**JavaScript Notes**

**Environments – NODE JS and DENO JS**

Variables

Variables are containers for storing information. Creating a variable in JavaScript is called "declaring" a variable. After the declaration, the variable is empty (it has no value).

**const** – Unchangeable variables.

**Ex-** const accountId = 12345;

let – Changeable variables.

Ex- let accountEmail = “[ayush@google.com](mailto:ayush@google.com)”;

var – Old way, Problem in block scope and functional scope, Avoid using it.

Ex- var accountPassword = “12345”;

No Keyword – Can declare variables without any keyword but does’t look professional.

Ex- accountCity = “Kushinagar”;

Note: If a variable is declare without any value then its default value will be undefined.

Ex- let accountState;

“use strict”;

This “use strict”; syntax treat all JS code as newer version.

# DATA TYPES

Data types in JavaScript define the data type that a variable can store. JavaScript includes primitive and non-primitive data types. The primitive data types in JavaScript include string, number, boolean, undefined, null, and symbol. The non-primitive data type includes the object.

PRIMITIVE DATA TYPES

**string** - A string is a sequence of one or more characters that may consist of letters, numbers, or symbols.

**Ex-** let name = “Ayush”;

**number** - Numbers are a fundamental data type in JavaScript and are used in many different ways. Limit = 2^53

**Ex-** let age = 21;

**bigInt** - JavaScript BigInt variables are used to store big integer values that are too big to be represented by a normal JavaScript Number.

**Ex-**

const big=BigInt(12345678901234567890123456789012345678);

**boolean** - In JavaScript, a boolean value is one that can either be TRUE or FALSE.

**Ex-** let isLoggedIn = false;

**undefined** - In JavaScript, the undefined data type is a primitive data type that denotes the absence of a value. It is one of the two falsy values in JavaScript, along with null.

**Ex-** let a;  
 console.log(typeof a); // undefined

**null** - In JavaScript, the null value represents the intentional absence of any object value. It is one of JavaScript's primitive values and is treated as falsy for boolean operations. The value null is written with a literal: null.

**Ex-** let myVariable = null;

console.log(typeof null); // object

**symbol** - The JavaScript Symbol is a primitive data type, just like Number, String, Boolean, etc. It represents a unique identifier and can be used in various ways.

**Ex-** const symbol = Symbol('description');

NON-PRIMITIVE DATA TYPE

**object** - With objects we can store a collection of data inside one entity. It is a non-primitive data type that consists of unordered key-value pairs.

**Ex-** const person = {  
 firstName: "John",  
 lastName: "Doe",  
 age: 50,  
 eyeColor: "blue"  
 };

TYPE CONVERSION

conversion in number

To convert any data type in number we use Number function.

Ex- let score = "33";

let value = Number(score); // value = 33, typeof value = number

Ex- let score1 = "33abc";

let value1 = Number(score1); // value1 = NaN, typeof value1 = number

Ex- let score2 = null;

let value2 = Number(score2); // value2 = 0, typeof value2 = number

Ex- let score3 = undefined;

let value3 = Number(score3); // value3 =NaN, typeof value3 = number

Ex- let score4 = true;

let value4 = Number(score4); // value4 = 1, typeof value4 = number

conversion in String

To convert any data type in number we use String function.

Ex- let num = 33;

let str = String(num); // str = ‘33’, typeof str = string

conversion in Boolean

To convert any data type in number we use Boolean function.

Ex- let isLoggedIn = 1;

Let bool = Boolean(isLoggedIn); // true

Ex- let isLoggedIn2 = 0;

let bool2 = Boolean(isLoggedIn2) // false

Ex- let isLoggedIn3 = ""; // Empty string

let bool3 = Boolean(isLoggedIn3); // false

Ex- let isLoggedIn4 = "Ayush";

let bool4 = Boolean(isLoggedIn4); // true

conversion in Operations

console.log("1" + 2); // 12

console.log(1 + "2"); // 12

console.log("1" + 2 + 2); // 122

console.log(1 + 2 + "2"); // 32

console.log(+true); // 1

console.log(+false); // 0

console.log(+""); // 0

Operations in JavaScript

console.log(2+2);

console.log(2-2);

console.log(2\*2);

console.log(2\*\*3); // Power

console.log(2/2);

console.log(2%3); // Remainder

Comparison in JavaScript

In JavaScript we use == (Equality operator), <, >, <=, >=, != to compare values.

Ex-

console.log(2<1); // false

console.log(2>1); // true

console.log(2 == 1); // false

console.log(2 != 1); // true

**Note:** Normal Operator (>, <, ==, <=, >=, !=) don’t compare data type its only compare value.

**Ex-**

console.log("2" > 1); // true

console.log("1" > 2); // false

**Note:** To compare Data Type and value both we use ===, !== operator.

**Ex-**

console.log("2" === 1); // false

console.log("1" !== 2); // true

Comparison with null

console.log(null < 0); // false

console.log(null > 0); // false

console.log(null == 0); // false

console.log(null >= 0); // true

**Note:** == (Equality Operator) convert null into NaN that’s why its result will be false and other comparison operators convert null into 0 that’s why (null>=0) and (null<=0) will be true.