Number Methods

Number methods help you work with numbers.

**Number Methods and Properties**

Primitive values (like 2.781 or 2020), cannot have properties and methods (because they are not objects).

But with JavaScript, methods and properties are also available to primitive values, because JavaScript treats primitive values as objects when executing methods and properties.

**toString() Method**

The toString() method returns a number as a string.

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

**Example**

<script>

let x = 101;

let st = x.toString();

document.write( st +"<br>"); //returns "101"

document.write( (45.88).toString() +"<br>"); //returns "45.88"

document.write( (39+60).toString() +"<br>"); //returns "99"

</script>

**valueOf() Method**

valueOf() returns a number as a number.

Example

<script>

var n = 111;  
 document.write( n.valueOf() );            // returns 111   
 document.write( ("111").valueOf() );        // returns 111  
 document.write( (111 + 111).valueOf() );   // returns 222

</script>

In JavaScript, a number can be a primitive value (typeof = number) or an object (typeof = object).

The valueOf() method is used internally in JavaScript to convert Number objects to primitive values.

There is no reason to use it in your code.

All JavaScript data types have a valueOf() and a toString() method.

**toExponential() Method**

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

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

**Example**

<script>

var n = 1000000000;

document.write( n.toExponential() +"<br>"); //returns 1e+7 (def is 0)

document.write( n.toExponential(2) +"<br>"); //returns 1.00e+7

document.write( n.toExponential(5) +"<br>"); //returns 1.00000e+7

</script>

The parameter is optional. If you don't specify it, JavaScript will not round the int.

**toFixed() Method**

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

**Example**

<script>

let n = 10000.7829;

document.write( n.toFixed(0) +"<br>" ); // returns 5 (by default is 0)

document.write( n.toFixed(1) +"<br>"); // returns 4.8

document.write( n.toFixed(5) +"<br>"); // returns 4.78290

document.write( new Number(n.toFixed(1)).toExponential(1) +"<br>");

document.write( (10/4).toFixed(2) );

</script>

**toPrecision() Method**

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

**Example**

<script>

let n = 10000.7829; //width of value is 9digit

document.write( n.toPrecision(5) +"<br>"); // returns 10001

document.write( n.toPrecision(7) +"<br>"); // returns 10000.78

document.write( n.toPrecision(10) +"<br>"); // returns 10000.78290

</script>

**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

These methods are not **number** methods, but **global** JavaScript methods.

**Global Methods**

JavaScript global methods 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() Method**

Number() can be used to convert different types of values to number format:

**Example**

**<script>**

var a=Number(true);

document.write( typeof a +"<br>"); //number

document.write( a +"<br>"); //returns 1

document.write( Number(false) +"<br>"); //returns 0

document.write( Number(null) +"<br>"); //returns 0

document.write( Number("30") +"<br>"); //returns 30

document.write( Number(" 30 ") +"<br>"); //returns 30

document.write( Number("30.33") +"<br>"); //returns 30.33

document.write( Number("30,33") +"<br>"); //returns NaN

document.write( Number("30 33") +"<br>"); //returns NaN

document.write( Number("Siva") +"<br>"); //returns NaN

**</script>**

**Note: If the number cannot be converted, NaN (Not a Number) is returned.**

**Number() Method Used on Dates**

Number() can also convert a date to a number:

**Example**

**<script>**

let d=new Date();

document.write( Number(d) );    // returns value in misec  
**</script>**

**Note: The Number() method above returns the number of milliseconds since 1-Jan-1970 00:00:00.0**

**parseInt() Method**

parseInt() parses a string and returns a whole number. Spaces are allowed. Only the first number is returned:

**Example**

<script>

let a=parseInt("10");

document.write( a +"<br>"); // returns 10

document.write( typeof a +"<br>"); // returns 10

document.write( parseInt("10.33") +"<br>"); // returns 10

document.write( parseInt("30 33") +"<br>"); // returns 30

document.write( parseInt("10 apples") +"<br>"); // returns 10

document.write( parseInt("mangos 50") +"<br>"); // returns NaN

document.write( parseInt("siva") +"<br>"); // returns NaN

document.write( parseInt(true) +"<br>"); // returns NaN

</script>

**Note: If the number cannot be converted, NaN (Not a Number) is returned.**

**parseFloat() Method**

parseFloat() parses a string and returns a number. Spaces are allowed. Only the first number is returned:

**Example**

<script>

let a=parseFloat("10");

document.write( a +"<br>"); // returns 10

document.write( typeof a +"<br>"); // returns 10

document.write( parseFloat("10.33") +"<br>"); // **returns 10.33**

document.write( parseFloat("30 33") +"<br>"); // returns 30

document.write( parseFloat("10 apples") +"<br>"); // returns 10

document.write( parseFloat("mangos 50") +"<br>"); // returns NaN

document.write( parseFloat("siva") +"<br>"); // returns NaN

document.write( parseFloat(true) +"<br>"); // returns NaN

</script>

**Note: If the number cannot be converted, NaN (Not a Number) is returned.**

**Number Properties**

|  |  |
| --- | --- |
| **Property** | **Description** |
| MAX\_VALUE | Returns the largest number possible in JavaScript |
| MIN\_VALUE | Returns the smallest number possible in JavaScript |
| POSITIVE\_INFINITY | Represents infinity (returned on overflow) |
| NEGATIVE\_INFINITY | Represents negative infinity (returned on overflow) |
| NaN | Represents a "Not-a-Number" value |

MAX\_VALUE returns the largest possible number.

Ex var n = Number.MAX\_VALUE;

MIN\_VALUE returns the lowest possible number.

Ex var n = Number.MIN\_VALUE;

POSITIVE\_INFINITY is returned on overflow

Ex var n = Number.POSITIVE\_INFINITY;

Ex var n = 1 / 0;

NEGATIVE\_INFINITY is returned on overflow:

Ex var x = Number.NEGATIVE\_INFINITY;

Ex var x = -1 / 0;

NaN is a JavaScript reserved word indicating that a number is not a legal number.

Ex var n = Number.NaN;

Trying to do arithmetic with a non-numeric string will result in NaN (Not a Number):

Ex var n = 100 / "a";  // NaN

**Example**

<script>

let a = Number.MAX\_VALUE;

document.write( a +"<br>");

document.write( Number.MIN\_VALUE + "<br>");

document.write( Number.POSITIVE\_INFINITY +"<br>");

let b = 20 / 0;

document.write( b +"<br>");

document.write( Number.NEGATIVE\_INFINITY +"<br>");

let c = -10 / 0;

document.write( c +"<br>");

document.write( Number.NaN +"<br>");

let d = Number("ram");

document.write( d +"<br>");

</script>

Number Properties Cannot be Used on Variables

Number properties belongs to the JavaScript's number object wrapper called **Number**.

These properties can only be accessed as Number.MAX\_VALUE.

Using *myNumber*.MAX\_VALUE, where *myNumber* is a variable, expression, or value, will return undefined:

Ex var n = 6;  
 var m = n.MAX\_VALUE;    // m becomes undefined