

# Callback Function in JavaScript

## Introduction

While writing a program for a small-scale application to an enterprise-level application function plays an important role. The functions keep the program organized, easy to understand and makes it reusable and increase the readability of a program. In this tutorial, we are discussing the callback function with an example.

## Callback Functions

The callback function is a function which is passed as an argument to another function. A callback function executes within the function to which it is passed. Callback functions are also known as high-order functions.

## Writing a Callback Function

```
let functionOne = function () {  
    console.log('I am a function, I am called inside a function');  
}  
  
let functionTwo = function(callback) {  
    console.log('I have a callback function');  
    callback();  
}  
  
functionTwo(functionOne);
```

Let's dive into the above example. We have defined two functions functionOne and functionTwo. functionOne is a function without any parameter whereas functionTwo accepts a callback as a parameter. When functionTwo is called we pass functionOne as an argument. The output is as follows.

I have a callback function

I am a function, I am called inside a function

functionOne will only execute after functionTwo completes its execution. Thus functionTwo is a callback function which accepts a function as a parameter. Callback helps in asynchronous programming, which simply means executing something after something else has been executed as in the below example.

```
setTimeout(function() => {  
    console.log('Hello');  
}, 5000);
```

The above code prints the string Hello in the terminal after five seconds.

This article aims to provide an introduction to callback functions for beginners in JavaScript. I hope the concept of the callback is clear.