Set Timeout:

The set timeout method calls a function after a number of milliseconds.

1 second = 1000 milliseconds.

| syntax: // setTimeout(callbackFunction, milliSecond); setTimeout(() => {  // Code to delay }, 1000); |
| --- |

Clear Timeout:

The clear timeout method is used to clear the time set by setTimeout

| syntax: var setTimeoutID = setTimeout(() => {  // Code to delay }, 2000); //To stop the setTimeout clearTimeout(setTimeoutID); |
| --- |

Set Interval:

The set interval methods calls a function at a specific time intervals (milliseconds)

| syntax: //setInterval(callbackFunction, milliSecond); setInterval(() => {  // code to call at every two second }, 2000); |
| --- |

Clear Interval:

The clear interval method is used to clear the time set by setInterval

| syntax: var setIntervalID = setInterval(() => {  // code to call at every two second }, 2000); //To stop the setInterval clearInterval(setIntervalID); |
| --- |

Local Storage:

The local storage allows storing key / value pairs in the browser. The data stored in local storage has no expiration date.

Maximum limit of local storage is 5MB.

| syntax: // SAVE data to localStorage localStorage.setItem(key, value); // key - string format, value - string format  //READ data from localStorage var value = localStorage.getItem(key);  // REMOVE data from localStorage localStorage.removeItem(key);  // CLEAR localStorage localStorage.clear(); |
| --- |

Session Storage:

The session storage allows storing key / value pairs in the browser. The data stored in session storage will clear, once the browser tab or the browser is closed.

| syntax: // SAVE data to sessionStorage sessionStorage.setItem(key, value); // key - string format, value - string format  //READ data from sessionStorage var value = sessionStorage.getItem(key);  // REMOVE data from sessionStorage sessionStorage.removeItem(key);  // CLEAR sessionStorage sessionStorage.clear(); |
| --- |

JSON:

JSON stands for JavaScript Object Notation. It is a format for storing and transporting data.

| {  "employees":[  {"firstName":"John", "lastName":"Doe"},  {"firstName":"Anna", "lastName":"Smith"},  {"firstName":"Peter", "lastName":"Jones"}  ] } |
| --- |

To Convert Object/Array to String:

| var student = {name: 'Karthick', age: 28}; var stringObject = JSON.stringify(student);  //'{"name":"Karthick","age":28}' |
| --- |

To Convert String to Object/Array:

| // '{"name":"Karthick","age":28}' var finalResult = JSON.parse(object); // {name: 'Karthick', age: 28} |
| --- |

Scope:

Scope determines the accessibility (visibility) of the variables. It has 3 types of scope.

1. Block Scope (let - only block scope)
2. Function/Local Scope
3. Global Scope

Hoisting:

Hoisting is a default JavaScript behavior of moving declarations to the top.

Error Handling:

1. Try : It defines the code block to run (we will write all our code inside try block)
2. Catch : It defines the code block to show error (It will return the error)

| function validate(){  try{  // code block  }  catch(error){  // display the error message  } } |
| --- |

Array:

To Create an empty array

1. Using Array Syntax [ ]

| var variableName = [ ]; |
| --- |

1. Using Array Method

| var variableName = new Array( ); |
| --- |

To Insert value into Array

1. Using Push Method → Add value at last of the Array list

| arrayVariableName.push(value\_1, value\_2, .... , value\_n); |
| --- |

1. Using UnShift Method → Add value at first of the Array list

| arrayVariableName.unshift(value\_1, value\_2, .... , value\_n); |
| --- |

1. Using Splice Method → Add value at a particular index of the Array list

| var addItemIndex = 1; //number var deleteCount = 2; //number var itemValue = ['name', 'milk']; //any value csv.splice(addItemIndex, deleteCount, itemValue); |
| --- |

To Remove value from Array

1. Using Pop Method → Removes the last element of an array

| arrayVariableName.pop(); |
| --- |

1. Using Shift Method → Removes the first element of an array

| arrayVariableName.shift(); |
| --- |

1. Using Splice Method → Removes the particular element of an array

| var removeItemIndex = 1; //number var deleteCount = 1; //number arrayVariableName.splice(removeItemIndex, deleteCount); |
| --- |

To Find Index in Array

1. Using indexOf Method → Used to find index value from list of String and Number

| arrayVariableName.indexOf(value) |
| --- |

1. Using findIndex → Used to find index value from list of Object, Array, String, Number

| arrayVariableName.findIndex((value) => return value.key === "value"); |
| --- |

To Iterate (Loop) an Array

1. forEach - forEach don't return anything. It just runs the callback function for each element of the array.

| array\_variable\_name.forEach((value, index) => {   console.log(value, index);  // Code inside will run for Number of element count in Array }); |
| --- |

1. Map - It returns new array by executing the callback function for each elements of the array

| array\_variable\_name.map((value, index) => {   console.log(value, index);  // Code inside will run for Number of element count in Array. And it will return a new Array  return data; }); |
| --- |

1. Filter - If the condition is true for an element, element is picked for the return array

| array\_variable\_name.filter((value, index) => {   console.log(value, index);  // Code inside will run for Number of element count in Array. And it will return a new Array  return condition;  }); |
| --- |

String:

In javascript, strings are used to Storing and Manipulating Text values.

To Create a String:

1. Using String Literals

| var variableName = " Text Value "; |
| --- |

1. Using String Constructor Method

| var variableName = new String(" Text Value "); |
| --- |

To Find Length of a String (Character Count):

It will give a character count value in number.

| stringVariableName.length // it will give number value |
| --- |

To Extract String Parts:

There are 3 methods in extracting a part of string

1. Slice Method → To extract a part of a string and return the extracted part in a new string. End count will always minus 1

| stringVariableName.splice(start, end) // end - 1 |
| --- |

1. SubString → It is similar to the Slice Method, Only difference is it cannot accept negative values.

| stringVariableName.substring(start, end) // end - 1 |
| --- |

1. SubStr → It is similar to the Slice Method, but it will take the length of the string.

| stringVariableName.substr(start, length) |
| --- |

Replacing String Content:

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

| stringVariableName.replace("Kumar", "Moch") |
| --- |

Convert Text to UpperCase and LowerCase

1. To convert a text into Upper Case

| stringVariableName.toUpperCase( ); //Upper case letter output |
| --- |

1. To convert a text into Lower Case

| stringVariableName.toLowerCase( ); //Lower case letter output |
| --- |

Combine Two or more Text:

1. Concat Method → It is used to join two or more string together

| stringVariableName.concat(stringVariableName2); |
| --- |

1. Plus (+) Operator

| stringVariableName + stringVariableName2; |
| --- |

1. Template String (ES6 Feature) → To use back-tick symbol

| `${stringVariableName1} ${stringVariableName2}` |
| --- |

Trim Method:

The Trim method is used to remove space from Beginning and End of a Text

| stringVariableName.trim(); |
| --- |

Search a Text value:

1. IndexOf Method → The indexOf() method returns the index of (the position of) the first occurrence of a specified text in a string

| stringVariableName.indexOf("Search Text") // output -> Positive index value |
| --- |

1. Last IndexOf Method → The lastIndexOf() method returns the index of the last occurrence of a specified text in a string

| stringVariableName.lastIndexOf("Search Text") |
| --- |

Number:

1. To String Method → It will return a number into String value.

| numberVariable.toString() // output → "100" |
| --- |

1. To Fixed Method → It will return a number with specified decimal points

| numberVariable.toFixed(decimalPosition) // decimalPosition → number |
| --- |

1. Parse Int Method → It will return a String into Whole number

| parseInt(StringNumber) // output → whole number |
| --- |

1. Parse Float Method → It will return a String into Decimal number

| parseFloat(StringNumber) // output → Decimal number |
| --- |

1. IsNaN Method → Is Not a Number → It will verify whether the given value is a number or not

| isNaN(value) // value → number = false, number → string, obj = true |
| --- |

Object:

It is a collection of Property with Key and Value pair.

1. Create a New Object

| var variable\_name = {  key1 : value,  key2 : value } |
| --- |

1. To Read values from an Object

| object\_variable\_name.key // way 1 → Static approach object\_variable\_name["key"] //way 2 → Dynamic approach |
| --- |

1. To Add new values in a Object

| object\_variable\_name.key = value // way 1 → Static approach object\_variable\_name["key"] = value //way 2 → Dynamic approach |
| --- |

1. To delete a value from Object

| delete object\_variable\_name.key; // way 1 → Static approach delete object\_variable\_name["key"]; //way 2 → Dynamic approach |
| --- |

Object Methods:

1. Has Own Property → The **hasOwnProperty()** method returns a boolean indicating whether the object has the specified property as its own property

| object\_variable\_name.hasOwnProperty("key") |
| --- |

1. To Copies Object → It copies all key/values from one or more *source objects* to a *target object*.

| var variable\_name = Object.assign({ }, source\_object); var variable\_name = {...source\_object}; //spread operator (ES6) |
| --- |

1. To Freeze an Object → A frozen object can no longer be changed; freezing an object prevents new properties from being added to it, existing properties from being removed.

| Object.freeze(object\_variable\_name) |
| --- |

1. To check whether an Object is Freeze

| Object.isFrozen(object\_variable\_name) // It will return a boolean |
| --- |

1. To Seal an Object → It will prevent new properties from being added to it.

| Object.seal(object\_variable\_name) |
| --- |

1. To check whether an Object is Seal

| Object.isSealed(object\_variable\_name) // It will return a boolean |
| --- |

Function:

It executes set of statement in block, to perform a particular task

| function function\_name1(argument1, argument2, … , argumentN){  //block of code }  function function\_name2(){  function\_name1(); }  run: function\_name(parameter1, parameter2, … , parameterN); |
| --- |

Class:

class are template for creating objects

Method:

Same as function

| class classname { method1(){ } method2(){ this.method1(); } method3(){ } } |
| --- |

ES6 Features:

Default Function Parameter - To set the default value for parameters of a function.

| function say(message ='Hi') {  console.log(message); }  say(); // 'Hi' say('Hello') // 'Hello' |
| --- |

Rest Parameter

| function fn(a,b,...args) {  //... } |
| --- |

Spread Operator

| var array\_variable\_name = [ ... old\_array\_variable]; var object\_variable\_name = { ... old\_object\_variable }; |
| --- |

Object Literal

| var first\_name = "Milk"; var last\_name = "Moch"; var obj = {  first\_name,  last\_name } |
| --- |

For of - It is an alternative way to Iterate an Array

| for(var value of studentList){  console.log(value) } |
| --- |