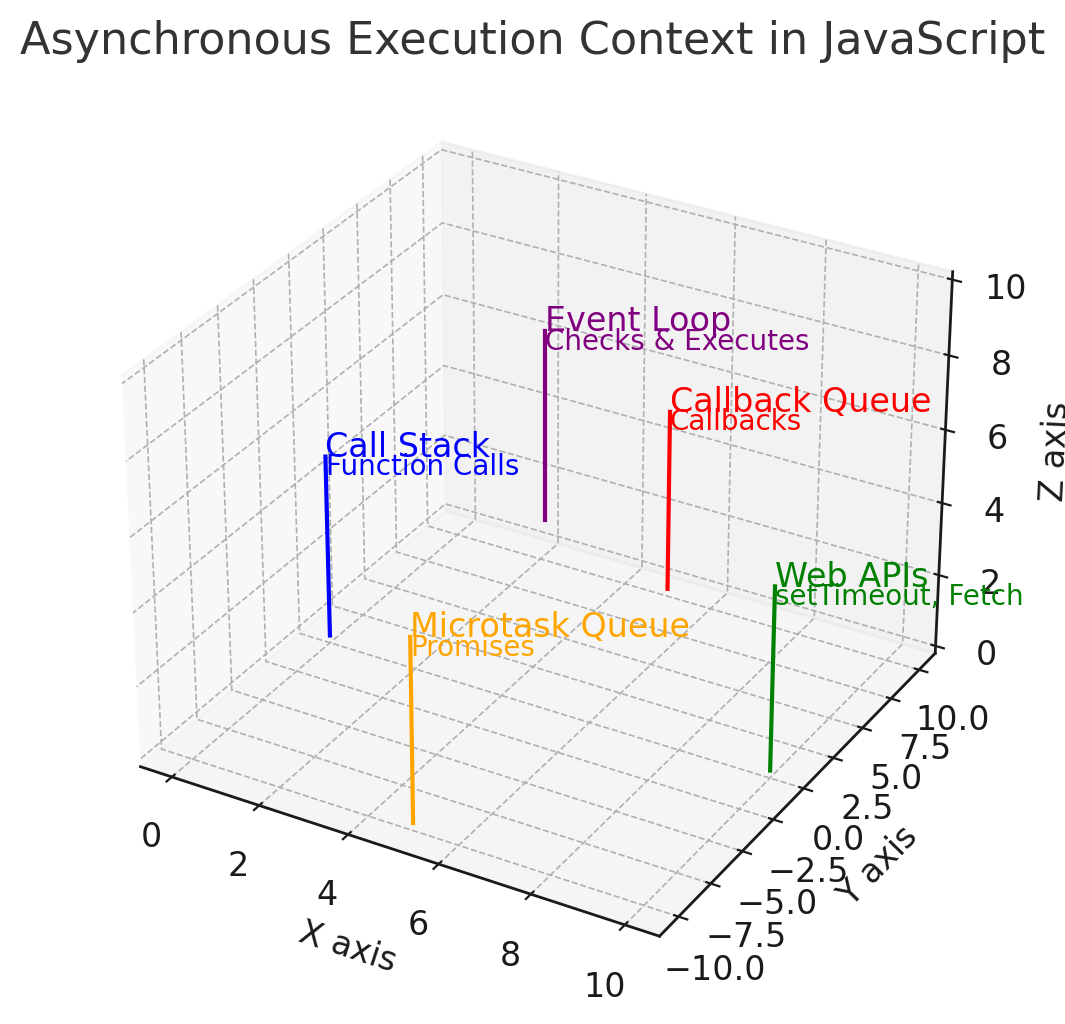
javascript engine is single threaded

We



Here's a 3D representation of the asynchronous execution context in JavaScript. The diagram visualizes the key components involved in asynchronous operations:

1. **Call Stack (Blue)**
   * Handles function calls and synchronous code execution.
   * Synchronous functions and operations are executed here.
2. **Web APIs (Green)**
   * Manages asynchronous operations like setTimeout, fetch, and event listeners.
   * After the operation is initiated, control returns to the call stack.
3. **Callback Queue (Red)**
   * Holds callbacks from asynchronous operations once they complete.
   * Callbacks are executed in a first-in, first-out order.
4. **Microtask Queue (Orange)**
   * Contains microtasks like promise resolutions and mutation observer callbacks.
   * Microtasks have a higher priority than the callback queue and are processed first.
5. **Event Loop (Purple)**
   * Continuously checks the call stack and, if empty, processes tasks from the microtask queue before moving to the callback queue.
   * Ensures smooth execution and responsiveness by managing the execution flow.

The diagram aims to illustrate how these components interact and how the event loop orchestrates the execution of asynchronous code, ensuring that microtasks are processed before callbacks. ​​

4o

Top of Form

Bottom of Form