# JavaScript Functions

**JavaScript functions** are used to perform operations. We can call JavaScript function many times to reuse the code.

#### **Advantage of JavaScript function**

There are mainly two advantages of JavaScript functions.

1. **Code reusability**: We can call a function several times so it save coding.
2. **Less coding**: It makes our program compact. We don’t need to write many lines of code each time to perform a common task.

## JavaScript Function Syntax

The syntax of declaring function is given below.

1. function functionName([arg1, arg2, ...argN]){
2. //code to be executed
3. }

JavaScript Functions can have 0 or more arguments.

## JavaScript Function Example

Let’s see the simple example of function in JavaScript that does not has arguments.

1. **<script>**
2. function msg(){
3. alert("hello! this is message");
4. }
5. **</script>**
6. **<input** type="button" onclick="msg()" value="call function"**/>**

#### **Output of the above example**

## JavaScript Function Arguments

We can call function by passing arguments. Let’s see the example of function that has one argument.

1. **<script>**
2. function getcube(number){
3. alert(number\*number\*number);
4. }
5. **</script>**
6. **<form>**
7. **<input** type="button" value="click" onclick="getcube(4)"**/>**
8. **</form>**

#### **Output of the above example**

Top of Form

Bottom of Form

## Function with Return Value

We can call function that returns a value and use it in our program. Let’s see the example of function that returns value.

1. **<script>**
2. function getInfo(){
3. return "hello javavscript! How r u?";
4. }
5. **</script>**
6. **<script>**
7. document.write(getInfo());
8. **</script>**

#### **Output of the above example**

## JavaScript Function Object

In JavaScript, the purpose of **Function constructor** is to create a new Function object. It executes the code globally. However, if we call the constructor directly, a function is created dynamically but in an unsecured way.

## Syntax

1. new Function ([arg1[, arg2[, ....argn]],] functionBody)

## Parameter

**arg1, arg2, .... , argn** - It represents the argument used by function.

**functionBody** - It represents the function definition.

## JavaScript Function Methods

Let's see function methods with description.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [apply()](https://www.javatpoint.com/javascript-function-apply-method) | It is used to call a function contains this value and a single array of arguments. |
| [bind()](https://www.javatpoint.com/javascript-function-bind-method) | It is used to create a new function. |
| [call()](https://www.javatpoint.com/javascript-function-call-method) | It is used to call a function contains this value and an argument list. |
| [toString()](https://www.javatpoint.com/javascript-function-tostring-method) | It returns the result in a form of a string. |

## JavaScript Function Object Examples

### **Example 1**

Let's see an example to display the sum of given numbers.

### **Example 2**

Let's see an example to display the power of provided value.

**<script>**

var pow=new Function("num1","num2","return Math.pow(num1,num2)");

document.writeln(pow(2,3));

**</script>**

JavaScript Function bind() method

The JavaScript Function **bind()** method is used to create a new function. When a function is called, it has its own this keyword set to the provided value, with a given sequence of arguments.

Syntax

1. function.bind(thisArg [, arg1[, arg2[, ...]]]

Parameter

**thisArg** - The this value passed to the target function.

**arg1,arg2,....,argn** - It represents the arguments for the function.

Return Value

It returns the replica of the given function along provided this value and initial arguments.

## JavaScript Function bind() method Example

### **Example 1**

Let's see a simple example of bind() method.

<!DOCTYPE html>

<html>

<body>

<script>

var website = {

name: "meena",

getName: function() {

return this.name;

}

}

var unboundGetName = website.getName;

var boundGetName = unboundGetName.bind(website);

document.writeln(boundGetName());

</script>

</body>

</html>

### **Example 2**

Let's see an example of bind() method.

<!DOCTYPE html>

<html>

<body>

<script>

// Here, this refers to global "window" object

this.name = "Oracle";

var website = {

name: "Javatscript",

getName: function() { return this.name; }

};

document.writeln(website.getName()); // Javatpoint

//It invokes at global scope

var retrieveName = website.getName;

document.writeln(retrieveName()); //Oracle

var boundGetName = retrieveName.bind(website);

document.writeln(boundGetName()); // Java

</script>

</body>

</html>

# JavaScript Function apply() method

The JavaScript Function **apply()** method is used to call a function contains this value and an argument contains elements of an array. Unlike call() method, it contains the single array of arguments.

## Syntax

1. function.apply(thisArg, [array])

## Parameter

**thisArg** - It is optional. The this value is given for the call to a function.

**array** - It is optional. It is an array-like object.

## JavaScript Function apply() method Example

### **Example 1**

Let's see an example to determine the maximum element.

1. **<script>**
2. var arr = [7, 5, 9, 1];
3. var max = Math.max.apply(null, arr);
4. document.writeln(max);
5. **</script>**

**Output:**

9

### **Example 2**

Let's see an example to determine the minimum element.

1. **<script>**
2. var arr = [7, 5, 9, 1];
3. var min = Math.min.apply(null, arr);
4. document.writeln(min);
5. **</script>**

**Output:**

1

### **Example 3**

Let's see an example to join arrays of same type.

1. **<script>**
2. var array = [1,2,3,4];
3. var newarray=[5,6,7,8]
4. array.push.apply(array, newarray);
5. document.writeln(array);
6. **</script>**

**Output:**

1,2,3,4,5,6,7,8

### **Example 4**

Let's see an example to join array of different type.

1. **<script>**
2. var array = [1,2,3,4];
3. var newarray=["One","Two","Three","Four"]
4. array.push.apply(array, newarray);
5. document.writeln(array);
6. **</script>**

**Output:**

1,2,3,4,One,Two,Three,Four

# JavaScript Function call() method

The JavaScript Function **call()** method is used to call a function contains this value and an argument provided individually. Unlike apply() method, it accepts the argument list.

## Syntax

1. function.call(thisArg, arg1,arg2,....,argn)

## Parameter

**thisArg** - It is optional. The this value is given for the call to function.

**arg1,arg2,...,argn** - It is optional. It represents the arguments for the function.

## Return Value

It returns the result of the calling function along provided this value and arguments.

Play Video

## JavaScript Function call() method Example

### **Example 1**

Let's see a simple example of call() method.

1. **<script>**
2. function Emp(id,name) {
3. this.id = id;
4. this.name = name;
5. }
6. function PermanentEmp(id,name) {
7. Emp.call(this,id,name);
8. }
9. document.writeln(new PermanentEmp(101,"John Martin").name);
10. **</script>**

**Output:**

John Martin

### **Example 2**

Let's see an example of call() method.

1. **<script>**
2. function Emp(id,name) {
3. this.id = id;
4. this.name = name;
5. }
6. function PermanentEmp(id,name) {
7. Emp.call(this,id,name);
8. }
9. function TemporaryEmp(id,name) {
10. Emp.call(this,id,name);
11. }
12. var p\_emp=new PermanentEmp(101,"John Martin");
13. var t\_emp=new TemporaryEmp(201,"Duke William")
14. document.writeln(p\_emp.id+" "+p\_emp.name+"**<br>**");
15. document.writeln(t\_emp.id+" "+t\_emp.name);**</script>**

**Output:**

101 John Martin

201 Duke William