A callback is a function that is passed as an argument to another function, and then called in that function to perform a task.

**Call back:** *"I will call back later!"*

A callback is a function passed as an argument to another function.

This technique allows a function to call another function.

A callback function can run after another function has finished.

### Why do we use callbacks?

When doing a complex task, we break that task down into smaller steps. To help us establish a relationship between these steps according to time (optional) and order, we use callbacks.

<https://www.freecodecamp.org/news/javascript-async-await-tutorial-learn-callbacks-promises-async-await-by-making-icecream/>

**Why do we need Callback Functions?**

JavaScript runs code sequentially in top-down order. However, there are some cases that code runs (or must run) after something else happens and also not sequentially. This is called **asynchronous programming.**

Callbacks make sure that a function is not going to run before a task is completed but will run right after the task has completed. It helps us develop asynchronous JavaScript code and keeps us safe from problems and errors.

<https://www.freecodecamp.org/news/javascript-callback-functions-what-are-callbacks-in-js-and-how-to-use-them/>

**Can we do same with simple function???**

Yes we can do call different function that will be invoked after some task happens. But 2 problems will occur 1- we have to call 2 functions if we use single function then 2- we can’t fully control over a function if we call first second will automatically call despite **we want or not.**

[**https://www.w3schools.com/js/js\_callback.asp**](https://www.w3schools.com/js/js_callback.asp)

**Comparison with simple function:**

In simple function we call but two problems will be faced described above:

1. In call back we can’t mutate the parent function definition as we did in w3 school example of myCalculator.
2. ~~With call back function the main function become~~ **~~high order function~~** ~~which make it more efficient and fast.~~

**Types of the Callbacks or call back function:**

Synchronous and Asynchronous callbacks:

synchronous callbacks are executed immediately, whereas the execution of asynchronous callbacks is deferred to a later point in time.

<https://www.javascripttutorial.net/javascript-callback/>

synchronous callbacks are executed immediately, whereas the execution of asynchronous callbacks is deferred to a later point in time.

[How can you tell if a callback is synchronous or asynchronous?](https://maximorlov.com/synchronous-vs-asynchronous-callbacks/#how-can-you-tell-if-a-callback-is-synchronous-or-asynchronous)

Whether a callback is executed synchronously or asynchronously depends on the function which calls it. If the function is asynchronous, then the callback is asynchronous too.

<https://maximorlov.com/synchronous-vs-asynchronous-callbacks/>

**if we have multiple callbacks or nested callback:**

Nesting many asynchronous functions inside callbacks is known as the **pyramid of doom** or the **callback hell**.  
To avoid the pyramid of doom, you use [promises](https://www.javascripttutorial.net/es6/javascript-promises/) or [async/await](https://www.javascripttutorial.net/es-next/javascript-async-await/) functions.

<https://www.javascripttutorial.net/javascript-callback/>

**How to achieve asynchronous with CallBacks?**

"Does taking a callback make a function asynchronous?"

Well, basically yes, but there's a catch. Every asynchronous function takes a function argument, but not every function that does so is asynchronous. It matters how the argument is used inside the function.