**String 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 |

**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 |
| [fromCharCode()](http://www.w3schools.com/jsref/jsref_fromcharcode.asp) | Converts Unicode values to 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 |
| [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 |
| [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 |

JavaScript global functions can be used on all JavaScript data types.

These are the most relevant methods, when working with numbers:

|  |  |
| --- | --- |
| **Method** | **Description** |
| Number() | Returns a number, converted from its argument. |
| parseFloat() | Parses its argument and returns a floating point number |
| parseInt() | Parses its argument and returns an integer |

**Number Methods**

JavaScript number methods are methods that can be used on numbers:

|  |  |
| --- | --- |
| **Method** | **Description** |
| toString() | Returns a number as a string |
| toExponential() | Returns a string, with a number rounded and written using exponential notation. |
| toFixed() | Returns a string, with a number rounded and written with a specified number of decimals. |
| toPrecision() | Returns a string, with a number written with a specified length |
| valueOf() | Returns a number as a number |

|  |  |
| --- | --- |
|  |  |

**The toString() Method**

**toString()** returns a number as a string.

All number methods can be used on any type of numbers, literals, variables, or expressions:

**Example**

var x = 123;  
x.toString();            // returns 123 from variable x  
(123).toString();        // returns 123 from literal 123  
(100 + 23).toString();   // returns 123 from expression 100 + 23

**The toExponential() Method**

**toExponential()** returns a string, with a number rounded and written using exponential notation.

A parameter defines the number of character behind the decimal point:

**Example**

var x = 9.656;  
x.toExponential(2);     // returns 9.66e+0  
x.toExponential(4);     // returns 9.6560e+0  
x.toExponential(6);     // returns 9.656000e+0

**The toFixed() Method**

**toFixed()** returns a string, with the number written with a specified number of decimals:

**Example**

var x = 9.656;  
x.toFixed(0);           // returns 10  
x.toFixed(2);           // returns 9.66  
x.toFixed(4);           // returns 9.6560  
x.toFixed(6);           // returns 9.656000

**The toPrecision() Method**

**toPrecision()** returns a string, with a number written with a specified length:

**Example**

var x = 9.656;  
x.toPrecision();        // returns 9.656  
x.toPrecision(2);       // returns 9.7  
x.toPrecision(4);       // returns 9.656  
x.toPrecision(6);       // returns 9.65600

**Converting Variables to Numbers**

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

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

## Date Get Methods

Get methods are used for getting a part of a date. Here are the most common (alphabetically):

|  |  |
| --- | --- |
| **Method** | **Description** |
| getDate() | Get the day as a number (1-31) |
| getDay() | Get the weekday a number (0-6) |
| getFullYear() | Get the four digit year (yyyy) |
| getHours() | Get the hour (0-23) |
| getMilliseconds() | Get the milliseconds (0-999) |
| getMinutes() | Get the minutes (0-59) |
| getMonth() | Get the month (0-11) |
| getSeconds() | Get the seconds (0-59) |
| getTime() | Get the time (milliseconds since January 1, 1970) |

## The getTime() Method

**getTime()** returns the the number of milliseconds since 01.01.1970:

## Example

<script>  
var d = new Date();  
document.getElementById("demo").innerHTML = d.getTime();  
</script>

## The getFullYear() Method

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

## Example

<script>  
var d = new Date();  
document.getElementById("demo").innerHTML = d.getFullYear();  
</script>

## The getDay() Method

**getDay()** returns the weekday as a number (0-6):

## Example

<script>  
var d = new Date();  
document.getElementById("demo").innerHTML = d.getDay();  
</script>

## Example

<script>  
var d = new Date();  
var days = ["Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday"];  
document.getElementById("demo").innerHTML = days[d.getDay()];  
</script>

Date Set Methods

Set methods are used for setting a part of a date. Here are the most common (alphabitically):

|  |  |
| --- | --- |
| **Method** | **Description** |
| setDate() | Set the day as a number (1-31) |
| setFullYear() | Set the year (optionally month and day yyyy.mm.dd) |
| setHours() | Set the hour (0-23) |
| setMilliseconds() | Set the milliseconds (0-999) |
| setMinutes() | Set the minutes (0-59) |
| setMonth() | Set the month (0-11) |
| setSeconds() | Set the seconds (0-59) |
| setTime() | Set the time (milliseconds since January 1, 1970) |

## The setFullYear() Method

**setFullYear()** sets a date object to a specific date. In this example, to January 14, 2020:

## Example

<script>  
var d = new Date();  
d.setFullYear(2020, 0, 14);  
document.getElementById("demo").innerHTML = d;  
</script>

## The setDate() Method

**setDate()** sets the day of the month (1-31):

## Example

<script>  
var d = new Date();  
d.setDate(20);  
document.getElementById("demo").innerHTML = d;  
</script>

The setDate() method can also be used to **add days** to a date:

## Example

<script>  
var d = new Date();  
d.setDate(d.getDate() + 50);  
document.getElementById("demo").innerHTML = d;  
</script>

## Date  Input - Parsing Dates

If you have an input value (or any string), you can use the **Date.parse()** method to convert it to milliseconds.

**Date.parse()** returns the number of milliseconds between the date and January 1, 1970:

## Example

<script>  
var msec = Date.parse("March 21, 2012");  
document.getElementById("demo").innerHTML = msec;  
</script>

You can then use the number of milliseconds to **convert it to a date** object:

## Example

<script>  
var msec = Date.parse("March 21, 2012");  
var d = new Date(msec);  
document.getElementById("demo").innerHTML = d;  
</script>

Compare Dates

Dates can easily be compared.

The following example compares today's date with January 14, 2100:

## Example

var today, someday, text;  
today = new Date();  
someday = new Date();  
someday.setFullYear(2100, 0, 14);  
  
if (someday > today) {  
    text = "Today is before January 14, 2100.";  
} else {  
    text = "Today is after January 14, 2100.";  
}  
document.getElementById("demo").innerHTML = text;