**JavaScript Notes**

**INTRODUCTION**

* **Java is:**
  + **Cross-platform**: Software or language can run on multiple operating systems (like Windows, macOS, Linux, Android, iOS) without major code changes.
  + **Dynamically Typed Language**: The datatype is automatically determined at run time.
  + It handles **server-side** as well as **browser-side** operations.
* **ECMAScript** is the official specification (standard) that defines how the JavaScript language should work.  
  Think of ECMAScript as the **blueprint**, and JavaScript as one of the **real-world implementations** of that blueprint.
* The **console** is a tool that shows messages and allows you to run JavaScript code interactively.
* **JavaScript is case sensitive.**

**Comments in JavaScript**

// a one line comment

/\* this is a longer,

\* multi-line comment

\*/

**Primitive Data Types**

🧱 There are **6 primitive types** in JavaScript:

| **Type** | **Description** | **Example** |
| --- | --- | --- |
| string | Text data | "Hello", 'World' |
| number | Numbers (integers or decimals) | 42, 3.14 |
| boolean | Logical values | true, false |
| null | Represents intentional absence of value | null |
| undefined | Variable declared but not assigned | undefined |
| symbol | Unique and immutable values (used in objects) | Symbol('id') |

**Other Concepts**

* Programming languages usually follow **PEMDAS** rule for order of operations.
* NaN stands for **Not a Number**.

**Variables**

Variables are like **labels** on boxes that store data.  
Example:

* When the label is **number**, you store numeric data.
* When the label is **string**, you store text data.

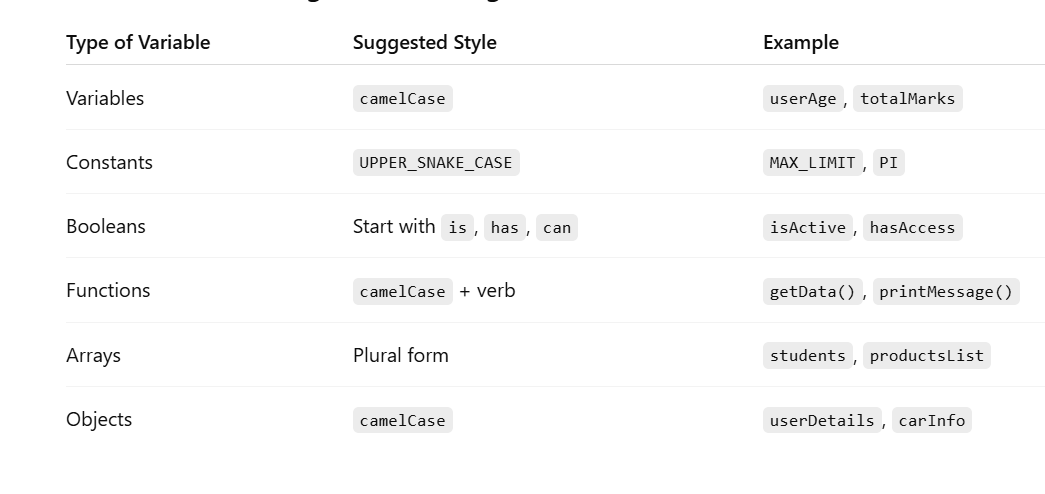
**Basic Syntax for Variable Declaration**

let VariableName = Value;

let year = 2004;

**Const** also used to initialize a variable just like **let** keyword but the only difference is we cant change or update the value after initializing.

**Var** is also used but it’s a traditional way of declaring a variable it act’s same like let.



**Strings in JavaScript**

Strings are **primitive data types** and are **indexed**, meaning each character has a position number starting from 0.

**Common String Methods**

let myString = "Madavaram yethishwar Rao";

// Extract substring from index 0 to 10 (excluding 10)

myString.slice(0, 10);

// Output: "Madavaram"

// Check if string ends with 'r'

myString.endsWith('r');

// Output: false

// Concatenate strings

myString.concat(" chintu", " Kumar");

// Output: "Madavaram yethishwar Rao chintu Kumar"

// Find index of first occurrence of "y"

myString.indexOf("y");

// Output: 10

// Get string length

myString.length;

// Output: 24

// Convert to uppercase

myString.toUpperCase();

// Output: "MADAVARAM YETHISHWAR RAO"

// Convert to locale uppercase (similar to toUpperCase)

myString.toLocaleUpperCase();

// Output: "MADAVARAM YETHISHWAR RAO"

// Split string into array (default splits by comma if argument is omitted)

myString.split();

// Output: ["Madavaram yethishwar Rao"]

// Add spaces around the string

myString = " " + myString + " ";

// Output: " Madavaram yethishwar Rao "

// Trim spaces and convert to locale uppercase

myString.toLocaleUpperCase().trim();

// Output: "MADAVARAM YETHISHWAR RAO"

More Examples with slice and replace

myString = "Madavaram yethishwar Rao";

myString.slice(0, 9);

// Output: "Madavaram"

myString.slice(10, 20);

// Output: "yethishwar"

myString.replace("Madavaram", "M.");

// Output: "M. yethishwar Rao"

**Template Literals**

Template literals allow combining expressions and variables inside strings in a clean, readable way, using backticks ` and ${} syntax.

const name = "Yethishwar";

const age = 21;

console.log(`My name is ${name} and I am ${age} years old.`);

// Output: My name is Yethishwar and I am 21 years old.

**Null vs Undefined**

| **Term** | **Meaning** |
| --- | --- |
| **Null** | Intentional absence of any value |
| **Undefined** | Variable declared but no value assigned (default) |

**Random Numbers in JavaScript**

Math.random() returns a random decimal between **0 (inclusive)** and **1 (exclusive)**.

**Generate random number between 1 and 10**

const randomNumber = Math.floor(Math.random() \* 10) + 1;

console.log(randomNumber);

Math.random() \* 10 generates a decimal between 0 and 9.999...

Math.floor() rounds down to an integer between 0 and 9

Adding 1 shifts range to 1 through 10

m

## Comparision Operators(<, >, ==, <=, >=, !=, ===)

* Double Equality(==) -> It ignore about the data types just checks the values

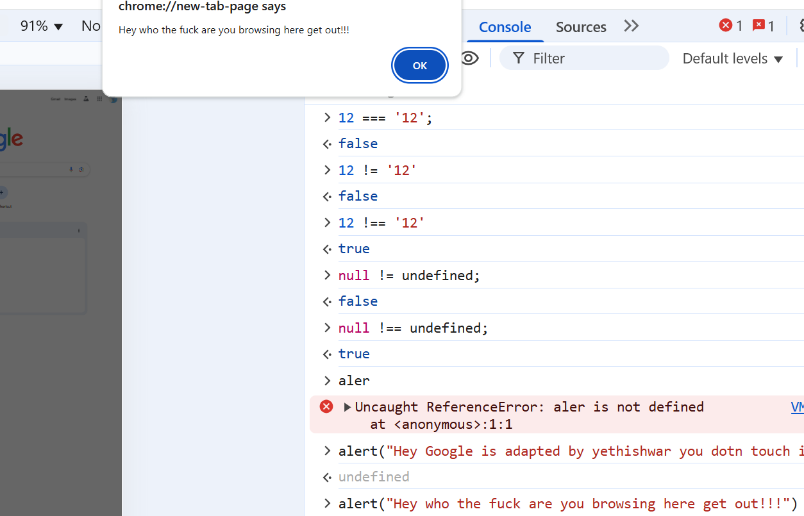
12 == '12';

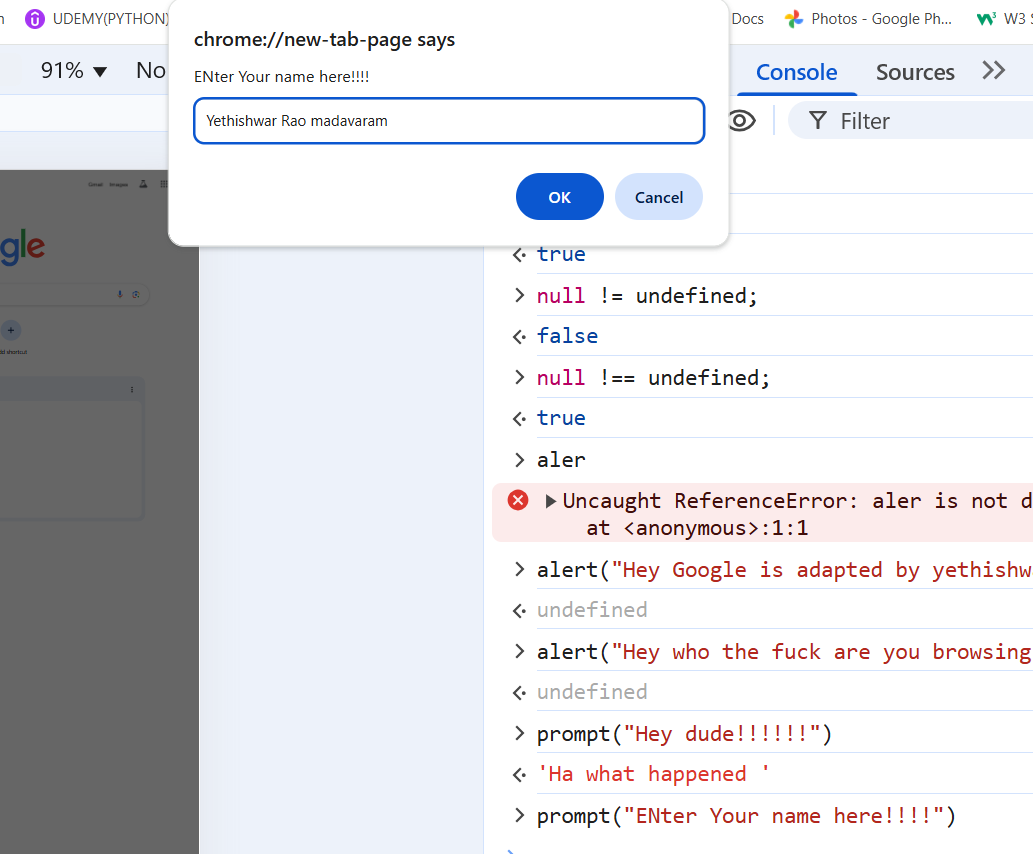
Null == undefined;

* True
* Triple Equality(===) -> It cares about the data types -> preferable

12 === '12';

* False





parseInt("8783");

* 8783

# Control Statements Example Code

let age  =  prompt("Please Enter Your Age??");

if(age > 65){

    console.log("You are a Senior citizen!!!")

}

else if(age > 35){

    console.log("You are Married!!!");

}

else if(age > 20){

    console.log("You are a Young Star!!!!");

}

else if(age > 10){

    console.log("You are a Kid!!!");

}

else{

    console.log("You Born recently how can you give input Here!!!");

}

p = "Hi Hello";

-> 'Hi Hello'

p.indexOf(" ");

-> 2

p = "HiHello";

-> 'HiHello'

p.indexOf(" "); //If no space exists in string means it will print -1 or else

-> -1 prints space index;

## Truthy values and Falsy values

Falsy values:- false, 0, “”, null, undefined, NaN , everything else is Truthy Values

## Logical operators

These operators are used to combine different expressions logically.

&&, ||, !

## Arrays

Ordered collection of values each element in the array having an index label starting from 0 by which we can access through those numbers.

Push -> add to the end

Pop -> remove from end

Shift -> remove from start

Unshift -> adds to start

Concat -> merge arrays

Includes -> look for a value

indexOf -> finding index of some element

join -> creates a string from an array

reverse -> reverses an array

slice -> copies a portion of an array

splice -> removes/ replaces elements splice(start, deletecount, replacement values);

sort -> sorts an array

## Javascript Object Literals

Objects are collection of preoperties… it is similar like dictionaries in python

Properties here are key-value pairs

let person = {

name: "Alice", // person.name -> Alice

age: 25,

isStudent: true,

greet: function() {

console.log("Hello!");

}

};

* We can store any type of datatype inside values in the keys.
* All keys are comverted to strings so that we will access values inside like this : **person[“name”]**
* We can also access values like this **person.name**
* We can modify values just like arrays bu tinstead of indices just assign keys **person.name = “new value”**
* Most common practice or usecase is including objects inside arrays…

## Repeating Stuff With Loops

| **Loop** | **When to Use** | **Best For** |
| --- | --- | --- |
| for | Fixed number of repetitions | Arrays, counters |
| while | Unknown repetitions | User input, waiting conditions |
| do...while | Must run once, then check | Menu-like inputs |
| for...of | Iterate values | Arrays, strings |
| for...in | Iterate keys | Objects |

## Looping over an arrays

const animals = [‘cheetah’, “Elephant”, “Dear”, “bear”]

for(let i = 0; i < animals.length; i++){

console.log(animals[i]);

}

## Nested Loops

for(let i = 0; i < 5; i++){

console.log(`This is Outer Loop: ${i}`);

for(let j = 0; j < 2; j++){

console.log("This is Inner LOOP");

}

}

let myarray= [["ethishwar", "aadharaf", "aeeff"],

["Madacaraa", "chintu","Preethi"]];

console.log("Yethishwar");

for (let i = 0; i < myarray.length; i++){

console.log(myarray[i]);

for(let j = 0; j < myarray[i].length; j++){

console.log(myarray[i][j]);

}

}

## While Loop

* Generally used for fixed number of iterations

## Break keyword

* Used to stop the iteration after certain condition is met

## For.. 0f

* Used when accessing elements in an array by going through its indices
* Used to access charecters inside a string

let myarray= [1, 2, 3, 4, 5, 5];

for(i of myarray){

console.log(i); // Prints 1 2 3 4 5 sequentially

}

## For.. in

* Used for Object literals to iterate over the elements
* We have separate functions to access keys, values and entries using these inbuilt functions
  + Object.keys()
  + Object.values()
  + Object.entries() respectivelys

## Functions

function functionName(parameters) {

// Code Logic

}

// RETURNING DAY OF THE WEEK :

function returnDay(num){

if(num <= 7 && num >= 1){

let days = {

1: "Monday",

2: "Tuesday",

3: "Wednesday",

4: "Thursday",

5: "Friday",

6: "Saturday",

7: "Sunday"

}

return days[num];

}

else{

return null;

}

}

result = returnDay(5);  
  
  
  
  
//SUM ARRAY ELEMENTS  
function sumArray(myarray){

let result = 0;

for(let i=0; i<myarray.length; i++){

result += myarray[i]

}

return result;

}

sumArray([1,2,4]);

//CAPITALIZING EXERCISE

function capitalize(mystring){

first\_letter = mystring[0].toUpperCase();

return first\_letter + mystring.slice(1, mystring.length+1);

}

capitalize("eggPlant");