JavaScript

Reference Types

VS

Non-Reference Types

1. Why It's Important

2. Understanding The Difference Between Reference And Non-Reference (Primitive) Types Is Crucial For Effective JavaScript Programming. Let's Dive In!

Sayyed Siddique

Swipe

1 Non-Reference (Primitive) Types

WHAT ARE PRIMITIVE TYPES?

Primitive Types Are Data That Is Stored Directly In The Location That The Variable Accesses.

1. **Number:** Let Num = 42;
2. **String:** Let Str = "Hello";
3. **Boolean:** Let Bool = True;
4. **Null:** Let N = Null;
5. **Undefined:** Let U;
6. **Symbol:** Let Sym = Symbol("Id");
7. **BigInt:** Let BigInt = 123n;

Sayyed Siddique

Swipe

2 Characteristics of Primitive Types

1. **Immutable:** Cannot Be Changed
2. **Stored Directly In The Variable**
3. **Copied By Value**

Sayyed Siddique

Swipe

3 Reference Types

WHAT ARE REFERENCE TYPES?

Reference Types Are Objects That Are Stored As A Reference In The Memory.

1. Object: Let Obj = { Name: "Alice" };

2. Array: Let Arr = [1, 2, 3];

3. Function: Let Func = Function() {};

Sayyed Siddique

Swipe

4 Characteristics of Reference Types

1. **Mutable:** Can Be Changed
2. **Stored As A Reference** (Memory Address)
3. Copied By Reference

Sayyed Siddique

Swipe

5 Differences in Behavior

1. Primitive Types: Copied By Value. Changing One Variable Doesn't Affect The Other

Example: Let A = 10; Let B = A; B = 20; // A Is Still 10

2. Reference Types: Copied By Reference. Changing One Variable Affects The Other.

Example: Let A = 10; Let B = A; B = 20; // A Is Still 10

Sayyed Siddique

Swipe

6 Example Code
Primitive Types

```
let a = 5;

let b = a;

b = 10;

console.log(a); // 5

console.log(b); // 10
```

Sayyed Siddique

Swipe

7

Example Code
Reference Types

```
let obj1 = { value: 5 };
let obj2 = obj1;
obj2.value = 10;
console.log(obj1.value); // 10
console.log(obj2.value); // 10
```

Sayyed Siddique

Swipe

Sayyed Siddique

Thanks for reading

Follow for more

“Please like; if you find value, consider following!„