A JavaScript is a light weighted single threaded, cross platform, object-oriented programming language.

It used to add interactivity & dynamic effects to the html & css contents. It can be used to manipulate HTML & CSS like Button click.

For eg, showing a popup on a button click or showing page progress on scroll etc.

JS made modern web development possible.

Some modern web apps running JS: FB, Netflix, Instagram are using JS frameworks and libraries.

Most famous modern web development JS based Frontend frameworks:

* + ReactJs, Angular, EmberJs, VueJs, ReactNative, VanillaJs

Most famous modern web development JS based Backend frameworks:

* + NodeJs, ExpressJs, MeteorJs, BackboneJs

**Course Outline:**

* JS Basics
* DOM & BOM
* Events Handling
* OOPS in JS
* AJAX & JSON
* Error Handling
* jQuery
* ES5 & ES6

**Basics of Programming Language:-**

Building Blocks of Programming Language:

These are some of the building blocks of a programming language:

* Variables
* Datatypes
* Operators
* Expressions

Variables:

You can think of variables as a way to store some data for later use or in other words, think of them as containers which hold whatever data you put inside them.

Syntax: var variable\_name = value; [OR] var variable\_name;

Eg:- var num1 = 10; var mName=”John”; [OR] var num2;

Operators:

Operator are symbols used to perform some actions on variables. JavaScript supports many operators like: Plus, Minus, Multiply, Divide, Comparison etc.

Eg:-

var num1 = 2;

var num2 = 4;

var num3 = 6;

var sum = num1 + num2 + num3;

In this case, the plus operator is used add all three values from the variables.

Program Flow:

JavaScript executes the code from top to bottom. It executes one line of code at a time. If an error occurs then the next line of code is not executed.

Expressions:

Expression is a line of code which contains a valid combination of variables, values and operators.

Even statements like empty for loop, empty if conditions are also expressions & they need not generate values.

For eg, var avg = (sum) /totalNum;

To Print on the browser’s console:

console.log(avg);

console.clear(); => This clears entire console & makes empty screen as a new one.

Eg:- for strings

var firstName = “Qaifi”;

console.log(firstName); => Qaifi

Data Types:

Data Type tells JavaScript what type of value we have stored in our variables. There are many data types in JavaScript, these Number and String are also data types. JavaScript also had few other data types like Boolean, Null, Undefined, Objects etc.

Variables & its Rules:

console.clear(); => Every time the script is running, it will clear the screen/console.

var secondNum = 20;

console.log(secondName);

//Camel case ( Recommended for JS)

var secondNum;

//Pascal case

var SecondNum;

//Snake case

var second\_num;

Rules for naming JS variables

* + Name can contain letters, digits, \_ and $ Eg:- var mySubject1, maths$
  + Name cannot start with a digit Eg:- var maths;
  + Name can start with an \_ or $ Eg:- var \_subjectmarks, $rankInClass;
  + Name are case sensitive Eg:- var x, X;
  + Name cannot be reserve keywords Eg:- var varName;

Numbers & its usages:

JavaScript Numbers are always 64-bit (0 – 63) Floating Point where the **number** is stored in 52 bits (0 - 51), the **exponent** is stored in next 10 bits (52 - 62) and the last 1 bit (63) is for **sign**.

* 0-51 => numbers stores
* 52-62 => exponents stores
* 63 (last bit) => sign stores

Note: Both integers and decimal numbers are of type Number. Infinity & NaN are also type of numbers. You can check it by isFinite(val) and isNan(val);

var num1 = 10; // Integer

var num2 = 10.6; // Decimal Number

console.log(typeof(num1)); // number

console.log(typeof(num2)); // number

Practice:

console.log(0/0); // NaN

console.log(0 / 5) // 0

console.log(5 / 0) // Infinity

console.log(3 \* 'a') // NaN

console.log('a' \* 3) // NaN

console.log('b' + 4) // "b4"

console.log(4 + 'b') // "4b"

console.log(5 - 'c') // NaN

console.log('c' - 5) // NaN

// Addition

var num3 = 15;

var sum = num1 + num3;

console.log(sum);

// Substraction

var num3 = 15;

var diff = num1 - num3;

console.log(diff);

// Multiplication

var num3 = 15;

var mul = num1 \* num3;

console.log(mul);

// Division

var num3 = 15;

var div = num1 / num3;

console.log(div);

// Division by Zero => It is possible in JS

var num3 = 15;

var divByZero = num1 / 0;

console.log(divByZero); => Infinity

console.log(typeof(divByZero)); => “number”

// mulByString => It is not possible in JS

var num3 = 15;

var mulByStr = num1 \* “str”;

console.log(mulByStr); => NaN – Not an Number

console.log(typeof(mulByStr)); => “number”

Built-in Number Functions:

Eg: 1

var num1 = 10;

var num1 = 10.56;

// .toString() number as input and returns a string

console.log(num1.toString()); => “10”

console.log(num2.toString()); => “10.56”

Eg: 2

var strNum1 = “10”;

var strNum2 = “10.43”;

var strNum3 = “ABC”;

// parseInt() string numeral as input and returns a number

console.log(parseInt(strNum1)); => 10

console.log(parseInt(strNum2)); => 10

console.log(parseInt(strNum3)); => NaN

// parseFloat() string numeral as an input and returns a floating number

console.log(parseFloat(strNum1)); => 10

console.log(parseFloat(strNum2)); => 10.43

console.log(parseFloat(strNum3)); => NaN

// toFixed() takes floating number and rounds it off to given position

var strFloat = 345.677893;

console.log(strFloat.toFixed()); => ‘346’

console.log(strFloat.toFixed(2)); => ‘345.68’

console.log(strFloat.toFixed(5)); => ‘345.67789

console.log(strFloat.toFixed(10)); => ‘345.6778930000

Built-in String Functions:

var myFirstStr = “This is a string for Javascript string functions, Javascript”;

// Returns the length of the string

console.log(myFirstStr.length); => 60

//find index of a string inside another string

console.log(myFirstStr.indexOf(‘JavaScript’)); => 21

console.log(myFirstStr.indexOf(‘This’)); => 0 // Gives index of First character of the word

// Find Last Index of a string inside another string

console.log(myFirstStr.lastIndexOf(‘Javascript’)); => 50

console.log(myFirstStr.lastIndexOf(‘This’)); => 0

// Get a part of out string slice(start\_position, end\_position) **SYNTAX** => 0 to n-1 [OR] 1 to normal count

// start\_position = 0 & end\_position = n-1 exact position starts from 1 where n is normal count

console.log(myFirstStr.slice(0,5)); => “This “

console.log(myFirstStr.slice(0,4)); => “This“

console.log(myFirstStr.slice(21, 31)); => “Javascript“

// slice() also accepts negative (-) index as well which returns from end (always it starts from last index) of a string.

console.log(myFirstStr.slice(-10)); => “Javascript“

// In slice(), we can also pass only start\_value of a string as first parameter & it ends at last position of string

console.log(myFirstStr.slice(5)); => “is a string for Javascript string functions, Javascript “

// Get sub string function – substr(startPos, length) **SYNTAX**

console.log(myFirstStr.substr(0, 4)); => “This“

console.log(myFirstStr.substr(21, 10)); => “Javascript“ // starts from 21st position & extract till length of 10 from 21st position

// In substr(), it starts from given num postn and end till last length of the whole string only if 2nd param not passed.

console.log(myFirstStr.substr(21)); => “Javascript string functions, Javascript“

//Diff between slice & substring

* Slice starts from position 1 & ends at given position. But Substring starts from position 1 & ends at count of place value of length from start position (start position as 1st parameter & length as 2nd parameter)

Built-in String Functions part-2

var exampleStr = “This is Javascript tutorial”;

//toUpperCase() – It converts our string => uppercase characters

console.log(exampleStr.toUpperCase())

// Res : “THIS IS JAVASCRIPT TUTORIAL”

//toLowerCase() – It converts our string => lowercase characters

console.log(exampleStr.toLowerCase())

// Res: “this is javascript tutorial”

//concat() – It merges two or more strings

var firstName = “Javascript”;

var lastName = “Playground”;

console.log(exampleStr.concat(firstName));

// Res: “This is Javascript tutorialJavascript”

console.log(exampleStr.concat(‘ ‘, firstName, ‘ ‘, lastName));

// Res: “This is Javascript tutorial Javascript Playground”

// We can also use ‘+’ to concat two or more strings

console.log(exampleStr + ‘ ‘ + firstName + ‘ ‘ + lastName);

// Res: “This is Javascript tutorial Javascript Playground”

// trim() – Its removes extra spaces

var extraSpaceString = ‘ mystring ‘;

console.log(extraSpaceString.trim()); // Res: “mystring”

var extraSpaceStringExample2 = ‘ my string ‘;

console.log(extraSpaceStringExample2.trim()); // Res: “my string”

// charAt() – This take a position as an arg and returns the character at that position

var charAtExampleString = ‘This is my test string’;

console.log(charAtExampleString.charAt(5)); // Res: “I”

console.log(charAtExampleString.charAt(10)); // Res: “ “

console.log(charAtExampleString.charAt(11)); // Res: “t”

// split() – It splits our string on the basis of the arguments passed

var sampleString = “This is my sample string”;

console.log(sampleString.split(‘ ‘)); // Res: [“This”, “is”, “my”, “sample”, “string”]

// space between the words breaks our string into array of multiple strings elements

var sampleString2 = “This,is,my,sample,string”;

console.log(sampleString2.split(‘,’)); // Res: [“This”, “is”, “my”, “sample”, “string”]

var sampleString3 = “This is awesome”;

console.log(sampleString3.split()); // Res: [“This is awesome”]

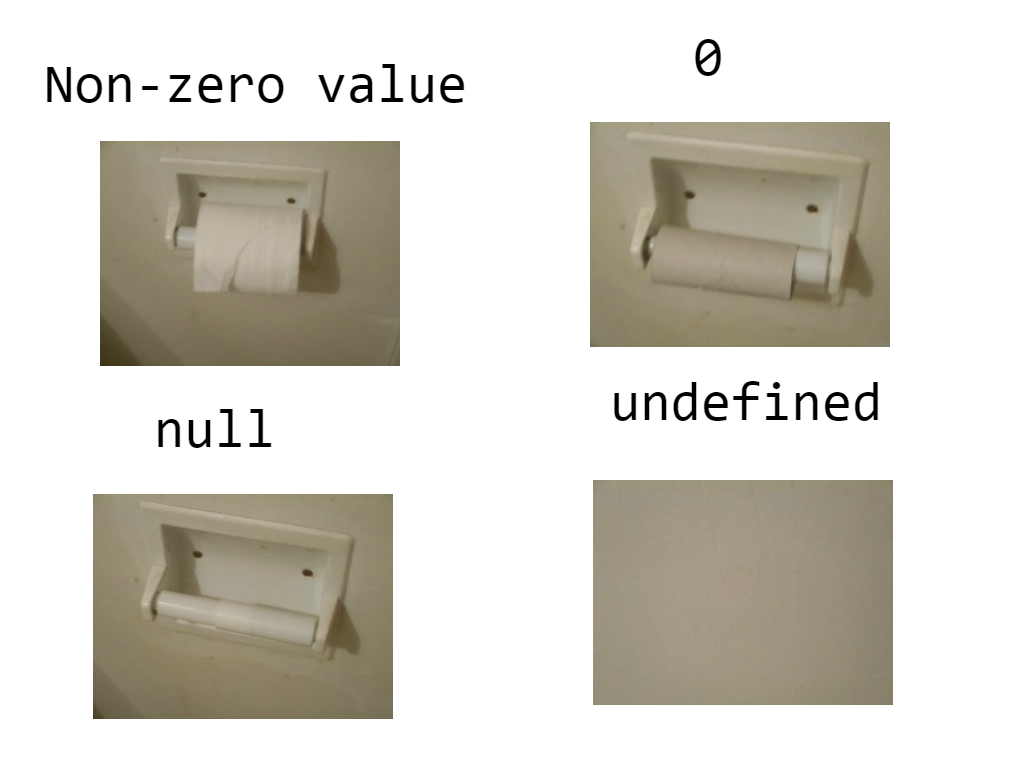
Null & Undefined

null:- Whereas null is a value assigned to a variable may be just represent that the variable holds no value when a variable will hold null. Could be assigned. Typeof null is object. No value, on purpose.

Eg:- var nVar = null ; // nVar holds the value null & null represents there is no value

console.log(nVar); // Res: null console.log(typeof(nVar)); // Res: object

// null is a value stored in ‘nVar’ variable | undefined == null // false | the bowl is empty



undefined:- When a variable will hold undefined means when a variable has been declared but has not been initialized or in another words not yet assigned values to that declared variable.

Has not been assigned. Declared, but not yet defined. Equal to NaN (Not A Number). Even the bowl is not exist.

Eg:- var myVar ; // myVar holds no value

console.log(myVar); // Res: undefined

// This tells us that there is no value in ‘myVar’ variable.

console.log(typeof(myVar)); // Res: undefined

undefined == null // true

Javascript – Conditional Statement: If-else

if(7>5){

console.log(true); // Res: true

}

if(5>7){

console.log(true);

}

else{

console.log(false);

} // Res: false

if{5>7){

console.log(‘5>7’);

} else if(7>8){

console.log(‘7>8’);

} else{

console.log(‘Nothing’);

} // Res: “Nothing”

Arithmetic Operators

These are the arithmetic operators:

* +, // Addition
* -, // Subtraction
* \*, // Multiplication
* /, // Division
* %, // Modulus
* a++, ++a// Post and Pre Increment (a is variable)
* a--, --a // Post and Pre Decrement

Eg:- var num1= 10;

var num2= 15;

var num3= 20;

// Addition

console.log(num1+num2); // Res: 25

console.log(num1+num2 + num3); // Res:45

// Substraction

console.log(num1-num2); // Res: 5

console.log(num3-num2 – num1); // Res: -5

// Multiplication

console.log(num1\*num2); // Res: 150

// Division

console.log(num2/num1); // Res: 1.5

// Modulus

console.log(num2 % num1); // Res: 5

// Increment – Post and Pre

Post Increment (a++): Post the variable value first and then Increments the value for next iteration.

Eg1:- console.log(num1++); // Res: 10

// It printed the current value of num1 and increments it & kept it safely.

console.log(num1); // Res: 11

// Next time it prints incremented value.

Eg2:- console.log(num1++); // Res:10

console.log(num1++); // Res: 11

console.log(num1); // Res: 12

Pre Increment (++a): Increments the variable value first and then Post that incremented value in same iteration/time.

console.log(++num1); // Res: 11

// Here it updates/increments the value +1 first and prints it while first time itself.

console.log(num1); // Res: 11

// Decrement – Post and Pre

Post Decrement (a--): Post the variable value first and then decrements the value for next iteration/time.

Eg:- console.log(num1--); // Res: 10

console.log(num1); // Res: 9

Pre Decrement (--a): Decrements the variable value first and then Post that decremented value in same iteration/time.

console.log(--num1); // Res: 9

console.log( num1); // Res: 9

JavaScript – Conditional Statement: Switch

Imagine if u want 10 conditions, then we have to write 10 if-else statements. That’s where switch comes to rescue.

var currentDay = ‘Mon’;

if(currentDay === ‘Mon’){

console.log(‘Timings: 10:00-06:00’)

} else if(currentDay == ‘Tue’){

console.log(‘Timings: 09:00-05:00’)

} else if(currentDay == ‘Wed’){

console.log(‘Timings: 09:30-05:30’)

} else if(currentDay == ‘Thu’){

console.log(‘Timings: 10:15-06:15’)

} else if(currentDay == ‘Fri’){

console.log(‘Timings: 09:05-05:05’)

} else if(currentDay == ‘Sat’){

console.log(‘Timings: 09:00-05:00’)

} else if(currentDay == ‘Sun’){

console.log(‘Timings: 09:00-01:00’)

}

// Res: “Timings: 10:00-06:00”

var currentDay = ‘Thu’;

switch(currentDay){

case ‘Mon’:

console.log(‘Timings: 10:00-06:00’)

break;

case ‘Tue’:

console.log(‘Timings: 09:00-05:00’)

break;

case ‘Wed’:

console.log(‘Timings: 09:30-05:30’)

break;

case ‘Thu’:

console.log(‘Timings: 10:15-06:15’)

break;

case ‘Fri’:

console.log(‘Timings: 09:05-05:05’)

break;

case ‘Sat’:

console.log(‘Timings: 09:00-05:00’)

break;

case ‘Sun’:

console.log(‘Timings: 09:00-01:00’)

break;

default:

console.log(‘Give correct day’);

}

// Res: “Timings: 10:15-06:15”

Assignment Operators: (=, +=, -=, \*=, /=, %=)

var num1 = 15;

var num2 = 30;

// =

num1 = 20;

// +=

[num1 += 10; // num1 = num1 + 10

console.log(num1); => Res: 25]

num1 += 10; // num1 = 15 + 10 = 25

num1 += num2; //num1 = 25 + 30

console.log(num1); => Res: 55

// -=

[num1 -= 10; // num1 = num1 - 10

console.log(num1); => Res: 5]

num1 -= 10; // num1 = 15 - 10 = 5

num1 -= num2; //num1 = 5 - 30

console.log(num1); => Res: -25

// \*=

[num1 \*= 20; // num1 = num1 \* 20

console.log(num1); => Res: 300]

// ‘/=’

[num1 /= 10; // num1 = num1 / 10

console.log(num1); => Res: 1.5]

// %=

[num1 %= 10; // num1 = num1 % 10

console.log(num1); => Res: 5]

Comparison and Logical Operators:

// Using == we can only compare both values

if(5 == ‘5’) {console.log(true)}; // Res: true

// Using === we can only compare both values and datatype

if(5 === ‘5’) {console.log(true)}; // Res: false

**// Here both operands data type are different left: number & right: string. Hence false.**

**if(5 != 4) {console.log(true) } else {console.log(false)} // Res: false**

**if(5 !== 5) {console.log(true) } else {console.log(false)} // Res: false**

**if(5 != ‘5’) {console.log(true) } else {console.log(false)} // Res: false**

**if(5 !== ‘5’) {console.log(true) } else {console.log(false)} // Res: true**

**if(5 < 10) {console.log(true) } else {console.log(false)} // Res: true**

**if(15 < 10) {console.log(true) } else {console.log(false)} // Res: false**

**if(15 < 15) {console.log(true) } else {console.log(false)} // Res: false**

**if(15 <= 15) {console.log(true) } else {console.log(false)} // Res: true**

**if(15 >=15) {console.log(true) } else {console.log(false)} // Res: true**

**if(15 > 15) {console.log(true) } else {console.log(false)} // Res: false**

**// Logical Operators &&, ||, !**

**// && AND**

**if(5<10 && 10<15) { console.log(true)} else {console.log(false)} // Res: true**

**if(5<10 && 10>15) { console.log(true)} else {console.log(false)} // Res: false**

**if(5>10 && 10<15) { console.log(true)} else {console.log(false)} // Res: false**

**if(5>10 && 10>15) { console.log(true)} else {console.log(false)} // Res: false**

**// || OR**

**console.log(5<10 || 10<15) // Res: true**

**console.log(!(5<10)) // Res: false**

**// Ternary Operator (expression) ? true statement: false statement**

**2 < 3 ? console.log(true) : console.log(false) // Res: true**

**20 < 3 ? console.log(true) : console.log(false) // Res: false**

Implicit Type Coercion:

Type coercion is conversion of one datatype to another datatype.

1. Implicit means automatically converts by JavaScript
2. Explicit means programmatically converts by developers (us)

Eg 1 :- var mStr = ‘Hello’;

var mNum = 5;

var mSum = mStr + mNum;

console.log(mSum); // Res: “Hello5”

console.log(typeof(mSum)); // Res: “string”

Eg 2:- console.log(‘hola’ + true); // Res: “holatrue”

console.log(typeof(‘hola’ + true)); // Res: “string”

Eg 3:- console.log(null + ‘hola’); // Res: “nullhola”

console.log(typeof(null + ‘hola’)); // Res: “string”

Eg 4:- console.log(50 + ‘hola’); // Res: “50hola”

console.log(typeof(50 + ‘hola’)); // Res: “string”

Eg 5:- var mValue = - ‘2’

console.log(mValue); // Res: -2

console.log(typeof(mValue)); // Res: “number”

Eg 6:- var mValue = + ‘2’

console.log(mValue); // Res: 2

console.log(typeof(mValue)); // Res: “number”

Eg 7:- var mValue = - ‘apple’

console.log(mValue); // Res: NaN

console.log(typeof(mValue)); // Res: “number”

Eg 5:- var mValue = 2>=‘2’

console.log(mValue); // Res: true

console.log(typeof(mValue)); // Res: “boolean”

// boolean

if(true){

console.log(‘True’);

} else {

Console.log(‘False’)

} // Res: “True”

// string

if(‘Apple’){

console.log(‘True’);

} else {

Console.log(‘False’)

} // Res: “True”

// zero means false; one means true

if(0){

console.log(‘True’);

} else {

Console.log(‘False’)

} // Res: “False”

// null

if(null){

console.log(‘True’);

} else {

Console.log(‘False’)

} // Res: “False”

// undefined

if(undefined){

console.log(‘True’);

} else {

Console.log(‘False’)

} // Res: “False”

// empty value

if(‘ ‘){

console.log(‘True’);

} else {

Console.log(‘False’)

} // Res: “False”

Practice Resources (Links):-

1. Intro to JS:

<https://careerfoundry.com/en/tutorials/web-development-for-beginners/an-introduction-to-javascript/>

1. JS Tutorial:

<https://javascript.info/>

1. Strings in JS:

<https://www.kirupa.com/html5/strings_in_javascript.htm>

1. Working with Strings:

<https://www.oreilly.com/library/view/javascript-cookbook/9781449390211/ch01.html>

1. Operators:

<https://www.tutorialrepublic.com/javascript-tutorial/javascript-operators.php>

1. Type of coercion:

<https://www.freecodecamp.org/news/js-type-coercion-explained-27ba3d9a2839/>

1. Objects:

<https://www.freecodecamp.org/news/objects-in-javascript-for-beginners/>

1. Understanding JS Objects:

<https://www.javascript.com/learn/objects>

1. JS Practice challenge:

<https://edabit.com/challenges/javascript>

1. Online Quiz:

<https://www.tutorialspoint.com/javascript/javascript_online_quiz.htm>

1. JS Basic – Exercise, Practice

<https://www.w3resource.com/javascript-exercises/javascript-basic-exercises.php>

1. Programming Example

<https://codescracker.com/js/program/index.htm>

1. Arrays Intro

<https://www.w3schools.com/js/js_arrays.asp>

1. Array methods

<https://www.w3schools.com/js/js_array_methods.asp>

1. Array const

<https://www.w3schools.com/js/js_array_const.asp>

1. [**Learn FREE Spoken English Easy and Quick on YouTube! SUBSCRIBE NOW!**](https://www.youtube.com/watch?v=zR7nFRIcKa0)

<https://www.youtube.com/watch?v=zR7nFRIcKa0>

1. Loops

<https://www.javatpoint.com/javascript-loop>

1. How do you find the missing number in a given integer array of 1 to 100?
2. How do you find the duplicate number on a given integer array?
3. How do you find the largest and smallest number in an unsorted integer array?
4. How do you find all pairs of an integer array whose sum is equal to a given number?
5. How do you find duplicate numbers in an array if it contains multiple duplicates?
6. How do you reverse an array in place in JavaScript? In place means you cannot create a new array. You have to update the original array.
7. How do you print duplicate characters from a string?
8. How do you check if two strings are anagrams of each other?
9. How do you print the first non-repeated character from a string?
10. How can a given string be reversed using recursion?
11. How do you check if a string contains only digits?
12. How are duplicate characters found in a string?
13. How do you count a number of vowels and consonants in a given string?
14. How do you count the occurrence of a given character in a string?
15. How do you find all permutations of a string?
16. How do you reverse words in a given sentence without using any in-built method?
17. How do you check if two strings are a rotation of each other?
18. How do you check if a given string is a palindrome?
19. For loop, of loop

<https://www.digitalocean.com/community/tutorials/for-loops-for-of-loops-and-for-in-loops-in-javascript>

1. Functions

<https://javascript.plainenglish.io/introduction-to-javascript-functions-a0a1687f2318>

1. Define & call a function

<https://learncodeweb.com/javascript/define-and-call-a-function-in-javascript-with-example/>

1. Questions

<https://www.programiz.com/javascript/examples>

1. Quiz

<https://www.javatpoint.com/javascript-quiz>

1. Jumpstart your javascript application development

<https://kandi.openweaver.com/?landingpage=javascript_all_projects&landingpage=javascript&utm_campaign=paid_search_productcapability_javascript&utm_medium=cpc&utm_source=google&utm_term=javascript_all_project&utm_content=code&gclid=Cj0KCQjw0oyYBhDGARIsAMZEuMuvCISOUwDa9pQnGzEuFKImub-ALGtegayso29EqSrbsm3ewiTwTZwaAhrtEALw_wcB>

1. DOM -Documentation Object Model

<https://www.geeksforgeeks.org/dom-document-object-model/>

1. Basics of DOM

<https://www.tutorialride.com/javascript/javascript-document-object-model.htm>

26. Collection of DOM methods / Functions

<https://www.w3schools.com/js/js_htmldom_document.asp>

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

27. Events

<https://www.javatpoint.com/javascript-events>

1. Handling Events in javascript

<https://www.javascripttutorial.net/javascript-dom/handling-events-in-javascript/>