**Javascript Numbers**

<!DOCTYPE HTML>

<html>

<head></head>

<body>

<h2>**JavaScript Numbers**</h2>

<script>

**/\*JavaScript Numbers**

**JavaScript has only one type of number. Numbers can be written with or without decimals.**

var x = 3.14; // A number with decimals

var y = 3; // A number without decimals

**Extra large or extra small numbers can be written with scientific (exponent) notation:**

var x = 123e5; // 12300000

var y = 123e-5; // 0.00123

**JavaScript Numbers are Always 64-bit Floating Point**

**Unlike many other programming languages, JavaScript does not define different types of numbers, like integers, short, long, floating-point etc.**

**JavaScript numbers are always stored as double precision floating point numbers, following the international IEEE 754 standard.**

**This format stores numbers in 64 bits, where the number (the fraction) is stored in bits 0 to 51, the exponent in bits 52 to 62, and the sign in bit 63:**

**Precision**

**Integers (numbers without a period or exponent notation) are accurate up to 15 digits.**

var x = 999999999999999; // x will be 999999999999999

var y = 9999999999999999; // y will be 10000000000000000

**The maximum number of decimals is 17, but floating point arithmetic is not always 100% accurate:**

var x = 0.2 + 0.1; // x will be 0.30000000000000004 \*/

var x=0.2+0.1;

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

//var x = (0.2 \* 10 + 0.1 \* 10) / 10; // x will be 0.3

**/\*Adding Numbers and Strings**

**WARNING !!**

**JavaScript uses the + operator for both addition and concatenation.**

**Numbers are added. Strings are concatenated.**

var x = 10;

var y = 20;

var z = x + y; // z will be 30 (a number)

**If you add two strings, the result will be a string concatenation:**

var x = "10";

var y = "20";

var z = x + y; // z will be 1020 (a string)

var x = 10;

var y = 20;

var z = "30";

var result = x + y + z; //3030

**JavaScript will try to convert strings to numbers in all numeric operations:**

**This will work:**

var x = "100";

var y = "10";

var z = x / y; // z will be 10

var x = "100";

var y = "10";

var z = x \* y; // z will be 1000

var x = "100";

var y = "10";

var z = x - y; // z will be 90

**But this will not work:**

var x = "100";

var y = "10";

var z = x + y; // z will not be 110 (It will be 10010)

**NaN - Not a Number**

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

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

var x = 100 / "Apple"; // x will be NaN (Not a Number)

**However, if the string contains a numeric value , the result will be a number:**

var x = 100 / "10"; // x will be 10

**You can use the global JavaScript function isNaN() to find out if a value is a number:**

var x = 100 / "Apple";

isNaN(x); // returns true because x is Not a Number

var x = NaN;

var y = 5;

var z = x + y; // z will be NaN

**Or the result might be a concatenation:**

Example

var x = NaN;

var y = "5";

var z = x + y; // z will be NaN5

**NaN is a number: typeof NaN returns number**

typeof NaN; // returns "number"

**Infinity**

**Infinity (or -Infinity) is the value JavaScript will return if you calculate a number outside the largest possible number.**

Example

var myNumber = 2;

while (myNumber != Infinity) { // Execute until Infinity

myNumber = myNumber \* myNumber;

}

**Division by 0 (zero) also generates Infinity:**

Example

var x = 2 / 0; // x will be Infinity

var y = -2 / 0; // y will be -Infinity

typeof Infinity; // returns "number"

Hexadecimal

JavaScript interprets numeric constants as hexadecimal if they are preceded by 0x.

Example

var x = 0xFF; // x will be 255

**Note the difference between (x==y) and (x===y).**

**Comparing two JavaScript objects will always return false.**

**The 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):**

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 characters 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 parameter is optional. If you don't specify it, JavaScript will not round the number.**

**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

**The valueOf() Method**

**valueOf() returns a number as a number.**

Example

var x = 123;

x.valueOf(); // returns 123 from variable x

(123).valueOf(); // returns 123 from literal 123

(100 + 23).valueOf(); // returns 123 from expression 100 + 23

**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**

**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**

**The Number() Method**

**Number() can be used to convert JavaScript variables to numbers:**

Example

Number(true); // returns 1

Number(false); // returns 0

Number("10"); // returns 10

Number(" 10"); // returns 10

Number("10 "); // returns 10

Number(" 10 "); // returns 10

Number("10.33"); // returns 10.33

Number("10,33"); // returns NaN

Number("10 33"); // returns NaN

Number("John"); // returns NaN

var x="100";

document.write(Number(x+10)+10); //10020

**The Number() Method Used on Dates**

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

**Note:The Number() method above returns the number of milliseconds since 1.1.1970.**

var x = new Date("2017-09-30");

document.getElementById("demo").innerHTML = Number(x); // returns 1506729600000

var x = Date("2017-09-30");

document.getElementById("demo").innerHTML = x; //Sun Mar 17 2019 16:10:35 GMT+0530 (India Standard Time)

**The parseInt() Method**

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

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

parseInt("10"); // returns 10

parseInt("10.33"); // returns 10

parseInt("10 20 30"); // returns 10

parseInt("10 years"); // returns 10

parseInt("years 10"); // returns NaN

**The parseFloat() Method**

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

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

parseFloat("10"); // returns 10

parseFloat("10.33"); // returns 10.33

parseFloat("10 20 30"); // returns 10

parseFloat("10 years"); // returns 10

parseFloat("years 10"); // returns NaN

**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**

var x = Number.MAX\_VALUE; //1.7976931348623157e+308

var x = Number.MIN\_VALUE; //5e-324

var x = Number.POSITIVE\_INFINITY; //Infinity

var x = Number.NEGATIVE\_INFINITY; //-Infinity

var x = 1 / 0; //Infinity

var x =-1 / 0; //-Infinity

document.getElementById("demo").innerHTML = Number.NaN; //NaN

var x = 6;

document.getElementById("demo").innerHTML = x.MAX\_VALUE; //undefined

</script>

</body>

</html>