**27.09.2022**

**Installation**

**Download node js for windows from web**

Node js :- provide environment to run the java script code which is written visual studio code

Download visual code from browser and it provide environment to write the java script code

**Run the code in visual code**

Step1. create the folder and type cmd in the file path

Step2: create folder inside the visual studio and add or dump .js files

Step3: Inorder to run the .js file open terminal and create a new terminal and give command as **node absolute path of .js file**  and click enter ( Make sure auto save option is enabled to save automatically)

**Java Script data types**

Java script uses data types like primitive data type and non primitive data type

Primitive data type:

They are

**28.09.2022**

**Interview Question 1: why variables are loosely typed in JS?**

In java Script variables are loosely typed because unlike in java we don’t define the data type of the variable

Eg: var a = 10

var b = "helo"

Here only variables are declared and initialized but not pointed to any of the data type so it is called as loosely typed

**Interview Question 2: Explicitly converting the data types of the variable?**

Most commonly used and allowed conversions are string 🡪 number and number 🡪 to string

**String 🡪 Number**

var b = "helo" // string

console.log(typeof(b)); // string

console.log(typeof(Number(b))); // number

**Number 🡪 String**

var a = 10 // number

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

console.log(typeof(String(a))); // String

**Interview Question 3: Difference between == and ===**

== compares only the values but not the datatype

=== compares the values an also the datatype of the varaiables

**FOR ==**

var a = 10;

var b = '10'

if (a==b) {

    console.log('both are equal');

}

else{

    console.log('both are not equal');

}

**OUTPUT:**

**both are equal**

**FOR ===**

var a = 10;

var b = '10'

if (a===b) {

    console.log('both are equal');

}

else{

    console.log('both are not equal');

}

**OUTPUT**

**Both are not equal**

var a = '10';

var b = '10'

if (a===b) {

    console.log('both are equal');

}

else{

    console.log('both are not equal');

}

**OUTPUT**

**Both are equal**

**Interview Question 4: Pre Incremental and post incremental**

In pre incremental the values are incremented and then assigned to the variable

In post incremental the values are assigned first and then incremented.

**Interview Question 5: different types of variable:- var, let, const.**

**Var:-**The variable which is declared as var can be re-declared and reinitialized multiple times.

**Let:-** The variable which is declared as let cannot be re-declared but can be reinitialized multiple times

**Const:-** The variable which is declared as const cannot be re-declared and reinitialized.

|  |  |  |
| --- | --- | --- |
| Variable name | Re-declaration | Re- initialization |
| Var | Possible | Possible |
| Let | Possible | Not possible |
| const | Not possible | Not possible |

**JS EXECUTION**

**On browser:**

**1.In line embedded execution**

<!DOCTYPE html>

<html lang="en">

<head>

    <title>Document</title>

</head>

<body>

    <script>

        document.write('hello everyone') 🡪 to write anything in the body of the web page

    </script>

</body>

</html>

**2.external line embedded execution**

**This can be used to add the .js file and can debug the code in the browser**

<!DOCTYPE html>

<html lang="en">

<head>

    <title>Document</title>

</head>

<body>

    <script src="./Basic/DataType.js"></script>

</body>

</html>

we can add multiple scripts in the script tag

To select path of .js file

./ 🡪 suggests the file in the same folder

If u want to move on to next folder use

../🡪 navigates to the preceding or ancestor

**OFF Browser: No browsers will be involved for code execution**

**Program execution in memory**

In initial stages when java script code is executed an object of window will be created and global variable also been created. The global variable is called this and as per global level, window should be strictly equal to this

There are 2 phases will be created

1. Creation phase
2. Execution phase

**1.Creation phase**

In creation phase the memory will be allotted for all variables as memory address as a variable name and will be initialized as undefined and in case of functions , a memory will be allotted as function name and implementation of the function is stored.

**2.Execution phase**

In execution phase all undefined variables are overridden with the actual values and in functions if it is invoked it will be executed.

**Function execution in memory**

**In intial stages** When the function is executed, a function execution context is created above the global execution context in call stack. The two type of phases performed here are

**1.variable hosting**

The memory will be allotted for all variables as memory address as a variable name and will be initialized as undefined and In execution phase all undefined variables are overridden with the actual values.

**2.Function hosting**

Initially the implementation of the function is stored in the memory which address is a function name and when it is invoked it will be executed.

Finally after the execution of the function the function execution context will be teared down from the call stack.