JavaScript Number Methods

The toString() method converts a number to a string.

## Converting Variables to Numbers

There are 3 JavaScript methods that can be used to convert variables to numbers:

* The Number() method
* The parseInt() method
* The parseFloat() method

# Arrays in JavaScript

In JavaScript, array is a single variable that is used to store different elements. It is often used when we want to store list of elements and access them by a single variable. Unlike most languages where array is a reference to the multiple variable, in JavaScript array is a single variable that stores multiple elements.

**There are 3 ways to construct array in JavaScript**

1. **By array literal**
2. **By creating instance of Array directly (using new keyword)**
3. **By using an Array constructor (using new keyword)**

## 1) JavaScript array literal

The syntax of creating array using array literal is given below:

var arrayname=[value1,value2.....valueN];

var fruits = [ "apple", "orange", "mango" ];

## 2) JavaScript Array directly (new keyword)

The syntax of creating array directly is given below:

var arrayname=new Array();

**<script>**

var i;

var emp = new Array();

emp[0] = "Arun";

emp[1] = "Deepak";

emp[2] = "Vijay";

for (i=0;i**<emp.length**;i++){

document.write(emp[i] + "**<br>**");

}

**</script>**

var fruits = new Array(5);0

**<!DOCTYPE html>**

**<html>**

**<body>**

**<h2>JavaScript Arrays</h2>**

**<script>**

**var num = new Array(5);**

**var i,ln;**

**for(i=0;i<num.length;i++)**

**{**

**num[i]=i+1;**

**}**

**for(i=0;i<num.length;i++)**

**{**

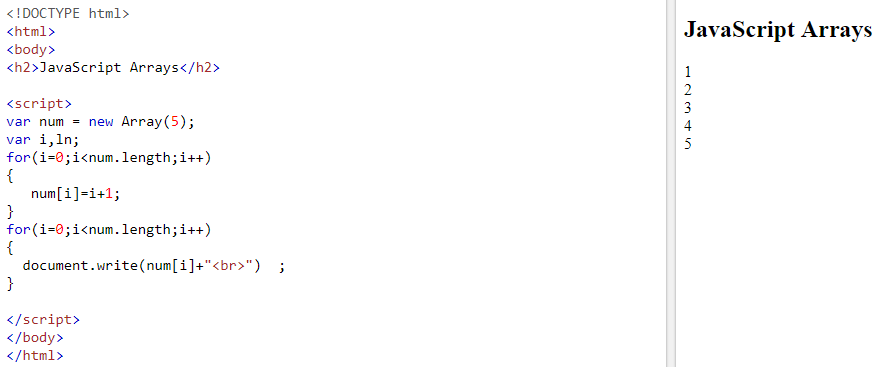
**document.write(num[i]+"<br>") ;**

**}**

**</script>**

**</body>**

**</html>**

****

## 3) JavaScript array constructor (new keyword)

**<script>**

var emp=new Array("Sandeep","Hardeep","Mandeep");

for (i=0;i**<emp.length**;i++){

document.write(emp[i] + "**<br>**");

}

**</script>**

## Array Properties and Methods

The real strength of JavaScript arrays are the built-in array properties and methods:

## 1.The length Property

The length property of an array returns the length of an array (the number of array elements).

### Example

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits.length;   // the length of fruits is 4

2.JavaScript Array reverse() method

**<!DOCTYPE html>**

**<html>**

**<body>**

**<script>**

**var arr=["sandeep","mandeep","hardeep"];**

**var rev=arr.reverse();**

**var i;**

**for(i=0;i<rev.length;i++)**

**{**

**document.write(rev[i]+"<br>");**

**}**

**</script>**

**</body>**

**</html>**

****

## 3.Sorting an Array

The sort() method sorts an array alphabetically:

**<!DOCTYPE html>**

**<html>**

**<body>**

**<script>**

**var arr=["san","deep","amit","geu"];**

**var myarry=arr.sort();**

**var i;**

**for(i=0;i<myarry.length;i++)**

**{**

**document.write(myarry[i]+"<br>");**

**}**

**</script>**

**</body>**

**</html>**

## 4.Merging (Concatenating) Arrays

The concat() method creates a new array by merging (concatenating) existing arrays:

**<!DOCTYPE html>**

**<html>**

**<body>**

**<script>**

**var arr1=["C","C++","Python"];**

**var arr2=["Java","JavaScript","Android"];**

**var arr3=arr1.concat(arr2);**

**var i;**

**for(i=0;i<arr3.length;i++)**

**{**

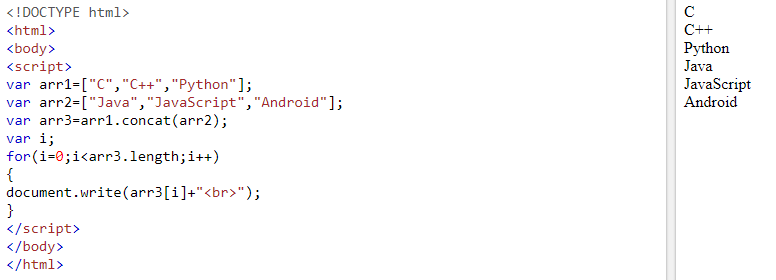
**document.write(arr3[i]+"<br>");**

**}**

**</script>**

**</body>**

**</html>**

****

## Popping

The pop() method removes the last element from an array:

The pop() method returns the value that was "popped out":

### Example

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits.pop(); // Removes the last element ("Mango") from fruits

**<!DOCTYPE html>**

**<html>**

**<body>**

**<script>**

**var arr=["Java","JavaScript","Android"];**

**var i;**

**for(i=0;i<arr.length;i++)**

**{**

**document.write(arr[i]+"<br>");**

**}**

**arr.pop();**

**document.write("<h2>New Values are</h2>");**

**for(i=0;i<arr.length;i++)**

**{**

**document.write(arr[i]+"<br>");**

**}**

**</script>**

**</body>**

**</html>**

****

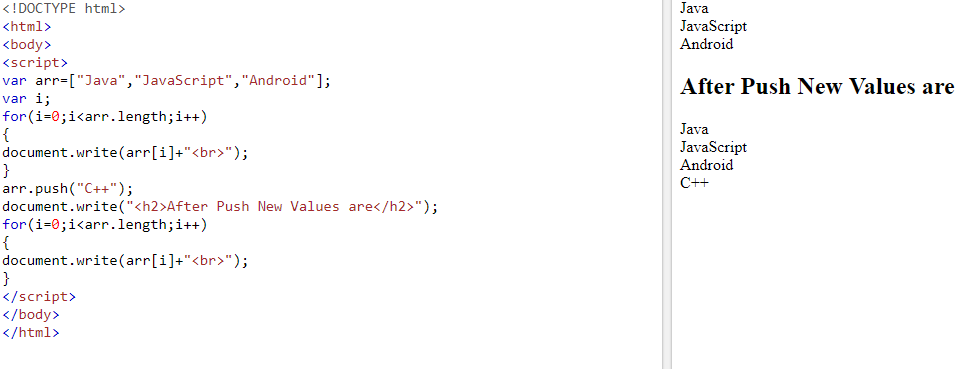
## Pushing

The push() method adds a new element to an array (at the end):

### Example

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits.push("Kiwi");       //  Adds a new element ("Kiwi") to fruits

The push() method returns the new array length:

****

**<!DOCTYPE html>**

**<html>**

**<body>**

**<p>The push() method returns the new array length.</p>**

**<button onclick="myFunction()">Try it</button>**

**<p id="demo1"></p>**

**<p id="demo2"></p>**

**<script>**

**var fruits = ["Banana", "Orange", "Apple", "Mango"];**

**document.getElementById("demo1").innerHTML = fruits;**

**function myFunction() {**

**document.getElementById("demo2").innerHTML = fruits.push("Kiwi");**

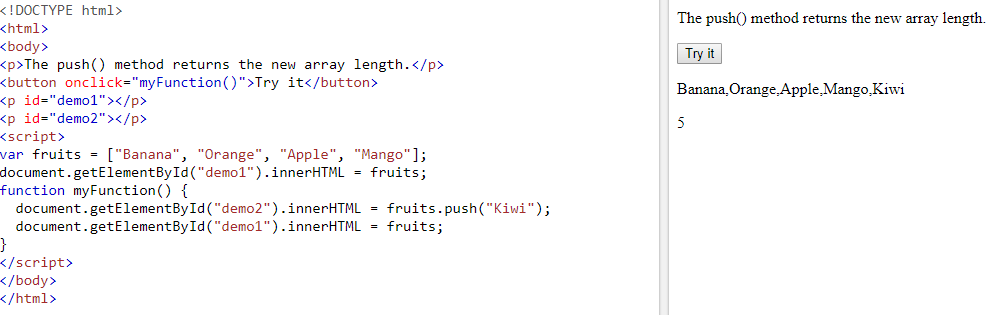
**document.getElementById("demo1").innerHTML = fruits;**

**}**

**</script>**

**</body>**

**</html>**

****

## Shifting Elements

Shifting is equivalent to popping, working on the first element instead of the last.

The shift() method removes the first array element and "shifts" all other elements to a lower index.

The shift() method returns the string that was "shifted out":

**<!DOCTYPE html>**

**<html>**

**<body>**

**<p id="demo1"></p>**

**<p id="demo2"></p>**

**<script>**

**var fruits = ["Banana", "Orange", "Apple", "Mango"];**

**document.getElementById("demo1").innerHTML = fruits;**

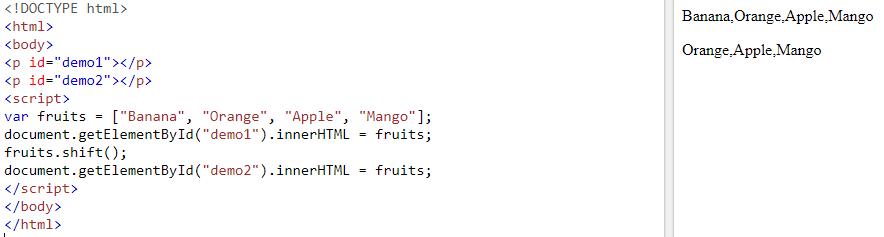
**fruits.shift();**

**document.getElementById("demo2").innerHTML = fruits;**

**</script>**

**</body>**

**</html>**

****

The unshift() method adds a new element to an array (at the beginning), and "unshifts" older elements:

**<!DOCTYPE html>**

**<html>**

**<body>**

**<button onclick="myFunction()">Try it</button>**

**<p id="demo"></p>**

**<hr>**

**<p id="demo2"></p>**

**<script>**

**var fruits = ["Banana", "Orange", "Apple", "Mango"];**

**document.getElementById("demo").innerHTML = fruits;**

**function myFunction() {**

**fruits.unshift("Lemon");**

**document.getElementById("demo2").innerHTML = fruits;**

**}**

**</script>**

**</body>**

**</html>**

****

## Splicing an Array

The splice() method can be used to add new items to an array:

The first parameter (2) defines the position **where** new elements should be **added** (spliced in).

The second parameter (0) defines **how many** elements should be **removed**.

The rest of the parameters ("Lemon" , "Kiwi") define the new elements to be **added**.

The splice() method returns an array with the deleted items:

**<!DOCTYPE html>**

**<html>**

**<body>**

**<p>The splice() method adds new elements to an array.</p>**

**<button onclick="myFunction()">Try it</button>**

**<p id="demo1"></p>**

**<p id="demo2"></p>**

**<script>**

**var fruits = ["Banana", "Orange", "Apple", "Mango"];**

**document.getElementById("demo1").innerHTML = "Original Array:<br>" + fruits;**

**function myFunction() {**

**fruits.splice(2, 0, "Lemon", "Kiwi");**

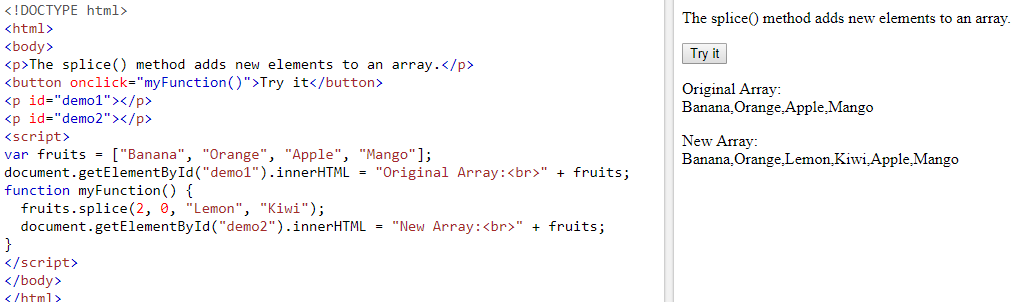
**document.getElementById("demo2").innerHTML = "New Array:<br>" + fruits;**

**}**

**</script>**

**</body>**

**</html>**

****

## Using splice() to Remove Elements

With clever parameter setting, you can use splice() to remove elements without leaving "holes" in the array:

### Example

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits.splice(0, 1);        // Removes the first element of fruits

The first parameter (0) defines the position where new elements should be **added** (spliced in).

The second parameter (1) defines **how many** elements should be **removed**.

The rest of the parameters are omitted. No new elements will be added.

**<!DOCTYPE html>**

**<html>**

**<body>**

**<p>The splice() methods can be used to remove array elements.</p>**

**<button onclick="myFunction()">Try it</button>**

**<p id="demo"></p>**

**<script>**

**var fruits = ["Banana", "Orange", "Apple", "Mango"];**

**document.getElementById("demo").innerHTML = fruits;**

**function myFunction() {**

**fruits.splice(1, 1);**

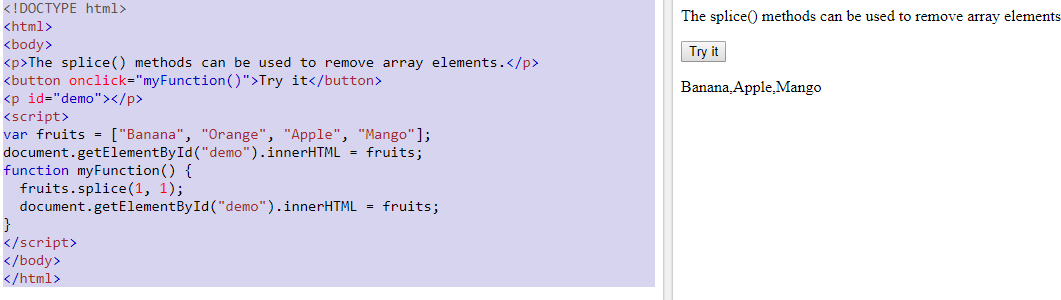
**document.getElementById("demo").innerHTML = fruits;**

**}**

**</script>**

**</body>**

**</html>**

****

## Slicing an Array

The slice() method slices out a piece of an array into a new array.

**<!DOCTYPE html>**

**<html>**

**<body>**

**<p id="demo"></p>**

**<script>**

**var fruits = ["Banana", "Orange", "Lemon", "Apple", "Mango"];**

**var citrus = fruits.slice(3);**

**document.getElementById("demo").innerHTML = fruits + "<br><br>" + citrus;**

**</script>**

**</body>**

**</html>**

****

**This example slices out a part of an array starting from array element 3 ("Apple"):**

**When the slice() method is given two arguments, it selects array elements from the start argument, and up to (but not included) the end argument:**

**<!DOCTYPE html>**

**<html>**

**<body>**

**<p id="demo"></p>**

**<script>**

**var fruits = ["Banana", "Orange", "Lemon", "Apple", "Mango"];**

**var citrus = fruits.slice(1,3);**

**document.getElementById("demo").innerHTML = fruits + "<br><br>" + citrus;**

**</script>**

**</body>**

**</html>**

****

JavaScript Array indexOf() method

It searches the specified element in the given array and returns the index of the first match.

**This method is case-sensitive.**

**The index position of first element in an array is always start with zero. If an element is not present in an array, it returns -1.**

**The indexOf() method is represented by the following syntax:**

**array.indexOf(element,index)**

<script>

var arr=["C","C++","Python","C++","Java"];

var result= arr.indexOf("C++");

document.writeln(result);

</script>

### Example 2

In this example, we will provide the index value from where the search starts.

<script>

var arr=["C","C++","Python","C++","Java"];

var result= arr.indexOf("C++",2);

document.writeln(result);

</script>

JavaScript Array lastIndexOf() method

It searches the specified element in the given array and returns the index of the last match.

 If an element is not present in an array, it returns -1.

<script>

var arr=["C","C++","Python","C++","Java"];

var result= arr.lastIndexOf("C++");

document.writeln(result);

</script>

JavaScript Date Objects

<!DOCTYPE html>

<html>

<body>

<h2>JavaScript new Date()</h2>

<p id="demo"></p>

<script>

var d = new Date();

document.getElementById("demo").innerHTML = d;

</script>

</body>

</html>

OutPut

## JavaScript new Date()

Mon Feb 25 2019 12:17:35 GMT+0530 (India Standard Time)

## Creating Date Objects

Date objects are created with the new Date() constructor.

There are **4 ways** to create a new date object:

new Date()  
new Date(year, month, day, hours, minutes, seconds, milliseconds)  
new Date(milliseconds)  
new Date(date string)

## The getFullYear() Method

The getFullYear() method returns the year of a date as a four digit number:

## The getMonth() Method

The getMonth() method returns the month of a date as a number (0-11):

<p id="demo"></p>

<script>

var d = new Date();

document.getElementById("demo").innerHTML = d.getMonth() + 1;

</script>

In JavaScript, the first month (January) is month number 0, so December returns month number 11.

You can use an array of names, and getMonth() to return the month as a name:

<p id="demo"></p>

<script>

var d = new Date();

var months = ["January","February","March","April","May","June","July","August","September","October","November","December"];

document.getElementById("demo").innerHTML = months[d.getMonth()];

</script>

## The getDate() Method

The getDate() method returns the day of a date as a number (1-31):

<p id="demo"></p>

<script>

var d = new Date();

document.getElementById("demo").innerHTML = d.getDate();

</script>

## The getHours() Method

The getHours() method returns the hours of a date as a number (0-23):

## The getMinutes() Method

The getMinutes() method returns the minutes of a date as a number (0-59):

## The getSeconds() Method

The getSeconds() method returns the seconds of a date as a number (0-59):

## The getMilliseconds() Method

The getMilliseconds() method returns the milliseconds of a date as a number (0-999):

## The getDay() Method

The getDay() method returns the weekday of a date as a number (0-6):

In JavaScript, the first day of the week (0) means "Sunday", even if some countries in the world consider the first day of the week to be "Monday"

You can use an array of names, and getDay() to return the weekday as a name:

<p id="demo"></p>

<script>

var d = new Date();

var days = ["Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday"];

document.getElementById("demo").innerHTML = days[d.getDay()];

</script>