**JavaScript day 4 (12-09-24)**

J. Rahul Yadav

**What are Datatypes and Types of Primitive Datatype?**

**DataType:** Data types are values which are stored in iables and inform the interpreter that what type of data we are handling .

Basically JavaScript is a **Dynamic language** so that we can **declare** the variables without telling what type of variable we are using .

**There are different types of datatypes . They are:-**

1. **Primitive .**
2. **Non-Primitive.**

Today let’s see Primitive Datatypes:-

**Primitive:-** primitives are most basic datatype which store only single value and immutable in nature.

* The values once assigned cannot be **modified** but can be **reassigned**.
* Each primitive can hold upto **single** value only.
* Primitives are stored in **stack** memory.
* These datatypes are **built-in datatypes** provided by **JavaScript**.

**Primitives are stored directly in variable**

For example:

let a = 5;

let b = a;

b = 10;

console.log(a); // Output: 5

console.log(b); // Output: 10

You can see that when I initialized **a** with 5 it will be stored in stack memory as **(a=5)** and when I used **(b=a)** I am passing the value of a which is already stored in stack memory and b will create the **new value** in stack memory which I passed as **a** .

And after that i **reassigned b = 10;** then o/p will be 10cause it is independent with **a** and

**A** and **b** are two different now

let a = 5;

let b = a;

b = a+10;

console.log(a); // Output: 5

console.log(b); // Output: 15

you can now see that **b** is independent of **a** and **b** has been created as separate **variable in stack memory** where I have been passed the value of **a** to it by **pass by value** .

**There are different types of primitives they are :-**

1. **Numbers**
2. **Boolean**
3. **String**
4. **Null**
5. **Undefined**
6. **Symbol** (Outdated).

**01.Numbers:-** Basically if any numbers are given as values then it will consider it self as numbers only but if we use **Type of** then we can know whether it is a int or float or double

**Let’s look into this:-**

1. **int**: whatever we write as numbers without points and prefix or suffix then it is stored as int only but it has some limit then after that it is stored as BigInt

Example:

var a = 68416;//it is int

1. **float:** If a number has decimal points then it is considered as float but it has a limit of 6decimal points after 6 it is considered as double.

**Example:**

var a = 10.1641// it is float

1. **double:** It is same as float only but the thing is float can store up to 6 decimal points but it can store more than that .

**Example:**

var a = 10.15115154185515; // it is double .

1. **octal:-** the values which are stored with a suffix of 0o or 0O are known as octal values

**Example:-**

var a = 0Oab12;//it is octa

1. **hexa:** the values which are stored with a suffix of 0x or 0X are known as octal values

**Example:-**

var a = 0Oab12;//it is hexa

1. **BigInt:** int also stores values up to a limit after that it is considered as BigInt

**Example:**-

var a = 6584515616352163521635241n//it is BigInt

**note:** n is denoted that it will be considered as BigInt.

**02.Boolean:** It is also a **primitive** where it is used for **logical** **operations**

* Boolean has only two values **True** or **False.**
* Value of true is **1**.
* Value of false is **0**.
* The default value of Boolean is **False**.

**Syntax**:

Boolean a = true; //true

**03.Null**: Null is also **primitive** data only but it is used when no value is assigned to a variable or to say it is not **points** to any **objects.**

* When **type of** is used to null it shows **object** as o/p.
* Null defines as **empty** **object**.
* Null refers to empty value **Null** = **0**;
* It is used in **absence** of any **object value**.

**04.Undefined**: It is also a **primitive** datatype and primarly if a **variable**, **function** is **declared** and not **assigned** then the value of the variable, function is **undef** right because of **GEC** and when we access it also by **logging** it shows **undef** only due to the **hoisting.**

* undef is not equal to Null
* undefined indicates that a variable has not been assigned a value.