**JavaScript Tutorial :**

1. **Change HTML Content:**

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

1. **Change HTML Attributes:**

document.getElementById('myImage').src='pic\_bulboff.gif';

1. **Change HTML CSS:**

document.getElementById("demo").style.fontSize = "25px";

1. **Hide HTML Element:**

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

1. **Add internal javascript:**

<script>  
function myFunction() {  
   document.getElementById("demo").innerHTML = "Paragraph changed.";  
}  
</script>

1. **Add External javascript:**

<script src="myScript.js"></script>

7. **Javascript onclick Event :**

<button type="button" onclick="myFunction()">Try it</button>  
  
<script>  
function myFunction() {  
   document.getElementById("demo").innerHTML = "Paragraph changed.";  
}  
</script>

8. **Javascript Display possibilities:**

* Writing into an HTML element, using **innerHTML**.

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

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

document.write(5 + 6);

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

window.alert(5 + 6);

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

console.log(5 + 6);

10. **Javascript Variables:**

var x;  
  
x = 6;

11. **Javascript Operators:**

(5 + 6) \* 10

11. **Javascript Expressions:**

5 \* 10

12. **Javascript Keywords:**

var x, y;

13. **Javascript Comments:**

Code after double slashes **//** or between **/\*** and **\*/** is treated as a **comment**.

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

14. **Javascript Identifiers:**

Identifiers are names.

In JavaScript, identifiers are used to name variables (and keywords, and functions, and labels).

15. **Javascript is case sensitive.**

16. **Javascript and camelCase:**

**Upper Camel Case (Pascal Case):**

FirstName, LastName, MasterCard, InterCity.

**Lower Camel Case:**

JavaScript programmers tend to use camel case that starts with a lowercase letter:

firstName, lastName, masterCard, interCity.

**17.JavaScript Character Set:**

JavaScript uses the **Unicode** character set.

Unicode covers (almost) all the characters, punctuations, and symbols in the world.

**18.Semicolons ;**

When separated by semicolons, multiple statements on one line are allowed:

a = 5; b = 6; c = a + b;

**19.JavaScript White Space:**

JavaScript ignores multiple spaces. You can add white space to your script to make it more readable.

A good practice is to put spaces around operators ( = + - \* / ):

var x = y + z;

**20.JavaScript Line Length and Line Breaks:**

For best readability, programmers often like to avoid code lines longer than 80 characters.

If a JavaScript statement does not fit on one line, the best place to break it, is after an operator:

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

**21.JavaScript Keywords**

break --Terminates a switch or a loop

continue -- Jumps out of a loop and starts at the top

debugger -- Stops the execution of JavaScript, and calls (if available) the debugging function

do ... while -- Executes a block of statements, and repeats the block, while a condition is true

for -- Marks a block of statements to be executed, as long as a condition is true

function -- Declares a function

if ... else -- Marks a block of statements to be executed, depending on a condition

return --- Exits a function

switch -- Marks a block of statements to be executed, depending on different cases

try ... catch -- Implements error handling to a block of statements

var -- Declares a variable

**22.JavaScript Comments :**

**Single Line Comments :**

// Change heading:  
document.getElementById("myH").innerHTML = "My First Page";  
// Change paragraph:  
document.getElementById("myP").innerHTML = "My first paragraph.";

**Multi-line Comments**

Multi-line comments start with /\* and end with \*/

/\*  
The code below will change  
the heading with id = "myH"  
and the paragraph with id = "myP"  
in my web page:  
\*/  
document.getElementById("myH").innerHTML = "My First Page";  
document.getElementById("myP").innerHTML = "My first paragraph.";

/\*  
The code below will change  
the heading with id = "myH"  
and the paragraph with id = "myP"  
in my web page:  
\*/  
document.getElementById("myH").innerHTML = "My First Page";  
document.getElementById("myP").innerHTML = "My first paragraph.";

**JavaScript Arithmetic Operators :**

|  |  |
| --- | --- |
| **Operator** | **Description** |
| + | Addition |
| - | Subtraction |
| \* | Multiplication |
| / | Division |
| % | Modulus |
| ++ | Increment |
| -- | Decrement |

**JavaScript Assignment Operators**

Assignment operators assign values to JavaScript variables.

|  |  |  |
| --- | --- | --- |
| **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 |

### Assignment

var x = 10;  
x += 5;

**JavaScript String Operators**

The + operator can also be used to add (concatenate) strings.

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

txt1 = "What a very ";  
txt1 += "nice day";

The result of txt1 will be:

What a very nice day

**JavaScript 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 |

**JavaScript Logical Operators**

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

**JavaScript Type Operators**

|  |  |
| --- | --- |
| **Operator** | **Description** |
| Typeof | Returns the type of a variable |
| Instanceof | Returns true if an object is an instance of an object type |

**JavaScript Assignment Operators**

Assignment operators assign values to JavaScript variables.

|  |  |  |
| --- | --- | --- |
| **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 |
| >>= | 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 |

**JavaScript Data Types**

JavaScript variables can hold many **data types**: numbers, strings, objects and more:

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

**JavaScript Arrays**

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

**JavaScript Objects**

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

In HTML, the global scope is the window object. All global variables belong to the window object.

var carName = "Volvo";  
  
// code here can use window.carName

# JavaScript Events :

<https://www.w3schools.com/jsref/dom_obj_event.asp>

Special characters :

var y = "We are the so-called \"Vikings\" from the north."

**String Methods and Properties**

**String Length**

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

**Finding a String in a String**

The **indexOf()** method returns the index of (the position of) the **first** occurrence of a specified text in a string:

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

The **lastIndexOf()** method returns the index of the **last** occurrence of a specified text in a string:

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

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

Both methods accept a second parameter as the starting position for the search:

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

## Searching for a String in a String

The **search()** method searches a string for a specified value and returns the position of the match:

### Example

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

## The slice() Method

**slice()** extracts a part of a string and returns the extracted part in a new string.

The method takes 2 parameters: the starting index (position), and the ending index (position).

This example slices out a portion of a string from position 7 to position 13:

### Example

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

The result of res will be:

Banana

If a parameter is negative, the position is counted from the end of the string.

This example slices out a portion of a string from position -12 to position -6:

### Example

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

The result of res will be:

Banana

If you omit the second parameter, the method will slice out the rest of the string:

### Example

var res = str.slice(7);

## The substring() Method

**substring()** is similar to slice().

The difference is that substring() cannot accept negative indexes.

### Example

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

The result of res will be:

Banana

## The substr() Method

**substr()** is similar to slice().

The difference is that the second parameter specifies the **length** of the extracted part.

### Example

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

The result of res will be:

Banana

## Replacing String Content

The **replace()** method replaces a specified value with another value in a string:

### Example

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

By default, the replace() function replaces **only the first** match:

### Example

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

To replace all matches, use a **regular expression** with a **/g** flag (global match):

### Example

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