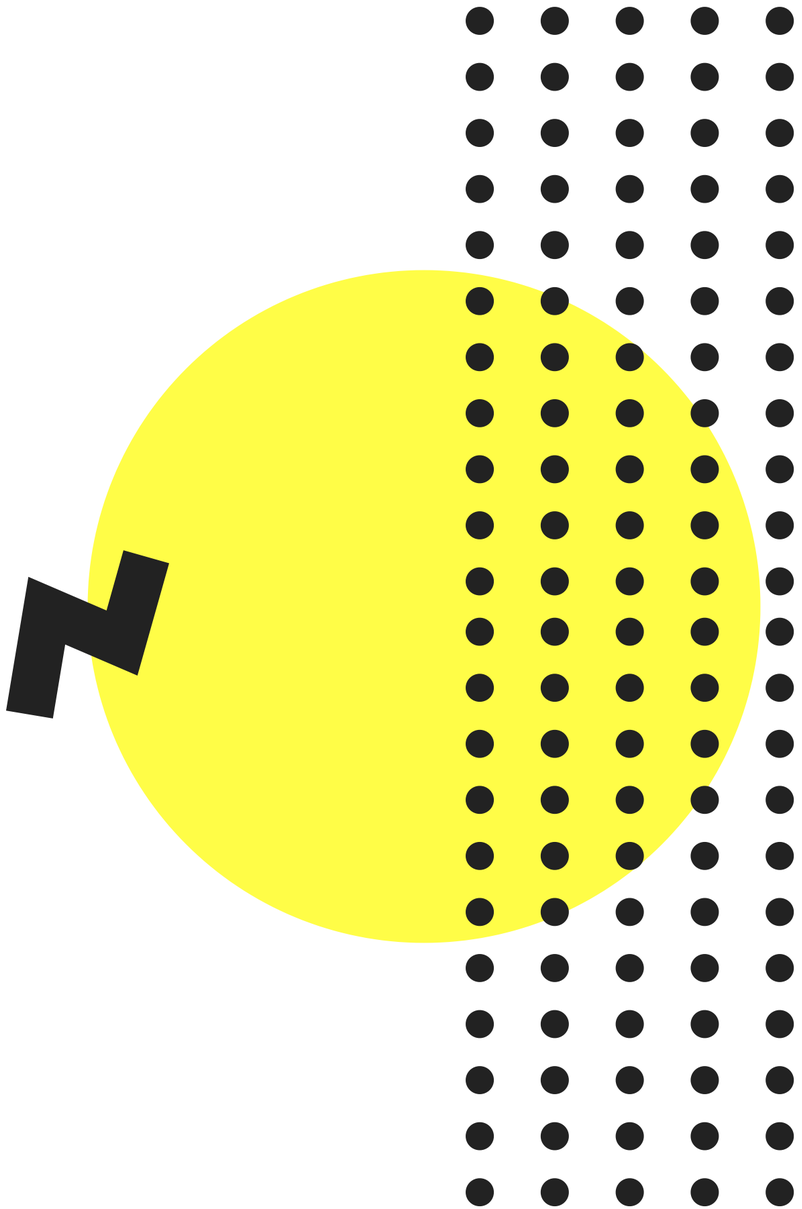
JavaScript

Primitive & Reference data types

FRANCISCO CHIARINO

What is what?

PRIMITIVE	REFERENCE
Boolean	Array
Null	Function
Undefined	Object
String	
Number	

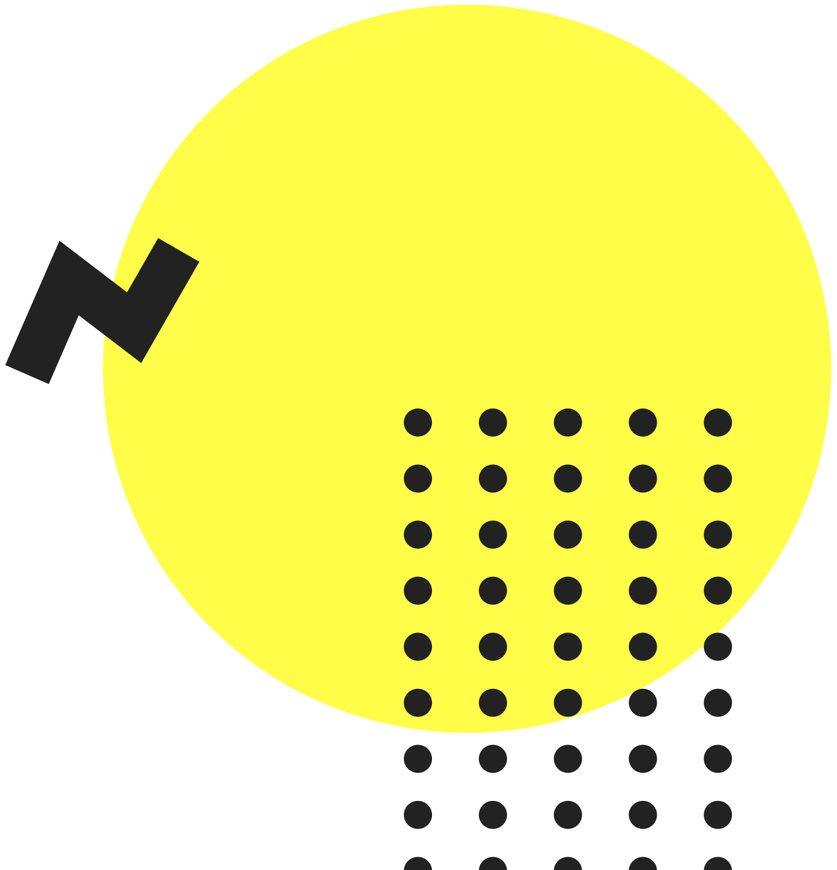


PRIMITIVE

When we assign a primitive value to a variable, we copy the value to value to the variable

```
const a = 10;
const b = 'dog';
const x = a;
const y = b;
```

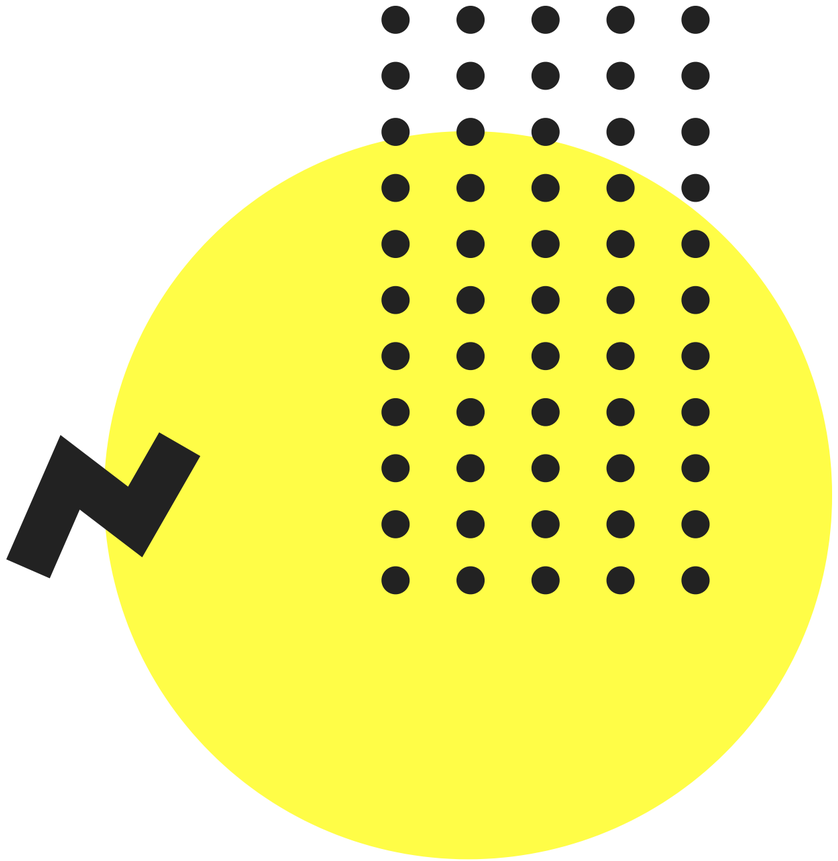
Changing one doesn't change the other.



REFERENCE

Reference values are created in the computer's memory. When we assign the, we are just pointing to the value's location in memory.

The variable doen't actually contain the value.



REFERENCE EXAMPLES

```
const reference = [];
reference.push(1);
```

The values is an address that points the memory location where the array is. When we change its value, the array in memory is what changes.

```
const myArray = [1,2];
const yourArray = myArray;
```

Each variable now contain a reference to the same array.

REMEMBER!

REASSIGNING A REFERENCE

Reassigning a reference value will replace the old reference.

REFERENCE TO SAME ITEM

If the variables contain an reference to the same item, the comparison will be true.

IDENTICAL PROPERTIES

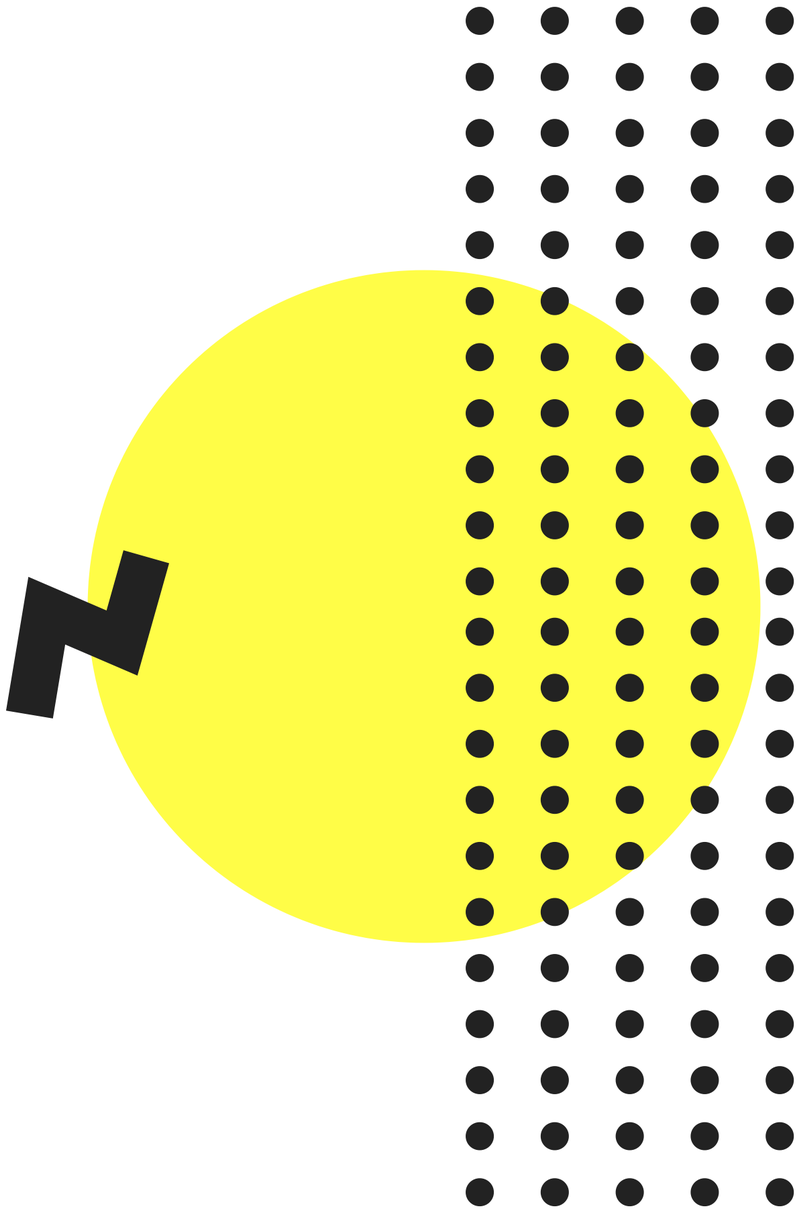
If two distinct objects contain identical properties, the comparison between the will be false.

FUNCTIONS

PRIMITIVE VALUES

When we pass primitive values into a function, the function copies the values into parameters.

```
const a = 100;
const b = 2;
const multiply = (x, y) => {
  return x * y;
};
const result = multiply(a, b);
```

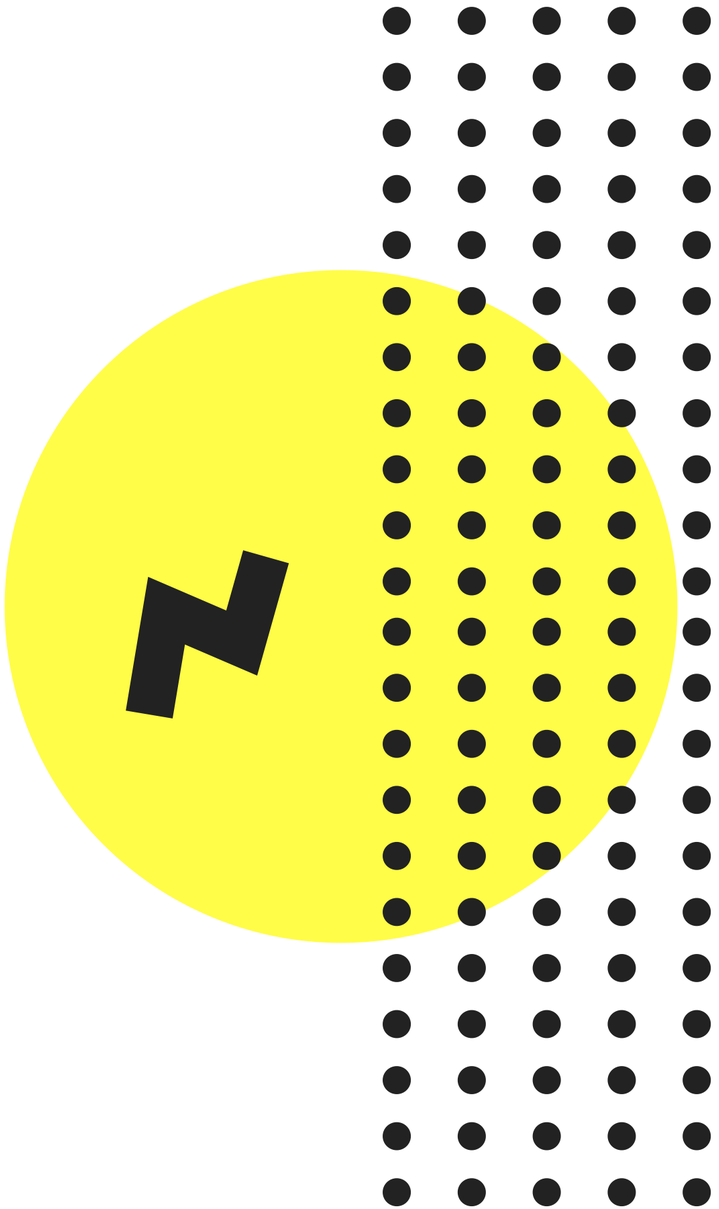


FUNCTIONS

REFERENCE VALUES

A function that takes an object can mutate the state of it's surrounding scope. Changes may persist.

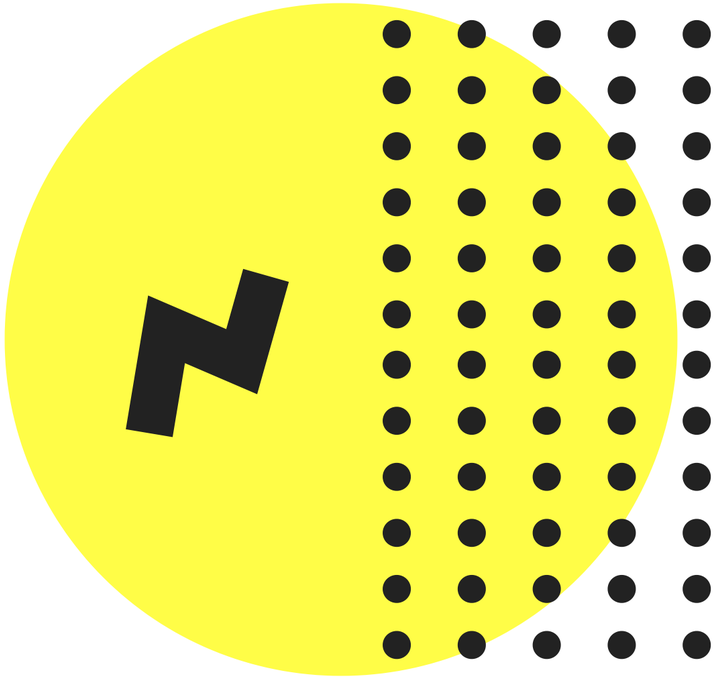
```
const changeAge = (person) => {
  person.age = 25;
  return person;
};
const norbert {
  name: 'Norbert';
  age: 50;
}
```



FUNCTIONS

```
const changedNorbert =
changeName(norbert);

console.log(changedNorbert);
console.log(norbert);
```



THANK YOU!

REFERENCE

codeburst.io