1. JavaScript is single-threaded, and the asynchronous behaviour is not part of the JavaScript language itself, rather they are built on top of the core JavaScript language in the browser (or the programming environment) and accessed through the browser APIs. For example, Web API performed asynchronous tasks using setTimeout or setInterval.

In the main JavaScript thread, there is an endlessly running single threaded loop which called event loop and it accepts call-back function and execute them on the main thread. In the event loop there is microtask queue where the call functions are wating to be executed when the main thread is not busy. For example, promises in java script, have asynchronous and synchronous parts. Only the parts of resolve and reject are going to be asynchronous and other parts are synchronous. Call-back function from promises has higher priority than other call-back function in microtask queue.

1. In order to change synchronous into asynchronous, we can us JavaScript Timers Both setTimeout or setInterval schedule a function to execute after a specified delay. Then call back function will be waiting in the microtask to be executed when event loop pushes it to the stack when the main method is empty.