JavaScript Arrays

- **What is an Array?**
- An array is a special variable, which can hold more than one value at a time.

- **Creating an Array**

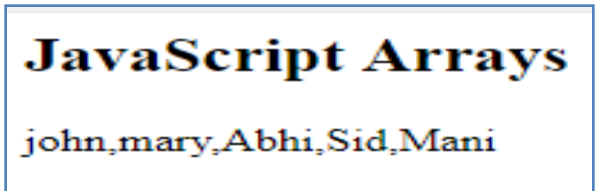
Syntax:

var *array_name* = [*item1*, *item2*, ...];

Example: arrays

```
<html>
<body>

<h2>JavaScript Arrays</h2>

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

<script>
var names = ["john", "mary", "Abhi", "Sid", "Mani"];
document.getElementById("demo").innerHTML = names;
</script>
</body>
</html>
```

Output:



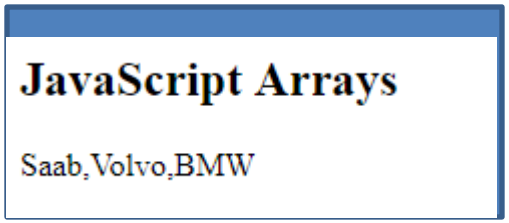
Other method to create arrays

- Using the JavaScript Keyword new
- The following example also creates an Array, and assigns values to it:
- Example

```
var cars = new Array("Saab", "Volvo", "BMW");
```

Example: using “new”

```
<html>
<body>

<h2>JavaScript Arrays</h2>

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

<script>
var cars = new Array("Saab", "Volvo", "BMW");

document.getElementById("demo").innerHTML = cars;
</script>

</body>
</html>
```



Access the Elements of an Array

- You refer to an array element by referring to the **index number**.
- This statement accesses the value of the first element in cars:

```
var name = cars[0];
```

Array Properties and Methods

Length property

```
<html>
<body>


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

<script>
var names = ["john", "mary", "Abhi", "Sid", "Mani"];

document.getElementById("demo").innerHTML = names.length;


</script>

</body>
</html>
```

Output: 5

Sorting an Array

The **sort()** method sorts an array alphabetically:

```
<html>
<body>


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

<script>
var names = ["John", "Cary", "Abhi", "Sid", "Mani"];

document.getElementById("demo").innerHTML = names.sort();


</script> </body> </html>
```

Output:
Abhi,Cary,John,Mani,Sid

Reversing an Array
reverse() method reverses the elements in an array

```
<script>
var names = ["John", "Cary", "Abhi", "Sid", "Mani"];

document.getElementById("demo").innerHTML = names.reverse();

</script>
```

Output:

Mani,Sid,Abhi,Cary,John

Pop() removes the last element from an array

```
<!DOCTYPE html>
<html>
<body>

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

<script>
var names = ["john", "mary", "Abhi", "Sid", "Mani"];
document.getElementById("demo").innerHTML = names;

names.pop();
document.getElementById("demo2").innerHTML = names;


</script></body></html>
```

Output:
john,mary,Abhi,Sid,Mani

john,mary,Abhi,Sid

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

```
<html>
<body>
<p id="demo"></p>

<script>
var names = ["john", "mary", "Abhi", "Sid", "Mani"];
names.push("jim");

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

</script></body></html>
```

Output:

john,mary,Abhi,Sid,Mani,jim

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

```
<html>
<body>
<p id="demo"></p>

<script>
var names = ["john", "mary", "Abhi", "Sid", "Mani"];
names.shift();

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

</script></body></html>
```

Output:
mary,Abhi,Sid,Mani

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

- **To add a new name “mac”**

```
<html>
<body>
<p id="demo"></p>

<script>
var names = ["john", "mary", "Abhi", "Sid", "Mani"];
names.unshift("mac");

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

</script></body></html>
```

Output:

mac,john,mary,Abhi,Sid,Mani

Changing Elements

- Array elements are accessed using their **index number**:
- Array **indexes** start with 0. [0] is the first array element, [1] is the second, [2] is the third ...

example

```
<html>
<body>
<p id="demo2"></p>

<script>
var fruits = ["Banana", "Orange", "Apple", "Mango"];

fruits[0] = "Kiwi";
document.getElementById("demo2").innerHTML = fruits;
</script>

</body>
</html>
```

Output:

Kiwi,Orange,Apple,Mango

Delete:

elements can be deleted by using **delete**:

```
<html>
<body>
<p id="demo"></p>

<script>
var names = ["john", "mary", "Abhi", "Sid", "Mani"];
delete names[1];

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

</script></body></html>
```

Output:
john,,Abhi,Sid,Mani

Splicing an Array
to add new items to an array

Has 2 parameters:

- The first parameter defines the position **where** new elements should be **added** (spliced in).
- The second parameter defines **how many** elements should be **removed**.

Example splice

```
<!DOCTYPE html>
<html><body>

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

<script>
var names = ["john", "mary", "Abhi", "Sid", "Mani"];
```

names.splice(2,0,"ABC", "DEF");

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

```
</script></body></html>
```

Output:
john,mary,ABC,DEF,Abhi,Sid,Mani

- Using splice() to Remove Elements
- E.g. to remove 1st element of names array:

```
var names = ["john", "mary", "Abhi", "Sid", "Mani"];
names.splice(0,1);
```

Output:

mary,Abhi,Sid,Mani

Merging (Concatenating) Arrays

```
<html>
<body>

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

<script>
var a = ["pizza", "burger"];
var b = ["Maggi", "Sub"];
```

var c = a.concat(b);

```
document.getElementById("demo").innerHTML = c;
</script>

</body>
</html>
```

Output:
pizza,burger,Maggi,Sub

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

- Example: slices out a part of an array starting from array element 2

```
<html>
<body>
<p id="demo"></p>

<script>
var names = ["john", "mary", "Abhi", "Sid", "Mani"];

var names_new=names.slice(2);
document.getElementById("demo").innerHTML = names_new;

</script></body></html>
```

Output:
Abhi,Sid,Mani

The slice() method can take two arguments

The method then selects elements from the start argument, and up to (but not including) the end argument.

```
<html>
<body>
<p id="demo"></p>

<script>
var names = ["john", "mary", "Abhi", "Sid", "Mani"];

var names_new=names.slice(1,3);
document.getElementById("demo").innerHTML = names_new;

</script></body></html>
```

Output:

mary,Abhi

JavaScript Array join() Method

- Join the elements of an array into a string
- The elements will be separated by a specified separator.
- The default separator is comma (,).

Using any other symbol as separator

e.g. use "&" as separator

```
<html>
<body>

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

<script>
   var fruits = ["Banana", "Orange", "Apple", "Mango"];
   var x = document.getElementById("demo");
   x.innerHTML = fruits.join("&");

</script>

</body>
</html>
```

Output:

Banana&Orange&Apple&Mango

Converting Arrays to Strings

- The JavaScript method **toString()** converts an array to a string of (comma separated) array values.

Example

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

document.getElementById("demo").innerHTML = fruits.toString();
```

Result

Banana,Orange,Apple,Mango