JavaScript

An *asynchronous operation* is one that allows the computer to “move on” to other tasks while waiting for the asynchronous operation to complete.  Asynchronous programming means that time-consuming operations don’t have to bring everything else in our programs to a halt.

Promises are objects that represent the eventual outcome of an asynchronous operation.

 A Promise object can be in one of three states:

* **Pending**: The initial state— the operation has not completed yet.
* **Fulfilled**: The operation has completed successfully and the promise now has a *resolved value*. For example, a request’s promise might resolve with a JSON object as its value.
* **Rejected**: The operation has failed and the promise has a reason for the failure. This reason is usually an Error of some kind.

We refer to a promise as *settled* if it is no longer pending.

setTimeout() is a Node API (a comparable API is provided by web browsers) that uses callback functions to schedule tasks to be performed after a delay. setTimeout() has two parameters: a callback function and a delay in milliseconds.