Event Loops in JavaScript

www.scribbler.live

What is the Event Loop?

- JavaScript is single-threaded, meaning it can do one thing at a time.
- But it feels asynchronous because of the Event Loop, which manages:

  - Function calls
  - Web APIs (like setTimeout)
  - Promises, async/await

- It allows non-blocking operations despite having just one main thread.

How It Works ?

Code gets executed in the Call Stack (one after another).

If something takes time, it’s offloaded to Web APIs.

Once ready, the callback goes to the Task Queue.

The Event Loop again checks:

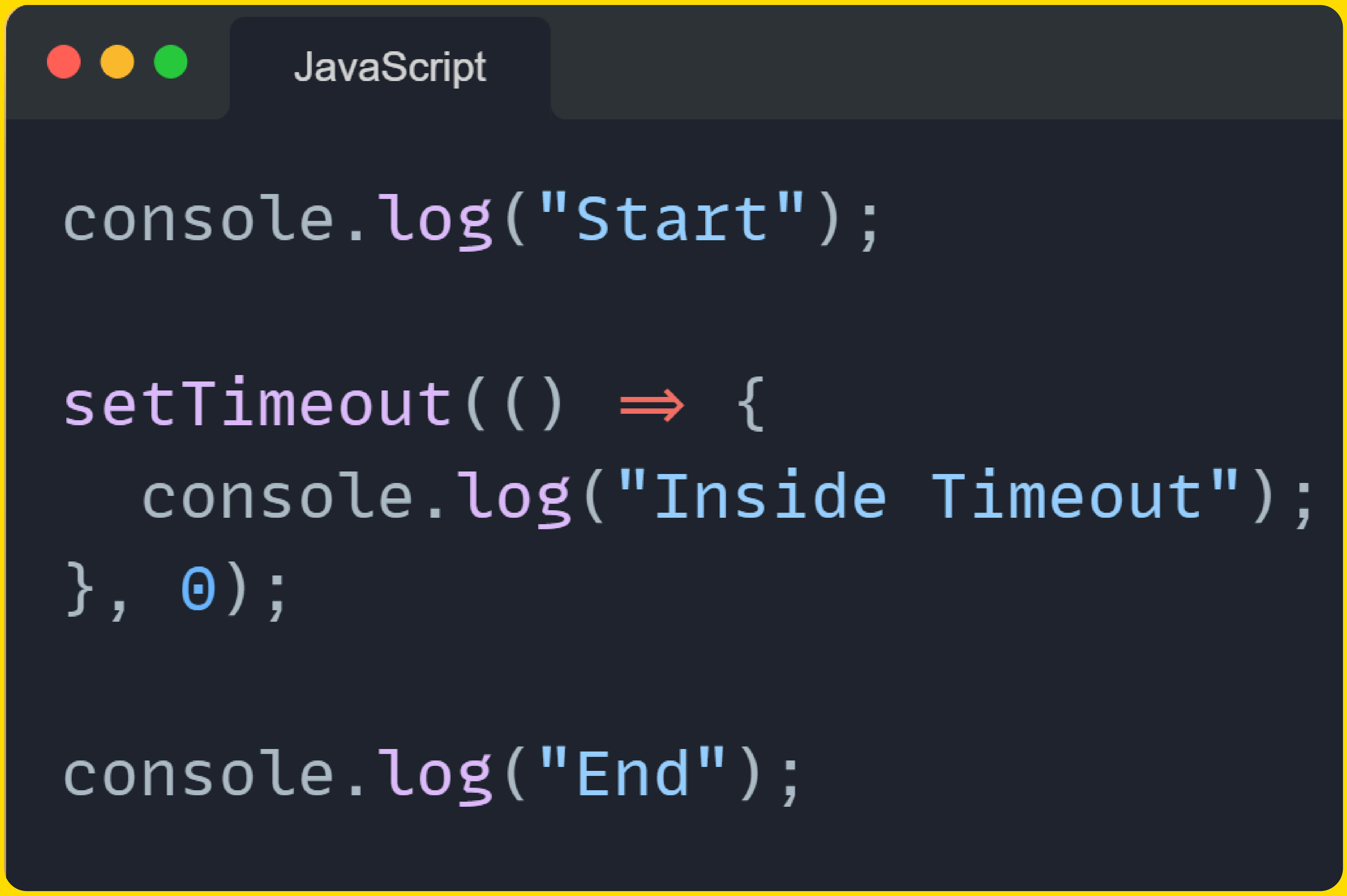
swipe

"Is the Call Stack empty?"

If yes - it pushes tasks from the Task Queue into the Call Stack.

Repeat — again and again, super fast!

Example



```javascript
console.log("Start");

setTimeout(() => {
  console.log("Inside Timeout");
}, 0);

console.log("End");
```

Why You Should Care?

Write non-blocking code

Understand async behavior better

Debug performance issues faster

Bonus Tip

- Microtasks (like Promise .then()) are prioritized before tasks from the task queue!
- Learn about the Microtask Queue for mastering modern async patterns.

Keep Exploring Javascript with us!

Share this with a friend who needs it and make sure to practice these in scribbler.

Scribbler.live

Free and Open Interface to experiment JavaScript