JavaScript Array

- JavaScript array is an object that represents a collection of similar type of 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];

- values are contained inside [ ] and separated by , (comma).

Let's see the simple example of creating and using array in JavaScript.

```
<script>

var emp=["Sonoo","Vimal","Ratan"];

for (i=0;i<emp.length;i++){

document.write(emp[i] + "<br/>");

}

</script>
```

The .length property returns the length of an array.

Output of the above example

```
Sonoo
Vimal
Ratan
```

2) JavaScript Array directly (new keyword)

The syntax of creating array directly is given below:

```
var arrayname=new Array();
```

Here, new **keyword is used to create instance of array.**

Let’s see the example of creating array directly.

```
<script>

var i;

var emp = new Array();

emp[0] = "Arun";

emp[1] = "Varun";

emp[2] = "John";


for (i=0;i<emp.length;i++){

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

}

</script>
```

Output of the above example

```
Arun
Varun
John
```

3) JavaScript array constructor (new keyword)

Here, you need to create instance of array by passing arguments in constructor so that we don't have to provide value explicitly.

The example of creating object by array constructor is given below.

```
<script>

var emp=new Array("Jai","Vijay","Smith");

for (i=0;i<emp.length;i++){

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

}

</script>
```

Output of the above example

```
Jai
Vijay
Smith
```

JavaScript Popup Boxes

- JavaScript has three kinds of popup boxes: Alert box, Confirm box, and Prompt box.

Alert Box

An alert box is often used if you want to make sure information comes through to the user.

When an alert box pops up, the user will have to click "OK" to proceed.

Example
alert("I am an alert box!");

code

```
<html>

<body>

<p>Click the button to display an alert box:</p>

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

<script>

function myFunction() {

   alert("I am an alert box!");

}

</script>

</body>

</html>
```

Confirm Box

- A confirm box is often used if you want the user to verify or accept something.
- When a confirm box pops up, the user will have to click either "OK" or "Cancel" to proceed.
- If the user clicks "OK", the box returns true. If the user clicks "Cancel", the box returns false.

Example

```
<html>

<body>

<p>Click the button to display a confirm box.</p>

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

<script>

function myFunction() {

   var x;

   if (confirm("Press a button!") == true) {

      x = "You pressed OK!";

   } else {

      x = "You pressed Cancel!";

   }

   document.write( x);
}

</script>


</body>

</html>
}
```

Prompt Box

A prompt box is often used if you want the user to input a value before entering a page.

When a prompt box pops up, the user will have to click either "OK" or "Cancel" to proceed after entering an input value.

If the user clicks "OK" the box returns the input value. If the user clicks "Cancel" the box returns null.

Example

```
<html>

<body>

<p>Click the button to demonstrate the prompt box.</p>

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

<script>

function myFunction() {

    var person = prompt("Please enter your name");
```

```
        if (person != null) {
        document.write("Hello " + person + "! How are you today?");
    }
}
</script>
</body>
</html>
```

Line Breaks

To display line breaks inside a popup box, use a back-slash followed by the character n.

Example

```
alert("Hello\nHow are you?");
```

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.

- **There** are two types of functions

  1. Built-in functions
  2. User defined functions

Built-in functions

- JavaScript has five functions built in to the language. They are eval, parseInt, parseFloat, number, and string.

1. eval

- Evaluates a string and returns a value.
- It converts strings to a numeric value.

eval(Expression)

Examples

```
var num = 2;
 eval("num + 200");
```

2. parseInt

Parses a string argument and returns an integer and convert number into integer.

Syntax:

parseInt(string)

string is a string that represents the value you want to parse.

```
<html>

<body>

<p>Click the button to parse different strings.</p>

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

<script>

function myFunction() {

    var a = parseInt("10") + "<br>";

    var b = parseInt("10.00") + "<br>";

    var c = parseInt("10.33") + "<br>";

    var d = parseInt("34 45 66") + "<br>";

    var e = parseInt("   60   ") + "<br>";

    var f = parseInt("40 years") + "<br>";

    var g = parseInt("He was 40") + "<br>";

    var n = a + b + c + d + e + f + g ;
    document.write( n);
}
</script>
</body>
</html>
```

3. **parseFloat**

Parses a string argument and returns a floating point number.

If string is not begin with a floating point than it returns NaN error.

Syntax:

```
        parseFloat(string)
```

```
<html>
<body>
<p>Click the button to parse different strings.</p>
<button onclick="myFunction()">Try it</button>
<script>
function myFunction() {
    var a = parseFloat("10") + "<br>";
    var b = parseFloat("10.00") + "<br>";
    var c = parseFloat("10.33") + "<br>";
    var d = parseFloat("34 45 66") + "<br>";
    var e = parseFloat("   60   ") + "<br>";
    var f = parseFloat("40 years") + "<br>";
    var g = parseFloat("He was 40") + "<br>";

    var n = a + b + c + d + e + f + g;
    document.write(n);
}
</script>
</body>
</html>
```

4. **Number**

   It converts corresponding value to number or give error NaN

Example

```
Var obj1=new string("123");
Var obj1=new boolean("false");
Var obj1=new boolean("true");

Document.write(number(obj1));
Document.write(number(obj2));
Document.write(number(obj3));
```

Output

```
123

0

1
```

5. **String**

   It converts corresponding value to string.

Example

```
Var obj1=new boolean(0);

Var obj1=new boolean(1);


Document.write(number(obj1));

Document.write(number(obj2));
```

Output

```
False

true
```

User defined functions

JavaScript Function Syntax

The syntax of declaring function is given below.

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

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.

```
<script>
function msg(){
alert("hello! this is message");
}
</script>
<input type="button" onclick="msg()" value="call function"/>
```

Function Arguments

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

```
<script>
function getcube(number){
```

```
alert(number*number*number);

}

</script>

<form>

<input type="button" value="click" onclick="getcube(4)"/>

</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.

```
<script>

function getInfo(){

return "hello javatpoint! How r u?";

}

</script>

<script>

document.write(getInfo());

</script>
```

JavaScript Events

HTML events are **"things"** that happen to HTML elements.

When JavaScript is used in HTML pages, JavaScript can **"react"** on these events.

JavaScript's interaction with HTML is handled through events that occur when the user or the browser manipulates a page.

When the page loads, it is called an event. When the user clicks a button, that click too is an event. Other examples include events like pressing any key, closing a window, resizing a window, etc.

Events	**Description**
onclick	occurs when element is clicked.
ondblclick	occurs when element is double-clicked.
onfocus	occurs when an element gets focus such as button, input, textarea etc.
onblur	occurs when form looses the focus from an element.
onsubmit	occurs when form is submitted.
onmouseover	occurs when mouse is moved over an element.
onmouseout	occurs when mouse is moved out from an element (after moved over).
onmousedown	occurs when mouse button is pressed over an element.
onmouseup	occurs when mouse is released from an element (after mouse is pressed).
onload	occurs when document, object or

	frameset is loaded.
onunload	occurs when body or frameset is unloaded.
onscroll	occurs when document is scrolled.
onresized	occurs when document is resized.
onreset	occurs when form is reset.
onkeydown	occurs when key is being pressed.
onkeypress	occurs when user presses the key.
onkeyup	occurs when key is released.