Saif Mujawar
@webbysaif

Pass by value and Pass by reference explained.

Saif Mujawar
@webbysaif

In JavaScript, primitive data types are passed by value and non-primitive data types are passed by reference.

For understanding passed by value and passed by reference, we need to understand what happens when we create a variable and assign a value to it,

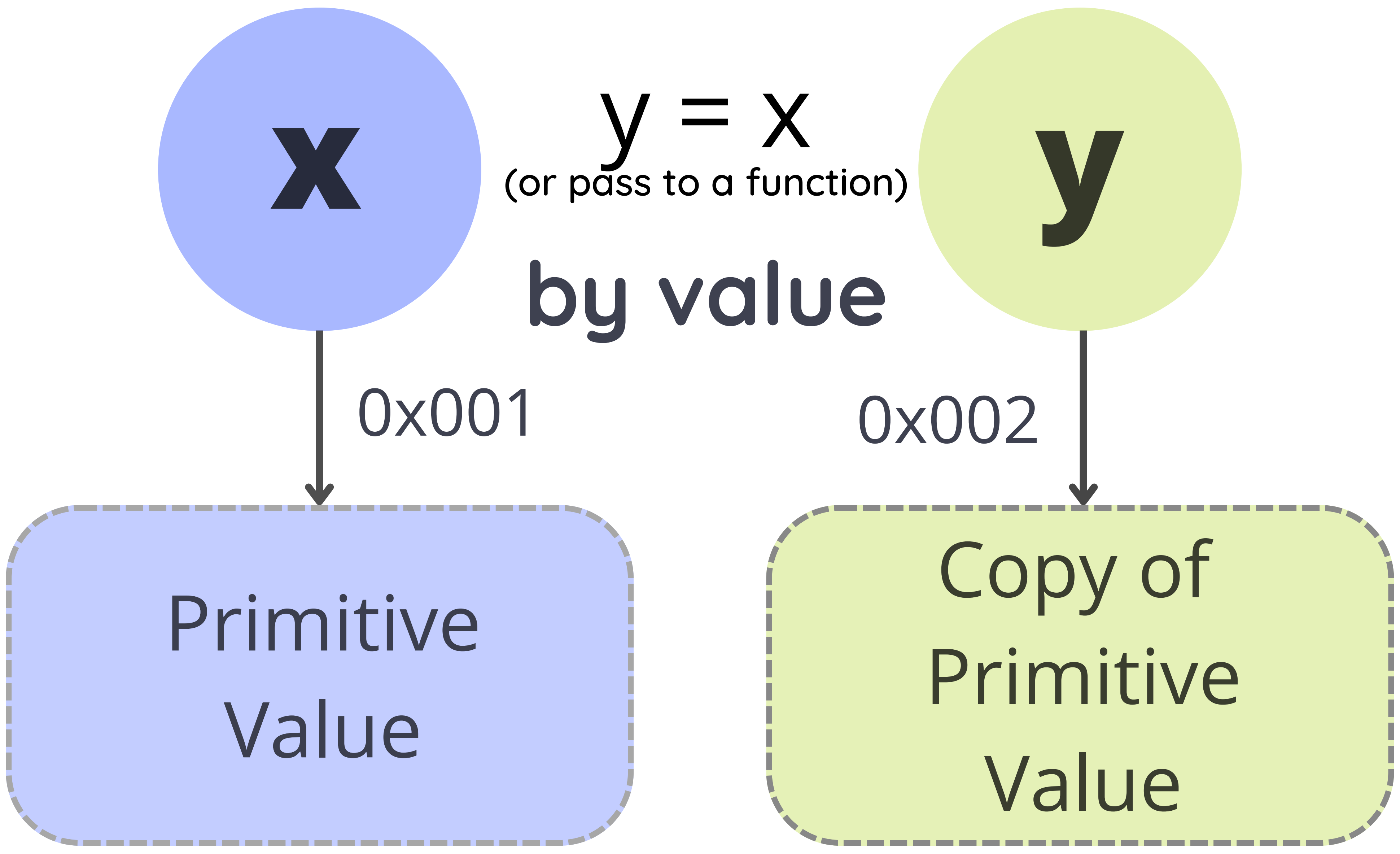
```
var a = 2;
```

In the above example, we created a variable a and assigned it a value of “2”. In the background, the “=” (assign operator) allocates some space in the memory, stores the value “2” and returns the location of the allocated memory space. Therefore, the variable a in the above code points to the location of the memory space instead of pointing to the value 2 directly.

Assign operator behaves differently when dealing with primitive and non-primitive data types,

Saif Mujawar
@webbysaif

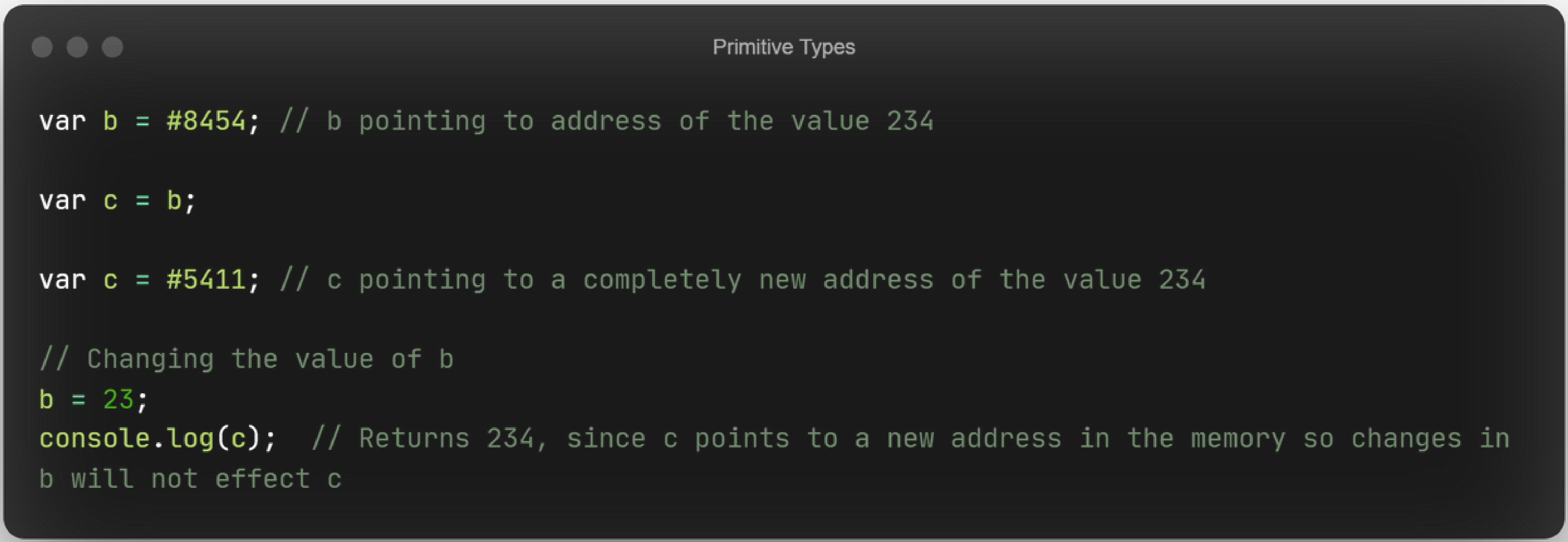
Assign operator dealing with primitive types:



```
var b = 234;
var c = b
```

Saif Mujawar
@webbysaif

In the above example, the assign operator knows that the value assigned to b is a primitive type (number type in this case), so when the second line code executes, where the value of b is assigned to c, the assign operator takes the value of b (234) and allocates a new space in the memory and returns the address. Therefore, variable c is not pointing to the location of variable b, instead, it is pointing to a new location in the memory.

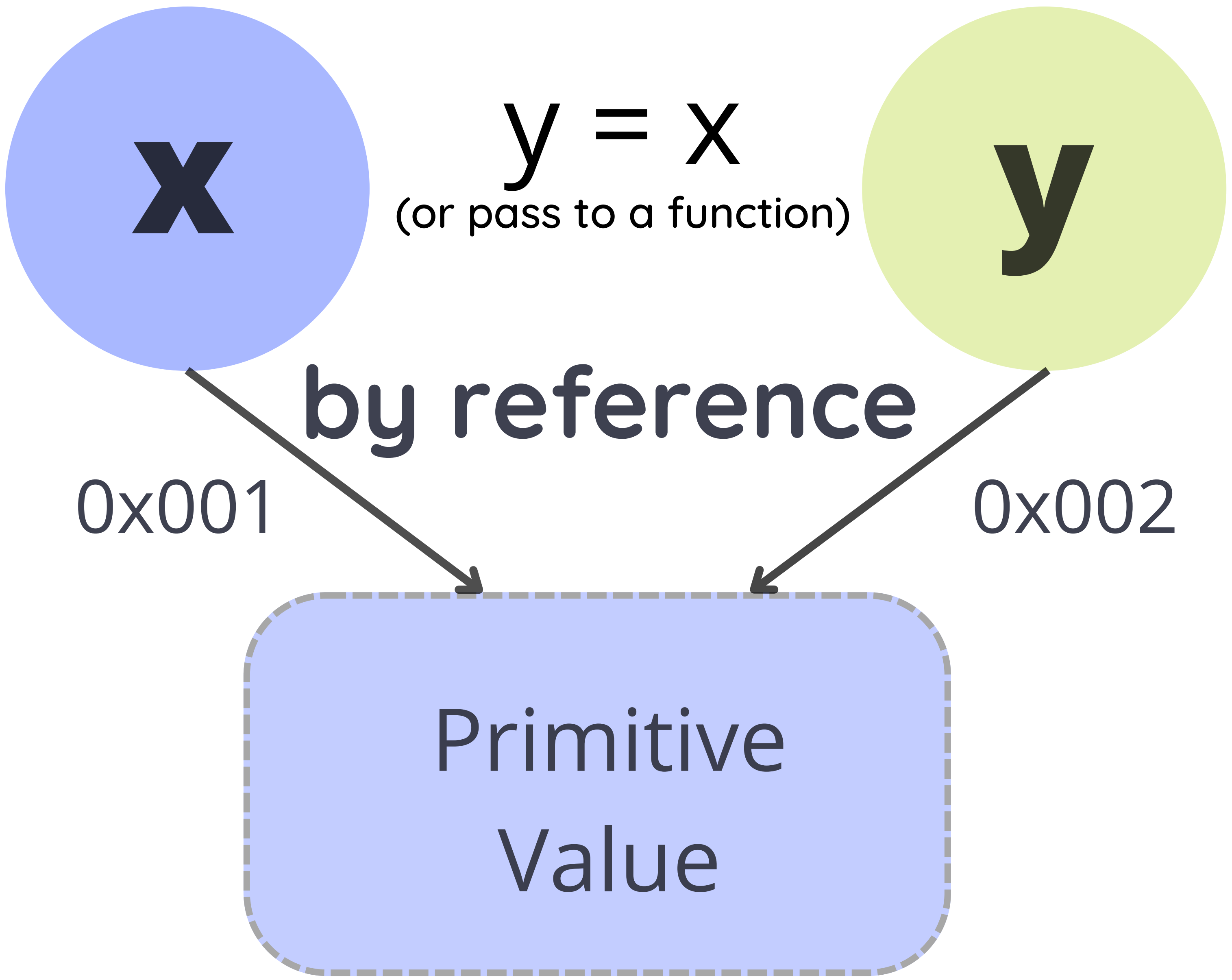


```
var b = #8454; // b pointing to address of the value 234

var c = b;

var c = #5411; // c pointing to a completely new address of the value 234

// Changing the value of b
b = 23;
console.log(c);  // Returns 234, since c points to a new address in the memory so changes in
b will not effect c
```

From the above example, we can see that primitive data types when passed to another variable, are passed by value. Instead of just assigning the same address to another variable, the value is passed and new space of memory is created.

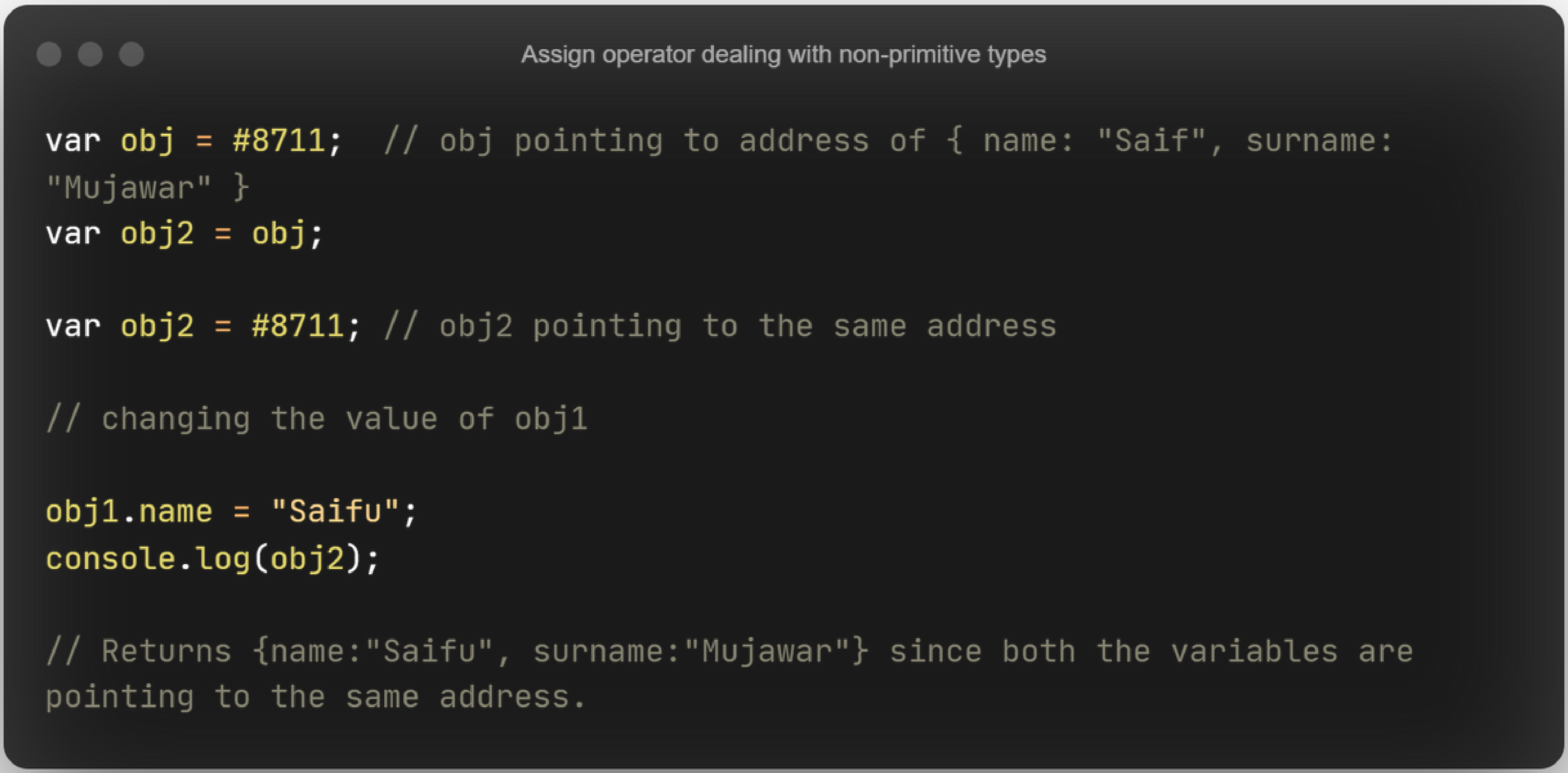
Saif Mujawar
@webbysaif

Assign operator dealing with non-primitive types:



```
var obj = { name: "Saif", surname: "Mujawar" };
var obj2 = obj;
```

In the above example, the assign operator directly passes the location of the variable obj to the variable obj2. In other words, the reference of the variable obj is passed to the variable obj2.



```
var obj = #8711;  // obj pointing to address of { name: "Saif", surname:
"Mujawar" }
var obj2 = obj;

var obj2 = #8711; // obj2 pointing to the same address

// changing the value of obj1

obj1.name = "Saifu";
console.log(obj2);

// Returns {name:"Saifu", surname:"Mujawar"} since both the variables are
pointing to the same address.
```

From the above example, we can see that while passing non-primitive data types, the assign operator directly passes the address (reference).

Therefore, non-primitive data types are always passed by reference.

Turn on notifications 🔔

And That's it!!!

Did you find it useful?

Saif Mujawar
@webbysaif

Follow for more!!