What is "Call by Value"?

In JavaScript, when you assign or pass a **primitive value** (like number, string, boolean, null, undefined, symbol, or bigint), **a copy** of that value is used — **not the original variable or its memory reference**.

🔑 **In simple words:**

JavaScript does not pass the original box — it gives you a copy of what's inside the box.

🧪 **Example:**

js

CopyEdit

```
let a = 10;

let b = a;  // Copy of a's value (10) is assigned to b


b = 20;     // Only b changes, a stays the same


console.log(a); // 10

console.log(b); // 20
```

✅ **Why?**

- a holds a primitive value (10).
- When b = a happens, **a copy** of 10 is made.
- Changing b does **not affect** a.

❗ **Important: They don't share memory**

- Primitives are **stored directly** in memory.
- When passed or assigned, they are **cloned**, not referenced.

What does "Dynamically Typed" mean?

In JavaScript:

- You **don’t need to declare the type** of a variable.
- The **type is determined at runtime**, based on the value assigned.
- A variable’s type **can change** during execution.