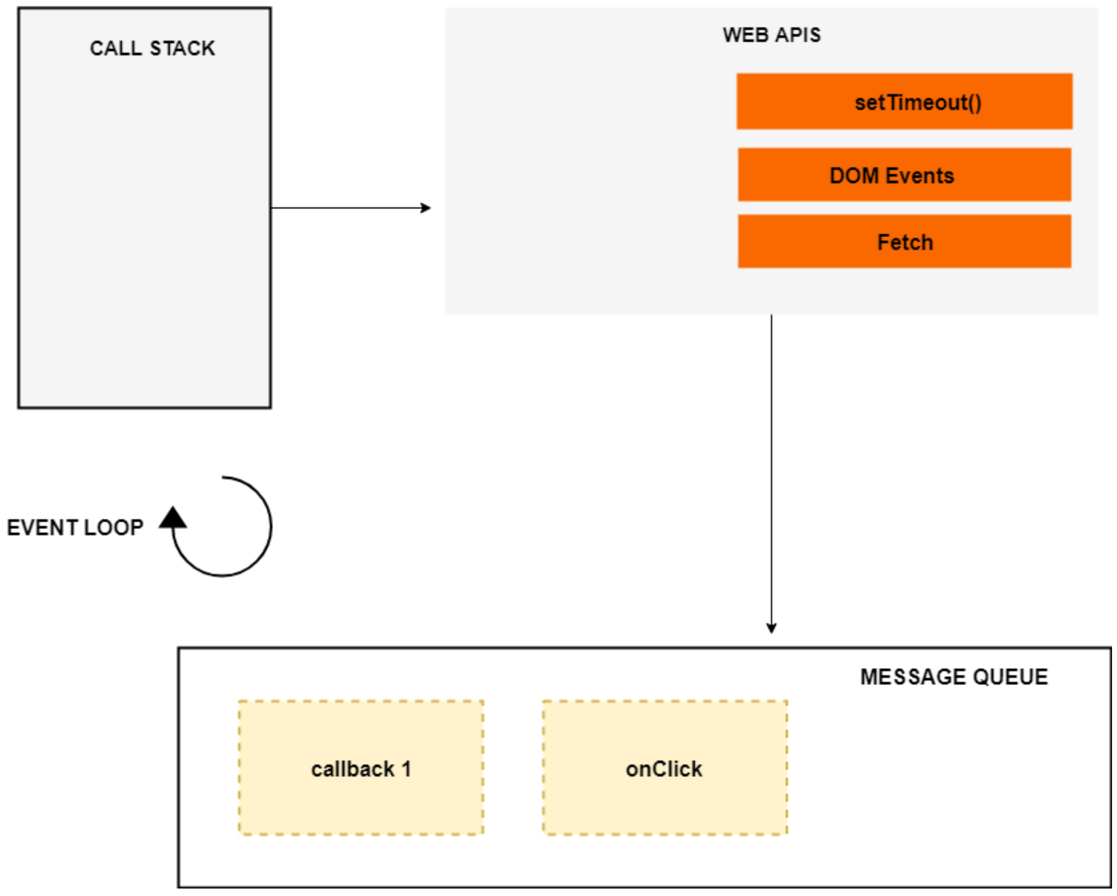
Understand Asynchronous

1. JavaScript engine is a single-threaded
2. Execution context: where JavaScript code is executed. The function code executes inside a function execution context. The global code executes inside a global execution context.
3. Call stack is a stack with a LIFO (Last in, First out) structure, which is used to store all execution context created during the code execution
4. setTimeout function is not a part of JavaScript engine. It's a part of something known as web APIs (in web browser) or C/C++ APIs (in node.js)
5. Browser's JavaScript runtime environment



6. Event Loop, web APIs and callback Queue are not part of JavaScript engine. It's a part of browser's JavaScript runtime environment or NodeJS runtime environment. In Nodejs, web APIs are replaced by C/C++ APIs
7. Callback queue is a FIFO structure
8. Event loop will look into the call stack to determine if the call stack is empty or not. If the call stack is empty, it looks into the callback queue to see whether any pending callback waiting to be executed
9. Suppose there's a callback in callback queue, event loop will push the callback to the top of the call stack
10. Callback queue can contain callbacks from DOM events (click event, keyboards event …)
11. ES6 Job Queue/ Micro-task Queue. ES6 introduced the concept of Job Queue/ Micro-task Queue which is used by Promise in JavaScript.
12. The difference between the callback Queue and Job Queue is that the Job Queue has a higher priority than the callback Queue, which means the promise Jobs inside the Job Queue will be executed before the callbacks inside the callback queue

To learn more and get OneNote, visit: https://blog.bitsrc.io/understanding-asynchronous-javascript-the-event-loop-74cd408419ff