# JAVA SCRIPT

## What is JavaScript?

- JavaScript is a scripting language that enables you to create dynamically updating content, control multimedia, animate images, and pretty much everything else.

The example below "finds" an HTML element (with id="demo"), and changes the element content (innerHTML) to "Hello JavaScript":

```
<!DOCTYPE html>
<html>
<body>
<h2>What Can JavaScript Do?</h2>
<p id="demo">JavaScript can change HTML content.</p>
<button type="button" onclick='document.getElementById("demo").innerHTML = "Hello JavaScript!"'>Click Me!</button>
</body>
</html>
```


# The <script> Tag

In HTML, JavaScript code is inserted between `<script>` and `</script>` tags.

```
<!DOCTYPE html>


<html>

<body>

<h2>JavaScript in Body</h2>

<p id="demo"></p>

<script>

document.getElementById("demo").innerHTML = "My First JavaScript";

</script>

</body>

</html>
```


## JavaScript Output:

JavaScript can "display" data in different ways:

- 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().

## JavaScript Statements:

```
<script>
let x, y, z;  // Statement 1
x = 5;        // Statement 2
y = 6;        // Statement 3
z = x + y;    // Statement 4
document.getElementById("demo").innerHTML =
"The value of z is " + z + ".";
</script>
```

## JavaScript Syntax:

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

## JavaScript Variables

- Automatically
- Using var
- Using let
- Using const

The var keyword was used in all JavaScript code from 1995 to 2015.

The let and const keywords were added to JavaScript in 2015.

The var keyword should only be used in code written for older browsers.

## JavaScript Comments:

Single line comments start with //.
Multi-line comments start with /* and end with */.

## Types of JavaScript Operators

There are different types of JavaScript operators:

- Arithmetic Operators
- Assignment Operators
- Comparison Operators
- String Operators
- Logical Operators
- Bitwise Operators
- Ternary Operators
- Type Operators

# JavaScript Functions:

A JavaScript function is a block of code designed to perform a particular task.

A JavaScript function is executed when "something" invokes it (calls it).

```
function myFunction(p1, p2) {
  return p1 * p2;
}
```

# JavaScript Function Syntax:

```
function name(parameter1, parameter2, parameter3) {
  // code to be executed
}
```

# Common HTML Events:

- onchange
- onclick
- onmouseover
- onmouseout
- onkeydown
- onload

# Escape Character:

| CODE | RESULT | DESCRIPTION |
| --- | --- | --- |
| \' | ' | Single Quote |
| \" | " | Double Quote |
| \\ | \ | Backslash |

# String Methods:

- String length
- String slice()
- String substring()
- String substr()
- String replace()
- String replaceAll()
- String toUpperCase()
- String toLowerCase()
- String concat()
- String trim()
- String trimStart()
- String trimEnd()
- String padStart()
- String padEnd()
- String charAt()
- String charCodeAt()
- String split()

# String Search Methods

- String indexOf()
- String lastIndexOf()
- String search()

- String match()
- String matchAll()
- String includes()
- String startsWith()
- String endsWith()

## How to Create a BigInt:

To create a BigInt, append n to the end of an integer or call BigInt()

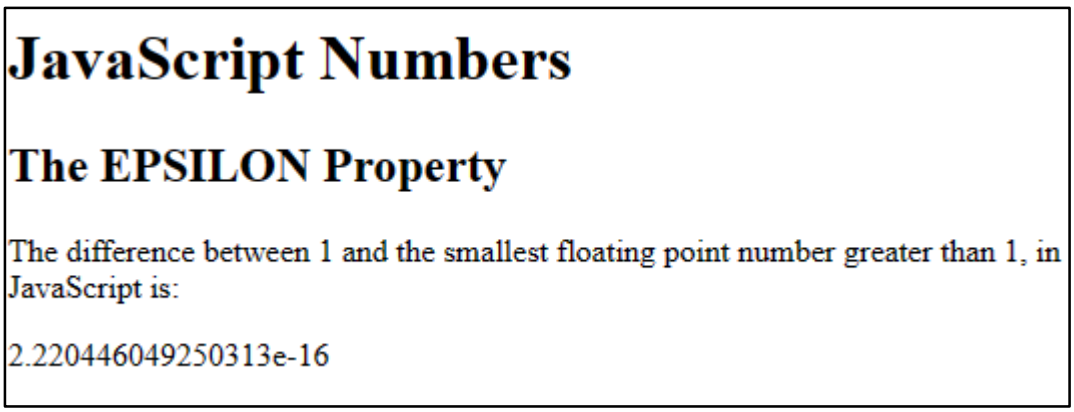
## JavaScript Number Methods:

| METHOD | DESCRIPTION |
|---|---|
| toString() | Returns a number as a string |
| toExponential() | Returns a number written in exponential notation |
| toFixed() | Returns a number written with a number of decimals |
| toPrecision() | Returns a number written with a specified length |
| ValueOf() | Returns a number as a number |

## JavaScript Number Properties:

<!DOCTYPE html>

```
<html>
<body>
<h1>JavaScript Numbers</h1>
<h2>The EPSILON Property</h2>
<p>The difference between 1 and the smallest floating point number greater than 1, in JavaScript is:</p>
<p id="demo"></p>
<script>
let x = Number.EPSILON;
document.getElementById("demo").innerHTML = x;
</script>
</body>
</html>
```




# Creating an Array

Using an array literal is the easiest way to create a JavaScript Array.

Syntax:

```
const array_name = [item1, item2, ...];
```

Example

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

Example

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

## JavaScript Array Methods

- Array length
- Array toString()
- Array pop()
- Array push()
- Array shift()
- Array unshift()
- Array join()
- Array delete()
- Array concat()
- Array flat()
- Array splice()
- Array slice()

## JavaScript Array forEach():

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

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

# JavaScript new Date()

new Date() creates a date object with the current date and time:

Example

```
const d = new Date();
```

# Math Properties (Constants)

The syntax for any Math property is : Math.property.

- Math.E
- Math.PI
- Math.SQRT2
- Math.SQRT1_2
- Math.LN2
- Math.LN10
- Math.LOG2E
- Math.LOG10E

# JavaScript Random:

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

# Conditional Statements:

```
if (hour < 18) {
  greeting = "Good day";
}
```

```
if (hour < 18) {
  greeting = "Good day";
} else {
  greeting = "Good evening";
}
```

```
if (time < 10) {
  greeting = "Good morning";
} else if (time < 20) {
  greeting = "Good day";
} else {
  greeting = "Good evening";
}
```

# Switch Statement

```
switch (new Date().getDay()) {
  case 0:
    day = "Sunday";
    break;
  case 1:
    day = "Monday";
    break;
  case 2:
     day = "Tuesday";
    break;
```

```
  case 3:
    day = "Wednesday";
    break;
  case 4:
    day = "Thursday";
    break;
  case 5:
    day = "Friday";
    break;
  case 6:
    day = "Saturday";
}
```

# For Loop

```
for (let i = 0; i < 5; i++) {
  text += "The number is " + i + "<br>";
}
```

---

```
const numbers = [45, 4, 9, 16, 25];
let txt = "";
for (let x in numbers) {
  txt += numbers[x];
}
```

---

```
const numbers = [45, 4, 9, 16, 25];

let txt = "";

numbers.forEach(myFunction);

function myFunction(value, index, array) {

  txt += value;

}
```

---

```
let language = "JavaScript";

let text = "";

for (let x of language) {

text += x;

}
```

---

# While Loop:

```
while (i < 10) {

  text += "The number is " + i;

  i++;

}
```

# JavaScript Maps:

A Map holds key-value pairs where the keys can be any datatype.

A Map remembers the original insertion order of the keys.

In JavaScript there are 5 different data types that can contain values:

- string
- number
- boolean
- object
- function

There are 6 types of objects:

- Object
- Date
- Array
- String
- Number
- Boolean

And 2 data types that cannot contain values:

- null
- undefined

# JS HTML DOM

* document.getElementById("myImage").src = "landscape.jpg";``` To modify the value of an attribute

**Use the getElementById method to find the  element, and change its text to "Hello".**

<script>

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

```
</script>
```

**Change the text of the first element that has the class name "test".**

```
<p class = "test"> </p>
<p class = "test"> </p>
<script>
  documeent.getElementsByClassName("test")[0].innerHTML = "Hello";
</script>
```

**Use HTML DOM to change the value of the image's src attribute.**

```
<script>
  document.getElementById("image").src = "pic_mountain.jpg";
</script>
```

**Use HTML DOM to change the value of the input field.**

```
<script>
  document.getElementById("myText").value = "Hello";
</script>
```

**Change the text color of the ```<p>``` element to "red".**

```
<p id="demo"></p>
```

```
<script>

  document.getElementById("demo").style.color = "red";

</script>
```

**Change the font size of the p element to 40 pixels.**

```
<p id="demo"></p>

<script>

  document.getElementById("demo").style.fontSize = "40px";

</script>
```

**Use the CSS display property to hide the p element.**

```
<p id="demo"></p>

<script>

  document.getElementById("demo").style.display = "none";

</script>
```

Use the eventListener to assign an onclick event to the ```<button>``` element.

```
<button id="demo">Click me1</button>

<script>

  document.getElementById("demo").addEventListener("click",
myFunction);

</script>
```