* **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];

**ckkjs020.js**

<html>

<body>

<script>

var emp=["Miral","Vimal","Misri"];

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

{

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

}

</script>

</body>

</html>

The .length property returns the length of an array.

(2) JavaScript Array directly (new keyword)

The syntax of creating array directly is given below:

var arrayname=new Array();

**ckkjs021.js**

<!-- JavaScript array with new keyword -->

<html>

<body>

<script>

var emp=new Array();

emp[0]="Mita";

emp[1]="Misri";

emp[2]="Gunjan";

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

{

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

}

</script>

</body>

</html>

(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.

**ckkjs022.js**

<!-- JavaScript array with constructor -->

<html>

<body>

<script>

var emp=new Array("Mita","Misri","Gunjan");

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

{

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

}

</script>

</body>

</html>

* **JavaScript String object**

The **JavaScript string** is an object that represents a sequence of characters.

There are 2 ways to create string in JavaScript

1. By string literal
2. By string object (using new keyword)

(1) By string literal

The string literal is created using double quotes. The syntax of creating string using string literal is given below:

var stringname="string value";

**ckkjs023.js**

var stringname="string value";

<!-- string literal -->

<html>

<body>

<script>

var str="God is great, it is checking of string literal";

document.write(str);

</script>

</body>

</html>

(2) By string object (using new keyword)

The syntax of creating string object using new keyword is given below:

var stringname=new String("string literal");

var stringname=new String("string literal");

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

**ckkjs024.js**

<!-- string constructor -->

<html>

<body>

<script>

var str=new String("God is great, it is checking of string literal");

document.write(str);

</script>

</body>

</html>

**JavaScript String object properties**

|  |  |
| --- | --- |
| **Property** | **Description** |
| [constructor](http://www.w3schools.com/jsref/jsref_constructor_string.asp) | Returns the string's constructor function |
| [length](http://www.w3schools.com/jsref/jsref_length_string.asp) | Returns the length of a string |
| [prototype](http://www.w3schools.com/jsref/jsref_prototype_string.asp) | Allows you to add properties and methods to an object |

**JavaScript String methods**

|  |  |
| --- | --- |
| **Method** | **Description** |
| [charAt()](http://www.w3schools.com/jsref/jsref_charat.asp) | Returns the character at the specified index (position) |
| [charCodeAt()](http://www.w3schools.com/jsref/jsref_charcodeat.asp) | Returns the Unicode of the character at the specified index |
| [concat()](http://www.w3schools.com/jsref/jsref_concat_string.asp) | Joins two or more strings, and returns a new joined strings |
| [endsWith()](http://www.w3schools.com/jsref/jsref_endswith.asp) | Checks whether a string ends with specified string/characters |
| [fromCharCode()](http://www.w3schools.com/jsref/jsref_fromcharcode.asp) | Converts Unicode values to characters |
| [includes()](http://www.w3schools.com/jsref/jsref_includes.asp) | Checks whether a string contains the specified string/characters |
| [indexOf()](http://www.w3schools.com/jsref/jsref_indexof.asp) | Returns the position of the first found occurrence of a specified value in a string |
| [lastIndexOf()](http://www.w3schools.com/jsref/jsref_lastindexof.asp) | Returns the position of the last found occurrence of a specified value in a string |
| [localeCompare()](http://www.w3schools.com/jsref/jsref_localecompare.asp) | Compares two strings in the current locale |
| [match()](http://www.w3schools.com/jsref/jsref_match.asp) | Searches a string for a match against a regular expression, and returns the matches |
| [repeat()](http://www.w3schools.com/jsref/jsref_repeat.asp) | Returns a new string with a specified number of copies of an existing string |
| [replace()](http://www.w3schools.com/jsref/jsref_replace.asp) | Searches a string for a specified value, or a regular expression, and returns a new string where the specified values are replaced |
| [search()](http://www.w3schools.com/jsref/jsref_search.asp) | Searches a string for a specified value, or regular expression, and returns the position of the match |
| [slice()](http://www.w3schools.com/jsref/jsref_slice_string.asp) | Extracts a part of a string and returns a new string |
| [split()](http://www.w3schools.com/jsref/jsref_split.asp) | Splits a string into an array of substrings |
| [startsWith()](http://www.w3schools.com/jsref/jsref_startswith.asp) | Checks whether a string begins with specified characters |
| [substr()](http://www.w3schools.com/jsref/jsref_substr.asp) | Extracts the characters from a string, beginning at a specified start position, and through the specified number of character |
| [substring()](http://www.w3schools.com/jsref/jsref_substring.asp) | Extracts the characters from a string, between two specified indices |
| [toLocaleLowerCase()](http://www.w3schools.com/jsref/jsref_tolocalelowercase.asp) | Converts a string to lowercase letters, according to the host's locale |
| [toLocaleUpperCase()](http://www.w3schools.com/jsref/jsref_tolocaleuppercase.asp) | Converts a string to uppercase letters, according to the host's locale |
| [toLowerCase()](http://www.w3schools.com/jsref/jsref_tolowercase.asp) | Converts a string to lowercase letters |
| [toString()](http://www.w3schools.com/jsref/jsref_tostring_string.asp) | Returns the value of a String object |
| [toUpperCase()](http://www.w3schools.com/jsref/jsref_touppercase.asp) | Converts a string to uppercase letters |
| [trim()](http://www.w3schools.com/jsref/jsref_trim_string.asp) | Removes whitespace from both ends of a string |
| [valueOf()](http://www.w3schools.com/jsref/jsref_valueof_string.asp) | Returns the primitive value of a String object |

String HTML Wrapper Methods

The HTML wrapper methods return the string wrapped inside the appropriate HTML tag.

These are not standard methods, and may not work as expected in all browsers.

|  |  |
| --- | --- |
| **Method** | **Description** |
| [anchor()](http://www.w3schools.com/jsref/jsref_anchor.asp) | Creates an anchor |
| [big()](http://www.w3schools.com/jsref/jsref_big.asp) | Displays a string using a big font |
| [blink()](http://www.w3schools.com/jsref/jsref_blink.asp) | Displays a blinking string |
| [bold()](http://www.w3schools.com/jsref/jsref_bold.asp) | Displays a string in bold |
| [fixed()](http://www.w3schools.com/jsref/jsref_fixed.asp) | Displays a string using a fixed-pitch font |
| [fontcolor()](http://www.w3schools.com/jsref/jsref_fontcolor.asp) | Displays a string using a specified color |
| [fontsize()](http://www.w3schools.com/jsref/jsref_fontsize.asp) | Displays a string using a specified size |
| [italics()](http://www.w3schools.com/jsref/jsref_italics.asp) | Displays a string in italic |
| [link()](http://www.w3schools.com/jsref/jsref_link.asp) | Displays a string as a hyperlink |
| [small()](http://www.w3schools.com/jsref/jsref_small.asp) | Displays a string using a small font |
| [strike()](http://www.w3schools.com/jsref/jsref_strike.asp) | Displays a string with a strikethrough |
| [sub()](http://www.w3schools.com/jsref/jsref_sub.asp) | Displays a string as subscript text |
| [sup()](http://www.w3schools.com/jsref/jsref_sup.asp) | Displays a string as superscript text |

(1) JavaScript String charAt(index) Method

The JavaScript String charAt() method returns the character at the given index.

**ckkjs025.js**

<!-- string fuctions -->

<html>

<body>

<script>

var str="God is great";

document.write(str.charAt(2));

</script>

</body>

</html>

(2) JavaScript String concat(str) Method

The JavaScript String concat(str) method concatenates or joins two strings.

**ckkjs026.js**

<!-- string fuctions -->

<html>

<body>

<script>

var s1="God is ";

var s2="great";

var s3=s1.concat(s2);

document.write(s3);

</script>

</body>

</html>

(3) JavaScript String indexOf(str) Method

The JavaScript String indexOf(str) method returns the index position of the given string.

**ckkjs027.js**

<!-- string fuctions indexof-->

<html>

<body>

<script>

var s1="God is great, beause it is great";

var n=s1.indexOf("great");

document.write(n);

</script>

</body>

</html>

Output:

7

(4) JavaScript String lastIndexOf(str) Method

The JavaScript String lastIndexOf(str) method returns the last index position of the given string

.

**<script>**

var s1="javascript from javatpoint indexof";

var n=s1.lastIndexOf("java");

document.write(n);

**</script>**

<script>

var s1="javascript from javatpoint indexof";

var n=s1.lastIndexOf("java");

document.write(n);

</script>

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=jsstringmethod4)

Output:

16

(5) JavaScript String toLowerCase() Method

The JavaScript String toLowerCase() method returns the given string in lowercase letters.

**<script>**

var s1="JavaScript toLowerCase Example";

var s2=s1.toLowerCase();

document.write(s2);

**</script>**

<script>

var s1="JavaScript toLowerCase Example";

var s2=s1.toLowerCase();

document.write(s2);

</script>

Output:

javascript tolowercase example

(6) JavaScript String toUpperCase() Method

The JavaScript String toUpperCase() method returns the given string in uppercase letters.

**<script>**

var s1="JavaScript toUpperCase Example";

var s2=s1.toUpperCase();

document.write(s2);

**</script>**

<script>

var s1="JavaScript toUpperCase Example";

var s2=s1.toUpperCase();

document.write(s2);

</script>

Output:

JAVASCRIPT TOUPPERCASE EXAMPLE

(7) JavaScript String slice(beginIndex, endIndex) Method

The JavaScript String slice(beginIndex, endIndex) method returns the parts of string from given beginIndex to endIndex. In slice() method, beginIndex is inclusive and endIndex is exclusive.

**<script>**

var s1="abcdefgh";

var s2=s1.slice(2,5);

document.write(s2);

**</script>**

<script>

var s1="abcdefgh";

var s2=s1.slice(2,5);

document.write(s2);

</script>

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=jsstringmethod7)

Output:

cde

(8) JavaScript String trim() Method

The JavaScript String trim() method removes leading and trailing whitespaces from the string.

**<script>**

var s1="     javascript trim    ";

var s2=s1.trim();

document.write(s2);

**</script>**

<script>

var s1=" javascript trim ";

var s2=s1.trim();

document.write(s2);

</script>

Output:

javascript trim

# JavaScript Date Object

The **JavaScript date** object can be used to get year, month and day. You can display a timer on the webpage by the help of JavaScript date object. You can use different Date constructors to create date object. It provides methods to get and set day, month, year, hour, minute and seconds.

## Constructor : You can use 4 variant of Date constructor to create date object.

1. Date()
2. Date(milliseconds)
3. Date(dateString)
4. Date(year, month, day, hours, minutes, seconds, milliseconds)

## JavaScript Date Methods

The important methods of date object are as follows:

Date Object Methods

|  |  |
| --- | --- |
| **Method** | **Description** |
| [getDate()](http://www.w3schools.com/jsref/jsref_getdate.asp) | Returns the day of the month (from 1-31) |
| [getDay()](http://www.w3schools.com/jsref/jsref_getday.asp) | Returns the day of the week (from 0-6) |
| [getFullYear()](http://www.w3schools.com/jsref/jsref_getfullyear.asp) | Returns the year |
| [getHours()](http://www.w3schools.com/jsref/jsref_gethours.asp) | Returns the hour (from 0-23) |
| [getMilliseconds()](http://www.w3schools.com/jsref/jsref_getmilliseconds.asp) | Returns the milliseconds (from 0-999) |
| [getMinutes()](http://www.w3schools.com/jsref/jsref_getminutes.asp) | Returns the minutes (from 0-59) |
| [getMonth()](http://www.w3schools.com/jsref/jsref_getmonth.asp) | Returns the month (from 0-11) |
| [getSeconds()](http://www.w3schools.com/jsref/jsref_getseconds.asp) | Returns the seconds (from 0-59) |
| [getTime()](http://www.w3schools.com/jsref/jsref_gettime.asp) | Returns the number of milliseconds since midnight Jan 1 1970, and a specified date |
| [getTimezoneOffset()](http://www.w3schools.com/jsref/jsref_gettimezoneoffset.asp) | Returns the time difference between UTC time and local time, in minutes |
| [getUTCDate()](http://www.w3schools.com/jsref/jsref_getutcdate.asp) | Returns the day of the month, according to universal time (from 1-31) |
| [getUTCDay()](http://www.w3schools.com/jsref/jsref_getutcday.asp) | Returns the day of the week, according to universal time (from 0-6) |
| [getUTCFullYear()](http://www.w3schools.com/jsref/jsref_getutcfullyear.asp) | Returns the year, according to universal time |
| [getUTCHours()](http://www.w3schools.com/jsref/jsref_getutchours.asp) | Returns the hour, according to universal time (from 0-23) |
| [getUTCMilliseconds()](http://www.w3schools.com/jsref/jsref_getutcmilliseconds.asp) | Returns the milliseconds, according to universal time (from 0-999) |
| [getUTCMinutes()](http://www.w3schools.com/jsref/jsref_getutcminutes.asp) | Returns the minutes, according to universal time (from 0-59) |
| [getUTCMonth()](http://www.w3schools.com/jsref/jsref_getutcmonth.asp) | Returns the month, according to universal time (from 0-11) |
| [getUTCSeconds()](http://www.w3schools.com/jsref/jsref_getutcseconds.asp) | Returns the seconds, according to universal time (from 0-59) |
| getYear() | Deprecated. Use the [getFullYear()](http://www.w3schools.com/jsref/jsref_getfullyear.asp) method instead |
| [now()](http://www.w3schools.com/jsref/jsref_now.asp) | Returns the number of milliseconds since midnight Jan 1, 1970 |
| [parse()](http://www.w3schools.com/jsref/jsref_parse.asp) | Parses a date string and returns the number of milliseconds since January 1, 1970 |
| [setDate()](http://www.w3schools.com/jsref/jsref_setdate.asp) | Sets the day of the month of a date object |
| [setFullYear()](http://www.w3schools.com/jsref/jsref_setfullyear.asp) | Sets the year of a date object |
| [setHours()](http://www.w3schools.com/jsref/jsref_sethours.asp) | Sets the hour of a date object |
| [setMilliseconds()](http://www.w3schools.com/jsref/jsref_setmilliseconds.asp) | Sets the milliseconds of a date object |
| [setMinutes()](http://www.w3schools.com/jsref/jsref_setminutes.asp) | Set the minutes of a date object |
| [setMonth()](http://www.w3schools.com/jsref/jsref_setmonth.asp) | Sets the month of a date object |
| [setSeconds()](http://www.w3schools.com/jsref/jsref_setseconds.asp) | Sets the seconds of a date object |
| [setTime()](http://www.w3schools.com/jsref/jsref_settime.asp) | Sets a date to a specified number of milliseconds after/before January 1, 1970 |
| [setUTCDate()](http://www.w3schools.com/jsref/jsref_setutcdate.asp) | Sets the day of the month of a date object, according to universal time |
| [setUTCFullYear()](http://www.w3schools.com/jsref/jsref_setutcfullyear.asp) | Sets the year of a date object, according to universal time |
| [setUTCHours()](http://www.w3schools.com/jsref/jsref_setutchours.asp) | Sets the hour of a date object, according to universal time |
| [setUTCMilliseconds()](http://www.w3schools.com/jsref/jsref_setutcmilliseconds.asp) | Sets the milliseconds of a date object, according to universal time |
| [setUTCMinutes()](http://www.w3schools.com/jsref/jsref_setutcminutes.asp) | Set the minutes of a date object, according to universal time |
| [setUTCMonth()](http://www.w3schools.com/jsref/jsref_setutcmonth.asp) | Sets the month of a date object, according to universal time |
| [setUTCSeconds()](http://www.w3schools.com/jsref/jsref_setutcseconds.asp) | Set the seconds of a date object, according to universal time |
| setYear() | Deprecated. Use the [setFullYear()](http://www.w3schools.com/jsref/jsref_setfullyear.asp) method instead |
| [toDateString()](http://www.w3schools.com/jsref/jsref_todatestring.asp) | Converts the date portion of a Date object into a readable string |
| toGMTString() | Deprecated. Use the [toUTCString()](http://www.w3schools.com/jsref/jsref_toutcstring.asp) method instead |
| [toISOString()](http://www.w3schools.com/jsref/jsref_toisostring.asp) | Returns the date as a string, using the ISO standard |
| [toJSON()](http://www.w3schools.com/jsref/jsref_tojson.asp) | Returns the date as a string, formatted as a JSON date |
| [toLocaleDateString()](http://www.w3schools.com/jsref/jsref_tolocaledatestring.asp) | Returns the date portion of a Date object as a string, using locale conventions |
| [toLocaleTimeString()](http://www.w3schools.com/jsref/jsref_tolocaletimestring.asp) | Returns the time portion of a Date object as a string, using locale conventions |
| [toLocaleString()](http://www.w3schools.com/jsref/jsref_tolocalestring.asp) | Converts a Date object to a string, using locale conventions |
| [toString()](http://www.w3schools.com/jsref/jsref_tostring_date.asp) | Converts a Date object to a string |
| [toTimeString()](http://www.w3schools.com/jsref/jsref_totimestring.asp) | Converts the time portion of a Date object to a string |
| [toUTCString()](http://www.w3schools.com/jsref/jsref_toutcstring.asp) | Converts a Date object to a string, according to universal time |
| [UTC()](http://www.w3schools.com/jsref/jsref_utc.asp) | Returns the number of milliseconds in a date since midnight of January 1, 1970, according to UTC time |
| [valueOf()](http://www.w3schools.com/jsref/jsref_valueof_date.asp) | Returns the primitive value of a Date object |

### JavaScript Date Example

Let's see the simple example to print date object. It prints date and time both.

Current Date and Time: <span id="txt"></span>

<script>

var today=new Date();

document.getElementById('txt').innerHTML=today;

</script>

Output:

Current Date and Time: Mon Dec 25 2017 12:27:44 GMT+0530 (India Standard Time)

Let's see another code to print date/month/year.

<script>

var date=new Date();

var day=date.getDate();

var month=date.getMonth()+1;

var year=date.getFullYear();

document.write("<br>Date is: "+day+"/"+month+"/"+year);

</script>

Output:

Date is: 25/12/2017

### JavaScript Current Time Example

Let's see the simple example to print current time of system.

Current Time: <span id="txt"></span>

<script>

var today=new Date();

var h=today.getHours();

var m=today.getMinutes();

var s=today.getSeconds();

document.getElementById('txt').innerHTML=h+":"+m+":"+s;

</script>

Output:

Current Time: 12:27:44

### JavaScript Digital Clock Example

Let's see the simple example to display digital clock using JavaScript date object.

There are two ways to set interval in JavaScript: by setTimeout() or setInterval() method.

Current Time: <span id="txt"></span>

<script>

window.onload=function(){getTime();}

function getTime()

{

var today=new Date();

var h=today.getHours();

var m=today.getMinutes();

var s=today.getSeconds();

// add a zero in front of numbers<10

m=checkTime(m);

s=checkTime(s);

document.getElementById('txt').innerHTML=h+":"+m+":"+s;

setTimeout(function(){getTime()},1000);

}

//setInterval("getTime()",1000);//another way

function checkTime(i){

if (i<10){

  i="0" + i;

 }

return i;

}

</script>

Output:

Current Time: 12:28:01

* **JavaScript Math Object**

The **JavaScript math** object provides several constants and methods to perform mathematical operation. Unlike date object, it doesn't have constructors.

|  |  |
| --- | --- |
| **Property** | **Description** |
| [E](http://www.w3schools.com/jsref/jsref_e.asp) | Returns Euler's number (approx. 2.718) |
| [LN2](http://www.w3schools.com/jsref/jsref_ln2.asp) | Returns the natural logarithm of 2 (approx. 0.693) |
| [LN10](http://www.w3schools.com/jsref/jsref_ln10.asp) | Returns the natural logarithm of 10 (approx. 2.302) |
| [LOG2E](http://www.w3schools.com/jsref/jsref_log2e.asp) | Returns the base-2 logarithm of E (approx. 1.442) |
| [LOG10E](http://www.w3schools.com/jsref/jsref_log10e.asp) | Returns the base-10 logarithm of E (approx. 0.434) |
| [PI](http://www.w3schools.com/jsref/jsref_pi.asp) | Returns PI (approx. 3.14) |
| [SQRT1\_2](http://www.w3schools.com/jsref/jsref_sqrt1_2.asp) | Returns the square root of 1/2 (approx. 0.707) |
| [SQRT2](http://www.w3schools.com/jsref/jsref_sqrt2.asp) | Returns the square root of 2 (approx. 1.414) |

|  |  |
| --- | --- |
| **Method** | **Description** |
| [abs(x)](http://www.w3schools.com/jsref/jsref_abs.asp) | Returns the absolute value of x |
| [acos(x)](http://www.w3schools.com/jsref/jsref_acos.asp) | Returns the arccosine of x, in radians |
| [asin(x)](http://www.w3schools.com/jsref/jsref_asin.asp) | Returns the arcsine of x, in radians |
| [atan(x)](http://www.w3schools.com/jsref/jsref_atan.asp) | Returns the arctangent of x as a numeric value between -PI/2 and PI/2 radians |
| [atan2(y, x)](http://www.w3schools.com/jsref/jsref_atan2.asp) | Returns the arctangent of the quotient of its arguments |
| [ceil(x)](http://www.w3schools.com/jsref/jsref_ceil.asp) | Returns x, rounded upwards to the nearest integer |
| [cos(x)](http://www.w3schools.com/jsref/jsref_cos.asp) | Returns the cosine of x (x is in radians) |
| [exp(x)](http://www.w3schools.com/jsref/jsref_exp.asp) | Returns the value of Ex |
| [floor(x)](http://www.w3schools.com/jsref/jsref_floor.asp) | Returns x, rounded downwards to the nearest integer |
| [log(x)](http://www.w3schools.com/jsref/jsref_log.asp) | Returns the natural logarithm (base E) of x |
| [max(x, y, z, ..., n)](http://www.w3schools.com/jsref/jsref_max.asp) | Returns the number with the highest value |
| [min(x, y, z, ..., n)](http://www.w3schools.com/jsref/jsref_min.asp) | Returns the number with the lowest value |
| [pow(x, y)](http://www.w3schools.com/jsref/jsref_pow.asp) | Returns the value of x to the power of y |
| [random()](http://www.w3schools.com/jsref/jsref_random.asp) | Returns a random number between 0 and 1 |
| [round(x)](http://www.w3schools.com/jsref/jsref_round.asp) | Rounds x to the nearest integer |
| [sin(x)](http://www.w3schools.com/jsref/jsref_sin.asp) | Returns the sine of x (x is in radians) |
| [sqrt(x)](http://www.w3schools.com/jsref/jsref_sqrt.asp) | Returns the square root of x |
| [tan(x)](http://www.w3schools.com/jsref/jsref_tan.asp) | Returns the tangent of an angle |

Math.sqrt(n)

The JavaScript math.sqrt(n) method returns the square root of the given number.

Square Root of 17 is: **<span** id="p1"**></span>**

**<script>**

document.getElementById('p1').innerHTML=Math.sqrt(17);

**</script>**

Output:

Square Root of 17 is: 4.123105625617661

Math.random()

The JavaScript math.random() method returns the random number between 0 to 1.

Random Number is: **<span** id="p2"**></span>**

**<script>**

document.getElementById('p2').innerHTML=Math.random();

**</script>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=jsmath2)

Output:

Random Number is: 0.19599995675218063

Math.pow(m,n)

The JavaScript math.pow(m,n) method returns the m to the power of n that is mn.

3 to the power of 4 is: **<span** id="p3"**></span>**

**<script>**

document.getElementById('p3').innerHTML=Math.pow(3,4);

**</script>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=jsmath3)

Output:

3 to the power of 4 is: 81

Math.floor(n)

The JavaScript math.floor(n) method returns the lowest integer for the given number. For example 3 for 3.7, 5 for 5.9 etc.

Floor of 4.6 is: **<span** id="p4"**></span>**

**<script>**

document.getElementById('p4').innerHTML=Math.floor(4.6);

**</script>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=jsmath4)

Output:

Floor of 4.6 is: 4

Math.ceil(n)

The JavaScript math.ceil(n) method returns the largest integer for the given number. For example 4 for 3.7, 6 for 5.9 etc.

Ceil of 4.6 is: **<span** id="p5"**></span>**

**<script>**

document.getElementById('p5').innerHTML=Math.ceil(4.6);

**</script>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=jsmath5)

Output:

Ceil of 4.6 is: 5

Math.round(n)

The JavaScript math.round(n) method returns the rounded integer nearest for the given number. If fractional part is equal or greater than 0.5, it goes to upper value 1 otherwise lower value 0. For example 4 for 3.7, 3 for 3.3, 6 for 5.9 etc.

Round of 4.3 is: **<span** id="p6"**></span><br>**

Round of 4.7 is: **<span** id="p7"**></span>**

**<script>**

document.getElementById('p6').innerHTML=Math.round(4.3);

document.getElementById('p7').innerHTML=Math.round(4.7);

**</script>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=jsmath6)

Output:

Round of 4.3 is: 4

Round of 4.7 is: 5

Math.abs(n)

The JavaScript math.abs(n) method returns the absolute value for the given number. For example 4 for -4, 6.6 for -6.6 etc.

Absolute value of -4 is: **<span** id="p8"**></span>**

**<script>**

document.getElementById('p8').innerHTML=Math.abs(-4);

**</script>**

[**Test it Now**](http://www.javatpoint.com/oprweb/test.jsp?filename=jsmath7)

Output:

Absolute value of -4 is: 4

* **Browser Object Model**

[**Browser Object Model (BOM)**](https://www.javatpoint.com/browser-object-model)

The **Browser Object Model** (BOM) is used to interact with the browser. The default object of browser is window means you can call all the functions of window by specifying window or directly. For example:

1. window.alert("hello javatpoint");

is same as:

1. alert("hello javatpoint");

You can use a lot of properties (other objects) defined underneath the window object like document, history, screen, navigator, location, innerHeight, innerWidth,

# javascript object model

# Window Object

1. [Window Object](https://www.javatpoint.com/window-object)
2. [Properties of Window Object](https://www.javatpoint.com/window-object)
3. [Methods of Window Object](https://www.javatpoint.com/window-object)
4. [Example of Window Object](https://www.javatpoint.com/window-object)

The **window object** represents a window in browser. An object of window is created automatically by the browser.

Window is the object of browser, **it is not the object of javascript**. The javascript objects are string, array, date etc.

## Window Object Properties

|  |  |
| --- | --- |
| **Property** | **Description** |
| [closed](https://www.w3schools.com/jsref/prop_win_closed.asp) | Returns a Boolean value indicating whether a window has been closed or not |
| [console](https://www.w3schools.com/jsref/obj_console.asp) | Returns a reference to the Console object, which provides methods for logging information to the browser's console ([See Console object](https://www.w3schools.com/jsref/obj_console.asp)) |
| [defaultStatus](https://www.w3schools.com/jsref/prop_win_defaultstatus.asp) | Sets or returns the default text in the statusbar of a window |
| [document](https://www.w3schools.com/jsref/dom_obj_document.asp) | Returns the Document object for the window ([See Document object](https://www.w3schools.com/jsref/dom_obj_document.asp)) |
| [frameElement](https://www.w3schools.com/jsref/prop_win_frameElement.asp) | Returns the <iframe> element in which the current window is inserted |
| [frames](https://www.w3schools.com/jsref/prop_win_frames.asp) | Returns all <iframe> elements in the current window |
| [history](https://www.w3schools.com/jsref/obj_history.asp) | Returns the History object for the window ([See History object](https://www.w3schools.com/jsref/obj_history.asp)) |
| [innerHeight](https://www.w3schools.com/jsref/prop_win_innerheight.asp) | Returns the height of the window's content area (viewport) including scrollbars |
| [innerWidth](https://www.w3schools.com/jsref/prop_win_innerheight.asp) | Returns the width of a window's content area (viewport) including scrollbars |
| [length](https://www.w3schools.com/jsref/prop_win_length.asp) | Returns the number of <iframe> elements in the current window |
| [localStorage](https://www.w3schools.com/jsref/prop_win_localstorage.asp) | Allows to save key/value pairs in a web browser. Stores the data with no expiration date |
| [location](https://www.w3schools.com/jsref/obj_location.asp) | Returns the Location object for the window ([See Location object](https://www.w3schools.com/jsref/obj_location.asp)) |
| [name](https://www.w3schools.com/jsref/prop_win_name.asp) | Sets or returns the name of a window |
| [navigator](https://www.w3schools.com/jsref/obj_navigator.asp) | Returns the Navigator object for the window ([See Navigator object](https://www.w3schools.com/jsref/obj_navigator.asp)) |
| [opener](https://www.w3schools.com/jsref/prop_win_opener.asp) | Returns a reference to the window that created the window |
| [outerHeight](https://www.w3schools.com/jsref/prop_win_outerheight.asp) | Returns the height of the browser window, including toolbars/scrollbars |
| [outerWidth](https://www.w3schools.com/jsref/prop_win_outerheight.asp) | Returns the width of the browser window, including toolbars/scrollbars |
| [pageXOffset](https://www.w3schools.com/jsref/prop_win_pagexoffset.asp) | Returns the pixels the current document has been scrolled (horizontally) from the upper left corner of the window |
| [pageYOffset](https://www.w3schools.com/jsref/prop_win_pagexoffset.asp) | Returns the pixels the current document has been scrolled (vertically) from the upper left corner of the window |
| [parent](https://www.w3schools.com/jsref/prop_win_parent.asp) | Returns the parent window of the current window |
| [screen](https://www.w3schools.com/jsref/obj_screen.asp) | Returns the Screen object for the window [(See Screen object)](https://www.w3schools.com/jsref/obj_screen.asp) |
| [screenLeft](https://www.w3schools.com/jsref/prop_win_screenleft.asp) | Returns the horizontal coordinate of the window relative to the screen |
| [screenTop](https://www.w3schools.com/jsref/prop_win_screenleft.asp) | Returns the vertical coordinate of the window relative to the screen |
| [screenX](https://www.w3schools.com/jsref/prop_win_screenx.asp) | Returns the horizontal coordinate of the window relative to the screen |
| [screenY](https://www.w3schools.com/jsref/prop_win_screenx.asp) | Returns the vertical coordinate of the window relative to the screen |
| [sessionStorage](https://www.w3schools.com/jsref/prop_win_sessionstorage.asp) | Allows to save key/value pairs in a web browser. Stores the data for one session |
| scrollX | An alias of [pageXOffset](https://www.w3schools.com/jsref/prop_win_pagexoffset.asp) |
| scrollY | An alias of [pageYOffset](https://www.w3schools.com/jsref/prop_win_pagexoffset.asp) |
| [self](https://www.w3schools.com/jsref/prop_win_self.asp) | Returns the current window |
| [status](https://www.w3schools.com/jsref/prop_win_status.asp) | Sets or returns the text in the statusbar of a window |
| [top](https://www.w3schools.com/jsref/prop_win_top.asp) | Returns the topmost browser window |

**Methods of window object**

The important methods of window object are as follows:

|  |  |
| --- | --- |
| **Method** | **Description** |
| alert() | displays the alert box containing message with ok button. |
| confirm() | displays the confirm dialog box containing message with ok and cancel button. |
| prompt() | displays a dialog box to get input from the user. |
| open() | opens the new window. |
| close() | closes the current window. |
| setTimeout() | performs action after specified time like calling function, evaluating expressions etc. |

**Example of alert() in javascript**

It displays alert dialog box. It has message and ok button.

**<script** type="text/javascript"**>**

function msg(){

alert("Hello Alert Box");

}

**</script>**

**<input** type="button" value="click" onclick="msg()"**/>**

**Example of confirm() in javascript**

It displays the confirm dialog box. It has message with ok and cancel buttons.

**<script** type="text/javascript"**>**

function msg(){

var v= confirm("Are u sure?");

if(v==true){

alert("ok");

}

else{

alert("cancel");  }

}

**</script>**

**<input** type="button" value="delete record" onclick="msg()"**/>**

**Example of prompt() in javascript**

It displays prompt dialog box for input. It has message and textfield.

**<script** type="text/javascript"**>**

function msg(){

var v= prompt("Who are you?");

alert("I am "+v);

}

**</script>**

**<input** type="button" value="click" onclick="msg()"**/>**

**Example of open() in javascript**

It displays the content in a new window.

**<script** type="text/javascript"**>**

function msg(){

open("http://www.javatpoint.com");

}

**</script>**

**<input** type="button" value="javatpoint" onclick="msg()"**/>**

**Example of setTimeout() in javascript**

It performs its task after the given milliseconds.

**<script** type="text/javascript"**>**

function msg(){

setTimeout(

function(){

alert("Welcome to Javatpoint after 2 seconds")

},2000);

}

**</script>**

**<input** type="button" value="click" onclick="msg()"**/>**

# Document Object Model

1. [Document Object](https://www.javatpoint.com/document-object-model)
2. [Properties of document object](https://www.javatpoint.com/document-object-model)
3. [Methods of document object](https://www.javatpoint.com/document-object-model)
4. [Example of document object](https://www.javatpoint.com/document-object-model)

The **document object** represents the whole html document.

When html document is loaded in the browser, it becomes a document object. It is the **root element** that represents the html document. It has properties and methods. By the help of document object, we can add dynamic content to our web page. As mentioned earlier, it is the object of window. So

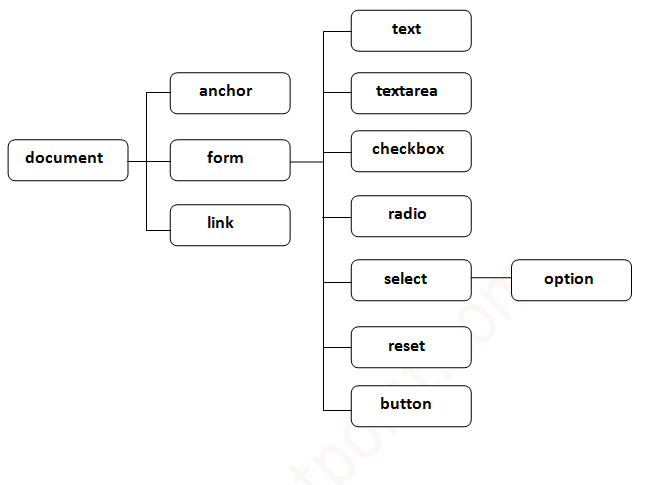
1. window.document

Is same as

1. document

According to W3C - *"The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."*

## Properties of document object

Let's see the properties of document object that can be accessed and modified by the document object. 

## Methods of document object

We can access and change the contents of document by its methods.

The important methods of document object are as follows:

|  |  |
| --- | --- |
| **Method** | **Description** |
| write("string") | writes the given string on the doucment. |
| writeln("string") | writes the given string on the doucment with newline character at the end. |
| getElementById() | returns the element having the given id value. |
| getElementsByName() | returns all the elements having the given name value. |
| getElementsByTagName() | returns all the elements having the given tag name. |
| getElementsByClassName() | returns all the elements having the given class name. |

### Accessing field value by document object

In this example, we are going to get the value of input text by user. Here, we are using **document.form1.name.value** to get the value of name field.

Here, **document** is the root element that represents the html document.

**form1** is the name of the form.

**name** is the attribute name of the input text.

**value** is the property, that returns the value of the input text.

Let's see the simple example of document object that prints name with welcome message.

<script type="text/javascript">

function printvalue(){

var name=document.form1.name.value;

alert("Welcome: "+name);

}

</script>

<form name="form1">

Enter Name:<input type="text" name="name"/>

<input type="button" onclick="printvalue()" value="print name"/>

</form>

Javascript - document.getElementById() method

The **document.getElementById()** method returns the element of specified id. In the previous page, we have used **document.form1.name.value** to get the value of the input value. Instead of this, we can use document.getElementById() method to get value of the input text. But we need to define id for the input field.

Let's see the simple example of document.getElementById() method that prints cube of the given number.

**<script** type="text/javascript"**>**

function getcube(){

var number=document.getElementById("number").value;

alert(number\*number\*number);

}

**</script>**

**<form>**

Enter No:**<input** type="text" id="number" name="number"**/><br/>**

**<input** type="button" value="cube" onclick="getcube()"**/>**

**</form>**

Bottom of Form

# Javascript - document.getElementsByName() method

The **document.getElementsByName()** method returns all the element of specified name.

The syntax of the getElementsByName() method is given below:

document.getElementsByName("name")

Here, name is required.

### Example of document.getElementsByName() method

In this example, we going to count total number of genders. Here, we are using getElementsByName() method to get all the genders.

<script type="text/javascript">

function totalelements()

{

var allgenders=document.getElementsByName("gender");

alert("Total Genders:"+allgenders.length);

}

</script>

<form>

Male:<input type="radio" name="gender" value="male">

Female:<input type="radio" name="gender" value="female">

<input type="button" onclick="totalelements()" value="Total Genders">

</form>

# Javascript - document.getElementsByTagName() method

The **document.getElementsByTagName()** method returns all the element of specified tag name. The syntax of the getElementsByTagName() method is given below:

document.getElementsByTagName("name")

Here, name is required.

### Example of document.getElementsByTagName() method

In this example, we going to count total number of paragraphs used in the document. To do this, we have called the document.getElementsByTagName("p") method that returns the total paragraphs.

<script type="text/javascript">

function countpara(){

var totalpara=document.getElementsByTagName("p");

alert("total p tags are: "+totalpara.length);

  }

</script>

<p>This is a pragraph</p>

<p>Here we are going to count total number of paragraphs by getElementByTagName() method.</p>

<p>Let's see the simple example</p>

<button onclick="countpara()">count paragraph</button>

### Another example of document.getElementsByTagName() method

In this example, we going to count total number of h2 and h3 tags used in the document.

<script type="text/javascript">

function counth2(){

var totalh2=document.getElementsByTagName("h2");

alert("total h2 tags are: "+totalh2.length);

}

function counth3(){

var totalh3=document.getElementsByTagName("h3");

alert("total h3 tags are: "+totalh3.length);

}

</script>

<h2>This is h2 tag</h2>

<h2>This is h2 tag</h2>

<h3>This is h3 tag</h3>

<h3>This is h3 tag</h3>

<h3>This is h3 tag</h3>

<button onclick="counth2()">count h2</button>

<button onclick="counth3()">count h3</button>

#### Output of the above example

## This is h2 tag

## This is h2 tag

### This is h3 tag

### This is h3 tag

### This is h3 tag

count h2 count h3

HTML/DOM events. So before learning JavaScript, let’s have some idea about events.

|  |  |
| --- | --- |
| **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. |

### JavaScript Form Object

Properties of form object, action – elements[], encoding, length, method, name, target, button, checkbox, FileUpload, hidden, password, radio, reset, select, submit, text, textarea and methods of form object.

#### Form Object:

form object is a Browser object of JavaScript used to access an HTML form. If a user wants to access all forms within a document then he can use the forms array. The form object is actually a property of document object that is uniquely created by the browser for each form present in a document. The properties and methods associated with form object are used to access the form fields, attributes and controls associated with forms.

#### Properties of Form Object:

* action
* elements[]
* encoding
* length
* method
* name
* target
* button
* checkbox
* FileUpload
* hidden
* password
* radio
* reset
* select
* submit
* text
* textarea

#### action:

action property of form object is used to access the action attribute present in HTML associated with the <form> tag. This property is a read or write property and its value is a string.

#### elements[]:

elements property of form object is an array used to access any element of the form. It contains all fields and controls present in the form. The user can access any element associated with the form by using the looping concept on the elements array.

#### encoding:

The encoding property of a form object is used to access the enctype attribute present in HTML associated with the <form> tag. This property is a read or write property and its value is a string. This property helps determine the way of encoding the form data.

#### length:

length property of form object is used to specify the number of elements in the form. This denotes the length of the elements array associated with the form.

#### method:

method property of form object is used to access the method attribute present in HTML associated with the <form> tag. This property is a read or write property and its value is a string. This property helps determine the method by which the form is submitted.

#### name:

name property of form object denotes the form name.

#### target:

target property of form object is used to access the target attribute present in HTML associated with the <form> tag. This property denotes the name of the target window to which form it is to be submitted into.

#### button:

The button property of form object denotes the button GUI control placed in the form.

#### checkbox:

checkbox property of form object denotes the checkbox field placed in the form.

#### FileUpload:

FileUpload property of form object denotes the file upload field placed in the form..

#### hidden:

The hidden property of form object denotes the hidden field placed in the form.

#### password:

password property of form object denotes the object that is placed as a password field in the form.

#### radio:

radio property of form object denotes the radio button field placed in the form.

#### reset:

As the name implies, the reset property of form object denotes the object placed as reset button in the form.

#### select:

select property of form object denotes the selection list object placed in the form.

#### submit:

submit property of form object denotes the submit button field that is placed in the form.

#### text:

text property of form object denotes the text field placed in the form.

#### textarea:

textarea property of form object denotes the text area field placed in the form.

#### Methods of form object:

* reset()
* submit()
* handleEvent()

#### reset():

reset() method of form object is used to reset a form.

#### submit():

submit() method of form object is used to submit a form.

#### handleEvent():

handleEvent() method of form object is used to start or invoke a form’s event handler for a specified event.

### JavaScript Frame Object

frame object, properties of frame object, frames, self, name, length, parent, methods of frame object, blur(), focus(), setInterval(), clearInterval(), setTimeout(expression, milliseconds), clearTimeout() and events associated with frame object.

#### Frame Object:

The frame object is a browser object of JavaScript used for accessing HTML frames. The user can use frames array to access all frames within a window. Using the indexing concept, users can access the frames array.

* The frames array index always starts with zero and not 1.
* The frame object is actually a child of the window object. These objects are created automatically by the browser and help users to control loading and accessing of frames.
* The properties and methods of frame object are similar to that of Window object in JavaScript.
* The frame object does not support close() method that is supported by window object.
* Using the <FRAMESET> document creates frame objects and each frame created is thus a property of window object.

#### Properties of frame object:

* frames
* name
* length
* parent
* self

#### frames:

The frames property of frame object denotes a collection or array of frames in a window and also in a frame set.

#### self:

As the name implies, the self property of frames object denotes the current frame. Using self property, the user can access properties of the current frame window.

#### name:

The name property of frame object denotes the name of the frame. The method of denoting the name attribute is performed by using the name attribute of the <frame> tag.

#### length:

The frames array has all the frames present within a window and the length property of the frame object denotes the length of the frames array or gives the number of frames present in a window or a frames array.

#### parent:

As the name implies, the parent property of frames object denotes the parent frame of the current frame.

#### Methods of frame object:

* blur()
* focus()
* setInterval()
* clearInterval()
* setTimeout(expression, milliseconds)
* clearTimeout(timeout)

#### blur():

blur() method of frame object removes focus from the object.

#### focus():

focus() method of frame object gives focus to the object.

#### setInterval():

setInterval() method of frame object is used to call a function of JavaScript or to evaluate an expression after the time interval specified in arguments has expired. The time interval in arguments is always specified in milliseconds.

#### clearInterval():

clearInterval method of frame object is used to cancel the corresponding defined setInterval method. This is written by referencing the setInterval method using its ID or variable.

#### setTimeout(expression, milliseconds):

setTimeout method of frame object can be used to execute any function, or access any method or property after a specified time interval given to this method as argument.

General syntax for the method setTimeout() is as below:

|  |
| --- |
| **setTimeout**(expression, milliseconds) |

**For example:**

|  |
| --- |
| **exforsys**=setTimeout ("test()", 3000) |

The time is always specified in milliseconds and in the above statement, the function test() is called after the specified time of 3000 milliseconds (3 seconds). This is stored in variable named exforsys.

There is confusion about the similarity between setTimeout() method and setInterval() method. The main difference between the two methods is the setInterval method will repeatedly call the referenced function or evaluate the expression until the user leaves the document. In the setTimeout method, the call executes only once after the specified time interval given as argument.

#### clearTimeout():

clearTimeout method of frame object is used to clear a specified setTimeout method. This is written by referencing the setTimeout method using its ID or variable.

General syntax for the method clearTimeout is as below:

|  |
| --- |
| **clearTimeout** ID\_of\_setTimeout |

**For example**

|  |
| --- |
| **clearTimeout** exforsys |

The above statement clears the setTimeout associated wit the ID named as exforsys, created in the earlier example.

#### Events associated with frame object:

Though the frame object and frames array have no event handlers associated directly with them, the following event handlers are used to access and control frame objects and frames array:

* onBlur
* onFocus
* OnLoad
* OnUnLoad