**Typescript:-**

●Let --  Execute inside a loop, it does not exist outside of the loop

●const - Read-only, you cannot reassign new values

●Tuple - Arrays with a fixed number of elements - [number, boolean, string]

●Enum - giving names to numeric values, strings - assign values enum

Foods {'bacon'= 18, 'tomato', 'lettuce'};

●for...of  -- objects such as arrays, strings

●for...in   -- loops through the properties of an object

●Template Literals -- back-tick (` `) (grave accent)  -- `The sum of ${a} `

●Arrow Functions --  var sum = (a, b) => a + b;

●Function Expression -- var sum = function(a, b) { return a + b; }

●this  -- current execution context of a function

An interface -- is defined with the keyword interface and it can include properties and method declarations using a function or an [arrow function](https://www.tutorialsteacher.com/typescript/arrow-function).

An interface can also define the type of an array where you can define the type of index as well as values.

●class  -- Class have Class constructor and Class method

●Extends -- Child class inherits from Parent

●super() -- Call parent's constructor

●constructor() -- creating and initializing and class prototype

●Module -- export or import variables, functions, classes

●Rest Parameters - is specified by prefixing a named parameter with rest operator (...)

●function myFunction(a, b, ...args) { return args; }

●Spread operator - performs the exact opposite function of the rest operator

●function addNumbers(a, b, c) { return a + b + c; } alert(addNumbers(...numbers))

interface IEmployee {

empCode: number;

name: string;

getSalary:(empCode: number) => number;

}

class Employee implements IEmployee {

empCode: number;

name: string;

constructor(code: number, name: string) {

this.empCode = code;

this.name = name;

}

getSalary(empCode:number):number {

return 20000;

}

}

let emp = new Employee(1, "Steve");

**Javascript Hints (Brendan Eich):-**

DOM - Document change, edit, delete, create, copy.

Object- group of data, organizing data, a complex variable,  with properties & methods. Literal type, keyword new type.{ }

Property - Characteristic of obj, name, value

Methods/Functions - Overload, Return type, void type, with & without parameter, action's of object, collect of code. Execute with function. Obj.fn();

Overload- single object multiple behaviors

Parameters - pass the values in fn

Arguments - invoked values in fn

Prototype - association with object, fn array.push(obj), fn.prototyoe = new fn();

Class - Generate object's. ( )

Constructor - object generate process, fn creates the instance prototypes.

Array- list data, stock house, hold multiple items.[ single, multiple, associative]

Getter setter - Read, write

Closures - fn inside the fn

Inheritance - prototype language

Encapsulation - storing data, hide/put data

Polymorphism - Architect rules( extend, implement, export, import)

Abstraction - Names architecture

Model - objects

View - user sees

Controller - manage Model & View

Events - Tightly, Loosely couple, control the structure

Target - children

Current target - parent, this, main div

Primitive data types = Non prototypes

- Boolen, number, string, undefined, null, true, false, integer, unit.

Non Primitive data types = Prototypes

Objects, Arrays

Array To Object -->

    // Using Object.assign() - Object.assign({}, ['a', 'b', 'c']).

    // Using the spread syntax - {...['a', 'b', 'c']}.

    // Using the Array.reduce(()=>{ }, { }) method.

    // Using the Object.fromEntries() method.

    // Using the Array.forEach((e,i)=>{ })

string to array -->

  // str.split,

// str.match(/[^.!?]+[.!?]+/g)

// Object.assign([], str);

**Array Methods -->**

concat()Joins arrays and returns an array with the joined arrays

constructorReturns the function that created the Array object's prototype

copyWithin()Copies array elements within the array, to and from specified positions

entries()Returns a key/value pair Array Iteration Object

every()Checks if every element in an array pass a test

fill()Fill the elements in an array with a static value

filter()Creates a new array with every element in an array that pass a test

find()Returns the value of the first element in an array that pass a test

findIndex()Returns the index of the first element in an array that pass a test

forEach()Calls a function for each array element

from()Creates an array from an object

includes()Check if an array contains the specified element

indexOf()Search the array for an element and returns its position

isArray()Checks whether an object is an array

join()Joins all elements of an array into a string

keys()Returns a Array Iteration Object, containing the keys of the original array

lastIndexOf()Search the array for an element, starting at the end, and returns its position

lengthSets or returns the number of elements in an array

map()Creates a new array with the result of calling a function for each array element

pop()Removes the last element of an array, and returns that element

prototype Allows you to add properties and methods to an Array object

push()Adds new elements to the end of an array, and returns the new length

reduce()Reduce the values of an array to a single value (going left-to-right)

reduceRight()Reduce the values of an array to a single value (going right-to-left)

reverse()Reverses the order of the elements in an array

shift()Removes the first element of an array, and returns that element

slice()Selects a part of an array, and returns the new array

some()Checks if any of the elements in an array pass a test

sort()Sorts the elements of an array

splice()Adds/Removes elements from an array

toString()Converts an array to a string, and returns the result

unshift()Adds new elements to the beginning of an array, and returns the new length

valueOf()Returns the primitive value of an array

**Class Methods -->**

constructor()A special method for creating and initializing objects created within a class

extends Extends a class (inherit)

static -Defines a static method for a class

super -Refers to the parent class

**Object Methods and Properties -->**

constructorReturns the function that created an object's prototype

keys()Returns an Array Iterator object with the keys of an object

prototypeLet you to add properties and methods to JavaScript objects

toString()Converts an object to a string and returns the result

valueOf()Returns the primitive value of an object

**Boolean Methods and Properties  -->**

constructorReturns the function that created JavaScript's Boolean prototype

prototypeAllows you to add properties and methods to the Boolean prototype

toString()Converts a boolean value to a string, and returns the result

valueOf()Returns the primitive value of a boolean

**String Methods   -->**

charAt()Returns the character at a specified index (position)

charCodeAt()Returns the Unicode of the character at a specified index

concat()Returns two or more joined strings

constructorReturns the string's constructor function

endsWith()Returns if a string ends with a specified value

fromCharCode()Returns Unicode values as characters

includes()Returns if a string contains a specified value

indexOf()Returns the index (position) of the first occurrence of a value in a string

lastIndexOf()Returns the index (position) of the last occurrence of a value in a string

lengthReturns the length of a string

localeCompare()Compares two strings in the current locale

match()Searches a string for a value, or a regular expression, and returns the matches

prototypeAllows you to add properties and methods to an object

repeat()Returns a new string with a number of copies of a string

replace()Searches a string for a value, or a regular expression, and returns a string where the values are replaced

search()Searches a string for a value, or regular expression, and returns the index (position) of the match

slice()Extracts a part of a string and returns a new string

split()Splits a string into an array of substrings

startsWith()Checks whether a string begins with specified characters

substr()Extracts a number of characters from a string, from a start index (position)

substring()Extracts characters from a string, between two specified indices (positions)

toLocaleLowerCase()Returns a string converted to lowercase letters, using the host's locale

toLocaleUpperCase()Returns a string converted to uppercase letters, using the host's locale

toLowerCase()Returns a string converted to lowercase letters

toString()Returns a string or a string object as a string

toUpperCase()Returns a string converted to uppercase letters

trim()Returns a string with removed whitespaces

trimEnd()Returns a string with removed whitespaces from the end

trimStart()Returns a string with removed whitespaces fropm the start

valueOf()Returns the primitive value of a string or a string object

**Statement Identifiers  -->**

Break Exits a switch or a loop

class Declares a class

const Declares a variable with a constant value

continue Breaks one iteration (in the loop) if a specified condition occurs, and continues with the next iteration in the loop

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 Loops through a block of code a number of times

for ... in Loops through the properties of an object

for ... of Loops through the values of an iterable object

function Declares a function

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

let Declares a variable

return Stops the execution of a function and returns a value from that function

switch arks a block of statements to be executed depending on different cases

throwh rows (generates) an error

try ... catch ... finally Marks the block of statements to be executed when an error occurs in a try block, and implements error handling

var Declares a variable

while Marks a block of statements to be executed while a condition is true