

# JavaScript III

---

functions

# Functions

A JavaScript function is a block of code designed to perform a particular task.

```
function myFunction(p1, p2) {  
    return p1 * p2;    // The function returns the product of p1 and p2  
}
```

A JavaScript function is executed when "something" invokes it (calls it).

```
myFunction(4, 3);
```

# Syntax

There are a few ways to use and declare functions in JavaScript, including function expression, concise expression, arrow expression, etc.

This is the most basic and common function declaration:

```
FUNCTION  
KEYWORD      IDENTIFIER  
|             |  
function greetWorld() {  
    console.log('Hello, World!');  
}
```

KEY

● Function body

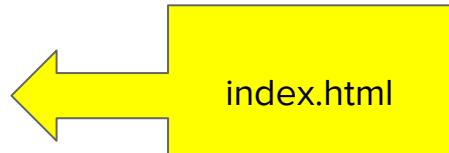
A function declaration consists of:

- The `function` keyword.
- The name of the function, or its identifier, followed by parentheses.
- A function body, or the block of statements required to perform a specific task, enclosed in the function's curly brackets, `{ }`.

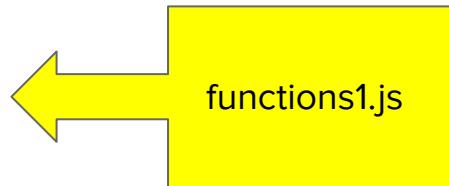
# Basic Example

Let's write a Hello World function:

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>JS Functions</title>
5  </head>
6
7  <body>
8
9  <div id="section1">
10 </div>
11
12 <script type="text/javascript" src="functions1.js"></script>
13
14 </body>
15 </html>
```



```
1  function HelloWorld(){
2      document.getElementById("section1").innerHTML = "Hello World!";
3  }
```



# Calling a function

In order to execute the code in a function we have to call it.

We can call a function just by referring to it's name and adding an opening and closing parentheses + ending semi-colon:



A screenshot of a code editor interface. On the left, there is a dark grey sidebar with the number '5' displayed in white. To the right of the sidebar, there is a large, semi-transparent blue rectangular overlay. Inside this overlay, the text 'myFunction();' is written in a light blue font. The background of the code editor shows some faint, illegible code.

# Call your function

In your “functions1.js” call your Hello World function.  
And preview the “index.html” file to see how it works.

## Now, delete the HelloWorld(); function call from functions1.js

Now, let's trigger the function call with a button click, Create a “functions1.html” file:

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>functions1</title>
5  </head>
6  <body>
7
8      <div id="section1">
9
10     </div>
11
12     <button onclick="HelloWorld();>submit</button>
13
14
15     <script type="text/javascript" src="functions1.js"></script>
16
17 </body>
18 </html>
```

# \* hoisting

The convention in most C-based languages is that functions have to be defined and declared before they can be called. JavaScript is an exception to this rule.

The **hoisting** feature in JavaScript which allows access to function declarations before they're defined.

```
1  HelloWorld();  
2  
3  function HelloWorld(){  
4      document.getElementById("section1").innerHTML = "Hello World!";  
5 }
```

A large yellow starburst graphic with several points, positioned to the right of the code snippet. It contains the text "This works too!" in black.

# Your turn

- In your JS file create another function called `getReminder()`
- This function should use the `.innerHTML` method to output the string ‘Water the Plants’ into a div with the id of “section2”. (You’ll need to create this div in your `.html` file.)
- In your JS file create another function called `greetInSpanish()`
- This function should use the `.innerHTML` method to display an appropriate image into a div with the id of “section3”. (You’ll need to create this div in your `.html` file.)
- In your html, create three more buttons to trigger each of these functions.

# Functional utility

One of the reasons functions are powerful and important concepts in programming is because we can wrap up complicated or laborious code as a single function and call it as many times as we need to. Let's demo this in a functions2.js file:

```
1  function sayThanks(){
2    console.log('Thank you for your purchase! We appreciate your business.');
3  }
4  sayThanks();
5  sayThanks();
6  sayThanks();
```

