**Functions in JavaScript | Web Development Tutorials #53**

In this tutorial, we are going to learn about the JavaScript functions and why should we use them in JavaScript. A JavaScript function is a block of code designed to perform a particular task. They are one of the building blocks in JavaScript.

Make a new file as *tut53.html*and add the boilerplate for the HTML template. In this tutorial, we will create a new JavaScript file and attach it in the <script> tag as follows-

<script src= “tut53.js”></script>

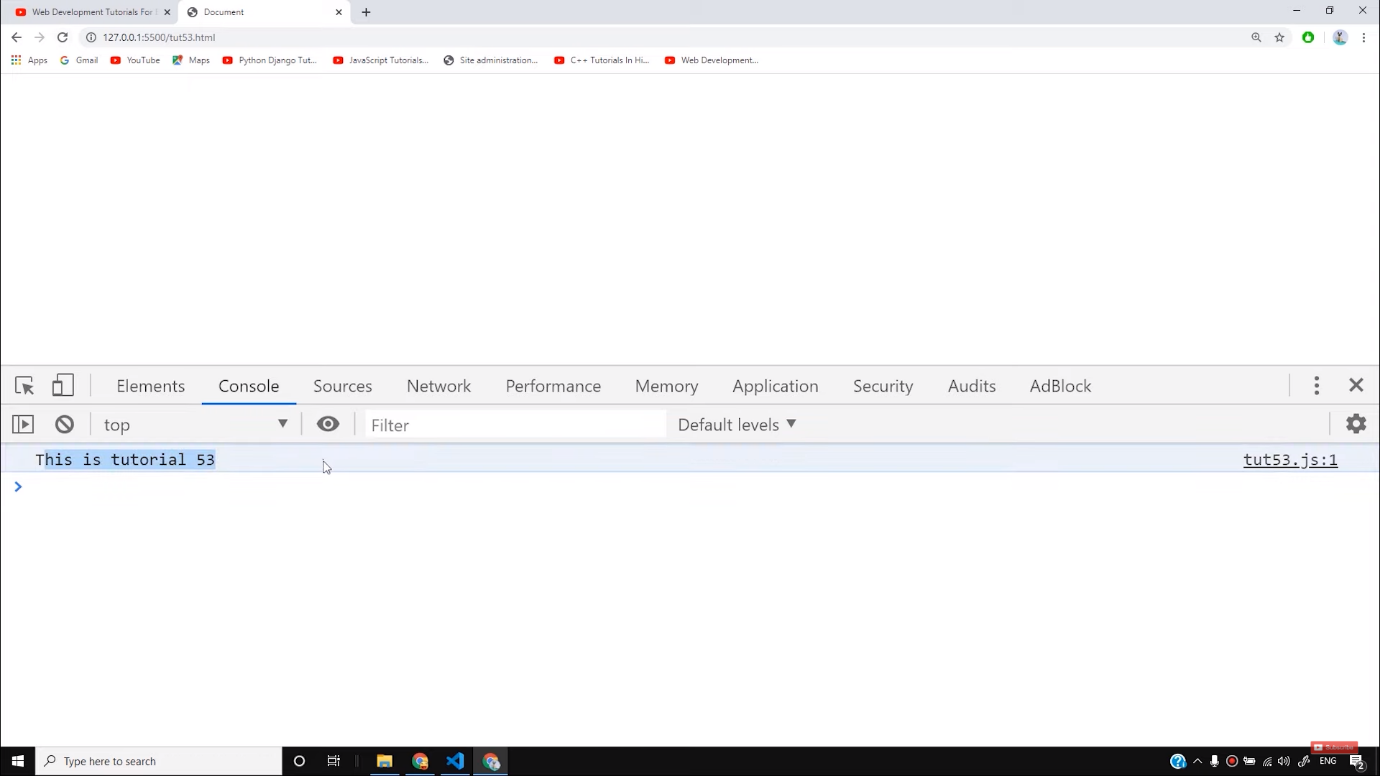
Copy

To check whether our js file is attached or not, we can write

console.log("This is tutorial 53");

Copy

If we get the result as follows, it means we are working correctly:



Let us now see how JavaScript functions help us. Assume that we have to repeatedly write the particular code for a certain task as shown below-

let name = "Harry";

let name1 = "Shubham";

let name2 = "Rohan";

let name3 = "Sagar";

console.log(name + “is a good boy”);

console.log(name1 + “is a good boy”);

console.log(name2 + “is a good boy”);

console.log(name3 + “is a good boy”);

Copy

As we can see in the above example, we are writing a particular set of codes every time. To avoid this JavaScript functions come into action. We can write the above code with the help of JavaScript functions as follows-

function greet(name){

console.log(name + " is a good boy");

}

let name = "Harry";

let name1 = "Shubham";

let name2 = "Rohan";

let name3 = "Sagar";

greet(name);

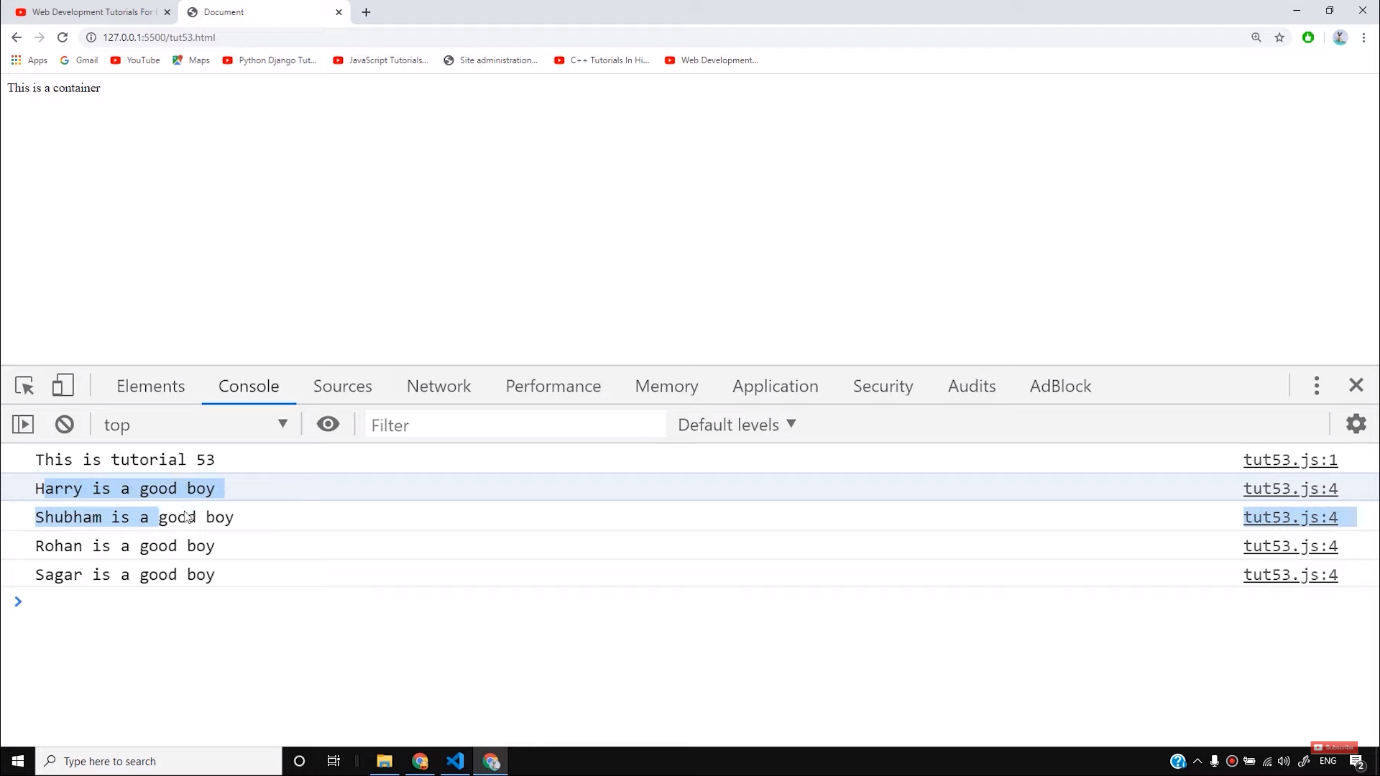
greet(name1);

greet(name2);

greet(name3);

Copy

By writing the above code, we will get the same output as follows-



We have increased the use of code reusability with the help of a JavaScript function. Also, we can increase more number of parameters in the JavaScript functions without writing the code again. Let us see the below code-

function greet(name, greetText){

console.log(greetText + " " + name);

console.log(name + " is a good boy");

}

let name = "Harry";

let name1 = "Shubham";

let name2 = "Rohan";

let name3 = "Sagar";

let greetText = "Good Morning";

greet(name, greetText);

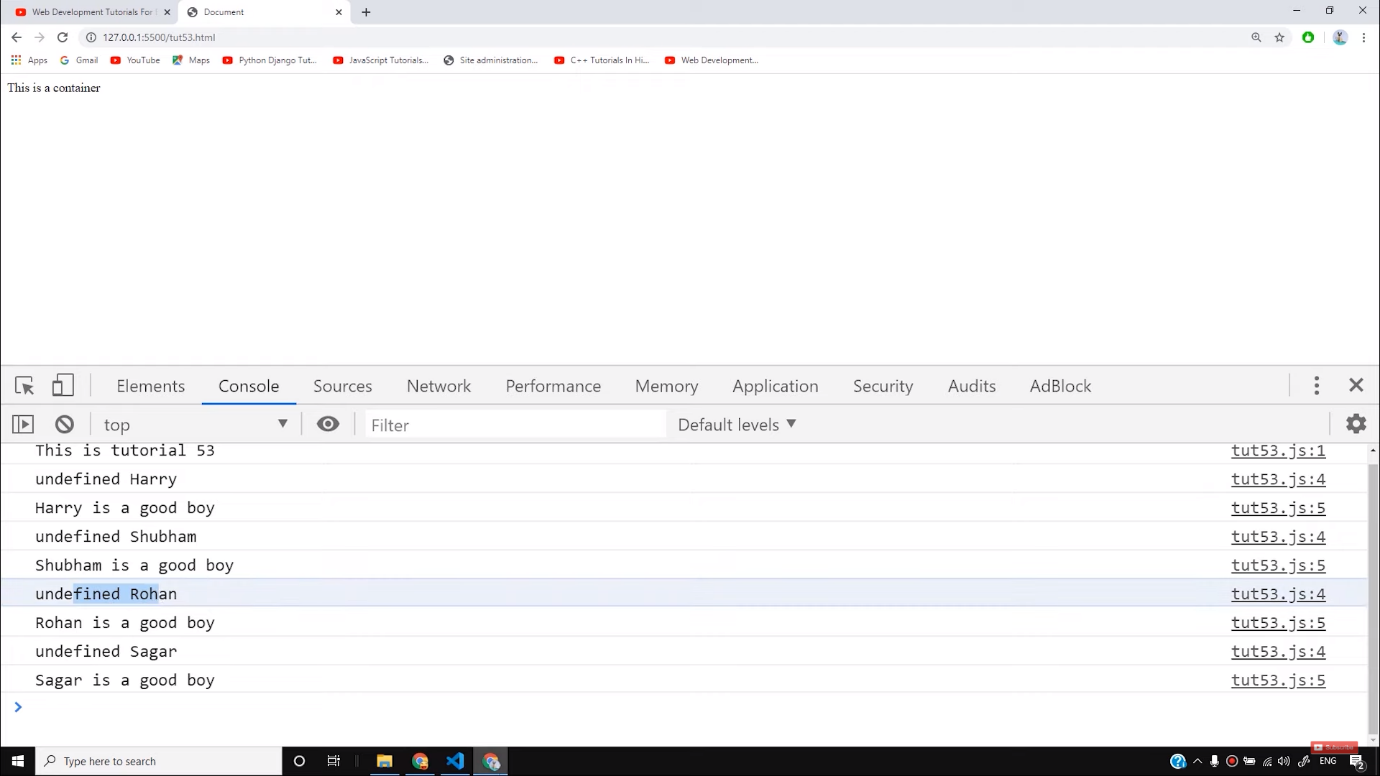
greet(name1, greetText);

greet(name2, greetText);

greet(name3, greetText);

Copy

We have to pass “greetText ” also in the function to make it executed. The function will only be called till you pass the value to it otherwise it will never be called. The output of the above code will be as follows-



We also use some functions to make it return a particular value. Return value means it gives you some output in return. For example, if we write-

let returnVal = greet(name3);

console.log(returnVal)

Copy

The output will be ***undefined***because the parameter *greet* does not return anything. It just prints the value through console.log. But if we make a new function as follows-

function sum(a,b,c){

let d = a + b + c;

return d;

}

let returnVal = sum(1,2,3);

console.log(returnVal)

Copy

You will get the output **6**here because the variable d is being returned here by the function.

So I hope you must have got some idea about JavaScript functions. Try to make some new functions by yourself like finding a maximum or minimum number in a group and keep practicing to make yourself better.

**Code as described/written in the video**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

</head>

<body>

<div class="container">

This is a container

</div>

</body>

<script>

console.log("This is tutorial 53");

function greet(name, greetText="Greetings from JavaScript"){

let name1 = "Name1";

console.log(greetText + " " + name);

console.log(name + " is a good boy");

}

function sum(a,b,c){

let d = a + b + c;

return d;

// This line will never execute (Unreachable code)

// console.log("Function is returned");

}

let name = "Harry";

let name1 = "Shubham";

let name2 = "Rohan";

let name3 = "Sagar";

let greetText1 = "Good Morning";

greet(name, greetText1);

greet(name1, greetText1);

greet(name2, greetText1);

// let returnVal = greet(name3);

// console.log(returnVal)

let returnVal = sum(1,2,3);

console.log(returnVal)

// console.log(name + " is a good boy");

// console.log(name1 + " is a good boy");

// console.log(name2 + " is a good boy");

// console.log(name3 + " is a good boy");

</script>

</html>

# JavaScript Functions

[❮ Previous](https://www.w3schools.com/js/js_datatypes.asp)[Next ❯](https://www.w3schools.com/js/js_objects.asp)

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

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

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_functions)

## **JavaScript Function Syntax**

A JavaScript function is defined with the function keyword, followed by a **name**, followed by parentheses **()**.

Function names can contain letters, digits, underscores, and dollar signs (same rules as variables).

The parentheses may include parameter names separated by commas:  
**(parameter1, parameter2, ...)**

The code to be executed, by the function, is placed inside curly brackets: **{}**

function name(parameter1, parameter2, parameter3) {  
  // code to be executed  
}

Function **parameters** are listed inside the parentheses () in the function definition.

Function **arguments** are the **values** received by the function when it is invoked.

Inside the function, the arguments (the parameters) behave as local variables.

A Function is much the same as a Procedure or a Subroutine, in other programming languages.

## **Function Invocation**

The code inside the function will execute when "something" **invokes** (calls) the function:

* When an event occurs (when a user clicks a button)
* When it is invoked (called) from JavaScript code
* Automatically (self invoked)

You will learn a lot more about function invocation later in this tutorial.

ADVERTISEMENT

## **Function Return**

When JavaScript reaches a return statement, the function will stop executing.

If the function was invoked from a statement, JavaScript will "return" to execute the code after the invoking statement.

Functions often compute a **return value**. The return value is "returned" back to the "caller":

### **Example**

Calculate the product of two numbers, and return the result:

let x = myFunction(4, 3);   // Function is called, return value will end up in x  
  
function myFunction(a, b) {  
  return a \* b;             // Function returns the product of a and b  
}

The result in x will be:

12

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_function_return)

## **Why Functions?**

You can reuse code: Define the code once, and use it many times.

You can use the same code many times with different arguments, to produce different results.

### **Example**

Convert Fahrenheit to Celsius:

function toCelsius(fahrenheit) {  
  return (5/9) \* (fahrenheit-32);  
}  
document.getElementById("demo").innerHTML = toCelsius(77);

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_farenheit_to_celsius)

## **The () Operator Invokes the Function**

Using the example above, toCelsius refers to the function object, and toCelsius() refers to the function result.

Accessing a function without () will return the function object instead of the function result.

### **Example**

function toCelsius(fahrenheit) {  
  return (5/9) \* (fahrenheit-32);  
}  
document.getElementById("demo").innerHTML = toCelsius;

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_farenheit_to_celsius_2)

## **Functions Used as Variable Values**

Functions can be used the same way as you use variables, in all types of formulas, assignments, and calculations.

### **Example**

Instead of using a variable to store the return value of a function:

let x = toCelsius(77);  
let text = "The temperature is " + x + " Celsius";

You can use the function directly, as a variable value:

let text = "The temperature is " + toCelsius(77) + " Celsius";

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_function_variable)

You will learn a lot more about functions later in this tutorial.

## **Local Variables**

Variables declared within a JavaScript function, become **LOCAL** to the function.

Local variables can only be accessed from within the function.

### **Example**

// code here can NOT use carName  
  
function myFunction() {  
  let carName = "Volvo";  
  // code here CAN use carName  
}  
  
// code here can NOT use carName