JavaScript Display

Using innerHTML

document.getElementById("demo").innerHTML = 5 + 6;

document.write()

```
document.write(5 + 6);
```

Using document.write() after an HTML document is loaded, will delete all existing HTML

window.alert()

```
window.alert(5 + 6);
```

console.log()

```
console.log(5 + 6);
```

JavaScript Variables

```
var x = 5;
var y = 6;
var z = x + y;
```

```
var person = "John Doe", carName = "Volvo", price = 200;
```

Re-Declaring JavaScript Variables

```
var carName = "Volvo";
var carName;
```

JavaScript Arithmetic

```
var x = 5 + 2 + 3; →10
```

```
var x = "John" + " " + "Doe"; →John Doe
```

```
var x = "5" + 2 + 3;→55
```

JavaScript Data Types

```
var length = 16;                                // Number
var lastName = "Johnson";                       // String
var x = {firstName:"John", lastName:"Doe"};     // Object
```

```
alert(x.name);→john
```

JavaScript Types are Dynamic

```
var x;           // Now x is undefined
x = 5;           // Now x is a Number
x = "John";      // Now x is a String
```

JavaScript Strings

```
var answer1 = "It's
alright";              // Single quote
inside double quotes
var answer2 = "He is called
'Johnny'";     // Single quotes inside
double quotes
var answer3 = 'He is called
"Johnny"';     // Double quotes inside
single quotes
```

JavaScript Numbers

```
var x1 = 34.00;      // Written
with decimals
var x2 = 34;         // Written
without decimals
```

JavaScript Booleans

```
var x = 5;
var y = 5;
var z = 6;
(x == y)        // Returns true
(x == z)        // Returns false
```

JavaScript Arrays

```
var cars =
["Saab", "Volvo", "BMW"];
```

JavaScript Objects

```
var person = {firstName:"John",
lastName:"Doe", age:50, eyeColor:"blue"};
```

The typeof Operator

```
typeof ""              // Returns "string"
typeof "John"          // Returns "string"
typeof "John Doe"      // Returns "string"


typeof 0               // Returns "number"
typeof 314             // Returns "number"
typeof 3.14            // Returns "number"
typeof (3)             // Returns "number"
typeof (3 + 4)         // Returns "number"
```

Undefined

```
var car;    // Value is undefined, type is undefined


typeof {name:'John', age:34} // Returns "object"
typeof [1,2,3,4]             // Returns "object" (not "array", see note
below)
typeof null                  // Returns "object"
typeof function myFunc(){}   // Returns "function"
```

Null

```
var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
person = null;    // Now value is null, but type is still an object
```

JavaScript Functions

```
function myFunction(p1, p2) {
  return p1 * p2;   // The function returns the product of p1 and p2
}
```

Function Return

```
var x = myFunction(4, 3);   // Function is called, return value will end up
in x

function myFunction(a, b) {
  return a * b;             // Function returns the product of a and b
}
```

```
function toCelsius(fahrenheit) {
  return (5/9) * (fahrenheit-32);
}
document.getElementById("demo").innerHTML = toCelsius(77);
```

→ 25

```
function toCelsius(fahrenheit) {
  return (5/9) * (fahrenheit-32);
}
document.getElementById("demo").innerHTML = toCelsius;
```

→ function toCelsius(f) { return (5/9) * (f-32); }

JavaScript Strings

length

```
var txt = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
var sln = txt.length;
```

→ 26

indexOf()

```
var str = "Please locate where 'locate' occurs!";
var pos = str.indexOf("locate");
```

→7

lastIndexOf()

```
var str = "Please locate where 'locate' occurs!";
var pos = str.lastIndexOf("locate");
```

→ 21

Both indexOf(), and lastIndexOf() return -1 if the text is not found

```
var str = "Please locate where 'locate' occurs!";
var pos = str.lastIndexOf("John");
```

→-1

Both methods accept a second parameter as the starting

```
var str = "Please locate where 'locate' occurs!";
var pos = str.indexOf("locate",15);
```

search()

```
var str = "Please locate where 'locate' occurs!";
var pos = str.search("locate");
```

slice()

```
var str = "Apple, Banana, Kiwi";
var res = str.slice(7, 13);
```

→ Banana

```
var str = "Apple, Banana, Kiwi";
var res = str.slice(-12, -6);
```

→ Banana

```
var res = str.slice(7);
```

```
var res = str.slice(-12);
```

→ Banana, Kiwi

substring

```
var str = "Apple, Banana, Kiwi";
var res = str.substring(7, 13);
```

→ Banana

substr

```
var str = "Apple, Banana, Kiwi";
var res = str.substr(7, 6);
```

→Banana

```
var str = "Apple, Banana, Kiwi";
var res = str.substr(7);
```

replace

```
str = "Please visit Microsoft!";
var n = str.replace("Microsoft", "W3Schools");
```

toUpperCase

```
var text1 = "Hello World!";       // String
var text2 = text1.toUpperCase();  // text2 is text1 converted to upper
```

toLowerCase

```
var text1 = "Hello World!";       // String
var text2 = text1.toLowerCase();  // text2 is text1 converted to lower
```

concat

```
var text = "Hello" + " " + "World!";
var text = "Hello".concat(" ", "World!");
```

trim

```
var str = "       Hello World!        ";
alert(str.trim());
```

charAt

```
var str = "HELLO WORLD";
str.charAt(0);            // returns H
```

charCodeAt

```
var str = "HELLO WORLD";
str.charCodeAt(0);
```

```
var str = "HELLO WORLD";
str[0];                   // returns H
```

```
var str = "HELLO WORLD";
str[0] = "A";             // Gives no error, but does not work
str[0];                   // returns H
```

split

```
var txt = "a,b,c,d,e";   // String
txt.split(",");          // Split on commas
txt.split(" ");          // Split on spaces
txt.split("|");          // Split on pipe
```

```
var str = "Hello";

var arr = str.split("");

var text = "";

var i;
```

```
for (i = 0; i < arr.length; i++) {
  text += arr[i] + "<br>"
}
```

→h

E

L

L

O

Numbers

```
var x = 123;
var y = new Number(123);

// typeof x returns number
// typeof y returns object

var x = 500;
var y = new Number(500);

// (x == y) is true because x and y have equal values

var x = new Number(500);
var y = new Number(500);

// (x == y) is false because objects cannot be compared
```

toString()

```
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
```

toExponential()

```
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
```

toPrecision()

```
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
```

valueOf()

```
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
```

Number()

```
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
```

parseInt()

```
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
```

parseFloat()

```
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
```

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

JavaScript Random

```
Math.random();
```

```
Math.floor(Math.random() * 11);      // returns a random integer from 0 to 10
```

```
Math.floor(Math.random() * 101);     // returns a random integer from 0 to 100
```

Boolean() Function

```
Boolean(10 > 9)        // returns true
```

JavaScript Math Object

```
Math.PI;            // returns 3.141592653589793
```

Math.round()

```
Math.round(4.7);    // returns 5
Math.round(4.4);    // returns 4
```

Math.pow()

```
Math.pow(8, 2);
```

Math.sqrt()

```
Math.sqrt(64);
```

Math.abs()

```
Math.abs(-4.7);     // returns 4.7
```

Math.ceil()

```
Math.ceil(4.4);     // returns 5
```

Math.floor()

```
Math.floor(4.7);    // returns 4
```

Creating an Array

```
var cars = ["Saab", "Volvo", "BMW"];

var cars = new Array("Saab", "Volvo", "BMW");

var name = cars[0];
```

Array Properties and Methods

```
var x = cars.length;   // The length property returns the number of
elements
var y = cars.sort();   // The sort() method sorts arrays
```

```
var fruits, text, fLen, i;
fruits = ["Banana", "Orange", "Apple", "Mango"];
fLen = fruits.length;

text = "<ul>";
for (i = 0; i < fLen; i++) {
  text += "<li>" + fruits[i] + "</li>";
}
text += "</ul>";
```

Converting Arrays to Strings

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo").innerHTML = fruits.toString();
```

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
document.getElementById("demo").innerHTML = fruits.join(" * ");
```

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.pop();               // Removes the last element ("Mango") from
fruits
```

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
var x = fruits.pop();       // the value of x is "Mango"
```

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.push("Kiwi");        //  Adds a new element ("Kiwi") to fruits
```

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.shift();             // Removes the first element "Banana" from
fruits
```

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.unshift("Lemon");    // Adds a new element "Lemon" to fruits
```

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.splice(2, 0, "Lemon", "Kiwi");
```

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
fruits.splice(0, 1);         // Removes the first element of fruits
```

```
var myGirls = ["Cecilie", "Lone"];
var myBoys = ["Emil", "Tobias", "Linus"];
```

```
var myChildren = myGirls.concat(myBoys);   // Concatenates (joins) myGirls
and myBoys


var fruits = ["Banana", "Orange", "Lemon", "Apple", "Mango"];
var citrus = fruits.slice(1);


function myArrayMax(arr) {
  return Math.max.apply(null, arr);
}


function myArrayMin(arr) {
  return Math.min.apply(null, arr);
}


var numbers = [45, 4, 9, 16, 25];
var over18 = numbers.filter(myFunction);

function myFunction(value, index, array) {
  return value > 18;
}
```

Date

```
var d = new Date();

var d = new Date(2018, 11, 24, 10, 33, 30);

var d = new Date("October 13, 2014 11:13:00");

var d = new Date(0);

var d = new Date(86400000);

d = new Date();

var d = new Date("2015-03");

var d = new Date("2015-03-25");
```

Get Date Methods

getFullYear() Get the year as a four digit number (yyyy)

getMonth() Get the month as a number (0-11)

getDate() Get the day as a number (1-31)

getHours() Get the hour (0-23)

getMinutes() Get the minute (0-59)

getSeconds() Get the second (0-59)

getMilliseconds() Get the millisecond (0-999)

getTime() Get the time (milliseconds since January 1, 1970)

getDay() Get the weekday as a number (0-6)

Date.now() Get the time. ECMAScript 5.

```
document.getElementById("demo").innerHTML = d.toString();
```

Strict mode

```
"use strict";
myFunction();

function myFunction() {
  y = 3.14;   // This will also cause an error because y is not declared
}
```

JavaScript Events

```
<button onclick="document.getElementById('demo').innerHTML = Date()">The
time is?</button>

<button onclick="displayDate()">The time is?</button>
```

onchange An HTML element has been changed

onclick The user clicks an HTML element

onmouseover The user moves the mouse over an HTML element

onmouseout The user moves the mouse away from an HTML element

onkeydown The user pushes a keyboard key

onload The browser has finished loading the page

The constructor Property

```
"John".constructor                 // Returns function String()  {[native
code]}
(3.14).constructor                 // Returns function Number()  {[native
code]}
false.constructor                  // Returns function Boolean() {[native
code]}
[1,2,3,4].constructor              // Returns function Array()   {[native
code]}
{name:'John',age:34}.constructor   // Returns function Object()  {[native
code]}
new Date().constructor             // Returns function Date()    {[native
code]}
function () {}.constructor         // Returns function Function(){[native
code]}
```

JavaScript Objects

```
var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
```

```
var person = {
  firstName: "John",
  lastName : "Doe",
  id       : 5566,
  fullName : function() {
    return this.firstName + " " + this.lastName;
  }
};

name = person.fullName();
```

JavaScript try and catch

```
try {
   if(x == "") throw "is empty";
   if(isNaN(x)) throw "is not a number";
   x = Number(x);
   if(x > 10) throw "is too high";
   if(x < 5) throw "is too low";
 }
 catch(err) {
   message.innerHTML = "Error: " + err + ".";
 }
 finally {
   document.getElementById("demo").value = "";
 }
```

Objects are Variables

```
var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
```

```
var person = new Object();
person.firstName = "John";
person.lastName = "Doe";
person.age = 50;
person.eyeColor = "blue";
```

```
var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"}

var x = person;
x.age = 10;
```

```
<div id="demo">

<h2>The XMLHttpRequest Object</h2>

<button type="button" onclick="loadDoc()">Change Content</button>

</div>


<script>

function loadDoc() {

  var xhttp = new XMLHttpRequest();
```

```
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementById("demo").innerHTML =
      this.responseText;
    }
  };
  xhttp.open("GET", "ajax_info.txt", true);
  xhttp.send();
}
</script>
```

```
// Storing data:
myObj = {name: "John", age: 31, city: "New York"};
myJSON = JSON.stringify(myObj);
localStorage.setItem("testJSON", myJSON);

// Retrieving data:
text = localStorage.getItem("testJSON");
obj = JSON.parse(text);
document.getElementById("demo").innerHTML = obj.name;
```

```
// returns John
person.name;

// returns John
person["name"];
```