JS

VERY VERY BASIC

Helping Material

https://www.w3schools.com/js/default.asp

Use it as a reference Guide

Where to write Js

Script Tag

```
<script>
document.getElementById("demo").innerHTML = "My First JavaScript";
</script>
```

Or Make a separate Js file and attach like below

```
<script src="/js/myScript1.js"></script>
```

Js Output

Writing into an HTML element, using innerHTML.

Writing into the HTML output using document.write().

Writing into an alert box, using window.alert().

Writing into the browser console, using console.log().

First use console.log() and alert for beginners

Statements

x stores the value 5

y stores the value 6

z stores the value 11

```
var x, y, z;        // Statement 1
x = 5;              // Statement 2
y = 6;              // Statement 3
z = x + y;          // Statement 4
```

Comments

```
var x = 5;    // I will be executed

// var x = 6;    I will NOT be executed
```

Case Sensitive

```
var lastname, lastName;
lastName = "Doe";
lastname = "Peterson";
```

Much Like Algebra

price1, price2, and total, are variables:

```
var price1 = 5;
var price2 = 6;
var total = price1 + price2;
```

Operators

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
**	Exponentiation (ES2016)
/	Division
%	Modulus (Division Remainder)
++	Increment
--	Decrement

Assignment Operators

Operator	Example	Same As
=	x = y	x = y
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y
**=	x **= y	x = x ** y

String Operators

```
var txt1 = "John";
var txt2 = "Doe";
var txt3 = txt1 + " " + txt2;
```

String And Numbers

10
55
Hello5

```
var x = 5 + 5;
var y = "5" + 5;
var z = "Hello" + 5;
```

Comparison Operators

Operator	Description
==	equal to
===	equal value and equal type
!=	not equal
!==	not equal value or not equal type
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to
?	ternary operator

Logical Operators

Operator	Description
&&	logical and
||	logical or
!	logical not

Data Types

JavaScript Types are Dynamic

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

Undefined

Undefined

In JavaScript, a variable without a value, has the value undefined. The type is also undefined.

Null

In JavaScript null is "nothing". It is supposed to be something that doesn't exist.

```
typeof undefined               // undefined
typeof null                    // object

null === undefined             // false
null == undefined              // true
```

Premitive Data

```
typeof "John"     // Returns "string"
typeof 3.14       // Returns "number"
typeof true       // Returns "boolean"
typeof false      // Returns "boolean"
typeof x          // Returns "undefined" (if x has no value)
```

JavaScript Functions

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

Declare and Call a Function

```
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
}
```

Objects Properties and Objects

	car.name = Fiat car.model = 500 car.weight = 850kg car.color = white	car.start() car.drive() car.brake() car.stop()

Defining an Object

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

Property	Property Value
firstName	John
lastName	Doe
age	50
eyeColor	blue

More Advance Usage

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

The **this** Keyword

In a function definition, this refers to the "owner" of the function.

In the example above, this is the person object that "owns" the fullName function.

In other words, this.firstName means the firstName property of this object.

JavaScript Events

HTML events are **"things"** that happen to HTML elements.

When JavaScript is used in HTML pages, JavaScript can **"react"** on these events.

Examples

- An HTML web page has finished loading
- An HTML input field was changed
- An HTML button was clicked

Binding Events

```
<button onclick=“alert(‘you clicked me’)">
   Click Me
</button>
```

Common HTML Events

Event	Description
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

JavaScript Arrays

Instead of

```
var car1 = "Saab";
var car2 = "Volvo";
var car3 = "BMW";
```

We Use Something like this

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

How to Access Array Element

```
var name = cars[0];
```

ForEach

```
var txt = "";
var numbers = [45, 4, 9, 16, 25];
numbers.forEach(myFunction);

function myFunction(value) {
  txt = txt + value + "<br>";
}
```

Array.map()

This example multiplies each array value by 2 and create a new value

```
var numbers1 = [45, 4, 9, 16, 25];
var numbers2 = numbers1.map(myFunction);
function myFunction(value) {
  return value * 2;
}
```

Array.filter()

The filter() method creates a new array with array elements that passes a test.

```
var numbers = [45, 4, 9, 16, 25];
var over18 = numbers.filter(myFunction);
function myFunction(value, index, array) {
  return value > 18;
}
```

Math Object

```
Math.PI;

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

Math.pow(8, 2);     // returns 64

Math.sqrt(64);     // returns 8

……
```