**SetTimeOut():**

The first two arguments to the function [setTimeout](https://developer.mozilla.org/en-US/docs/Web/API/setTimeout) are a message to add to the queue and a time value (optional; defaults to 0). The time value represents the (minimum) delay after which the message will be pushed into the queue. If there is no other message in the queue, and the stack is empty, the message is processed right after the delay. However, if there are messages, the setTimeout message will have to wait for other messages to be processed. For this reason, the second argument indicates a minimum time — not a guaranteed time.

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Event_loop>

<https://dev.to/debadeepsen/the-thing-about-settimeout-g9b>

From [MDN setTimeout()](https://developer.mozilla.org/en-US/docs/Web/API/WindowTimers/setTimeout#Explanation):

Code executed by setTimeout() is run in a separate execution context to the function from which it was called.

[**Does setTimeout() really execute in parallel?**](https://softwareengineering.stackexchange.com/questions/314773/does-settimeout-really-execute-in-parallel)

But does this mean it executes in parallel to any other code that is currently in process?

No, it does not mean that. It will just execute later, when the given time has passed.

Javascript fundamentally does not support multithreading because there are no language facilities to prevent data corruption from simultaneous access nor for having isolated threads communicate via messaging.

A Javascript engine simply processes a queue of events sequentially on a single thread. When the event queue is empty, that thread idles. In a browser, events are added to the queue by user input, page loading, etc. In node.js events can also be HTTP requests or hardware events. And setTimeout() simply adds another kind of event with the additional condition that it should only be added to the queue after a certain time has passed.

The "separate execution context" mentioned in the documentation just means that the this reference will be different than in the function where setTimeout() is called.

[**https://softwareengineering.stackexchange.com/questions/314773/does-settimeout-really-execute-in-parallel**](https://softwareengineering.stackexchange.com/questions/314773/does-settimeout-really-execute-in-parallel)

**2 interview question**

[**https://medium.com/@svetabuben/javascript-settimeout-under-the-hood-technical-literacy-dc70edbc2698**](https://medium.com/@svetabuben/javascript-settimeout-under-the-hood-technical-literacy-dc70edbc2698)