JavaScript Pass-By-Value

Summary: This tutorial explains how JavaScript pass-by-value works and gives you some examples of passing primitive and reference values to a function.

> Before going forward with this tutorial, you should have good knowledge of the primitive and reference values, and the differences between them.

JavaScript pass-by-value or pass-by-reference

In JavaScript, all function arguments are always passed by value. This means that JavaScript copies the values of the variables into the function arguments.

Any changes that you make to the arguments inside the function do not reflect the passing variables outside of the function. In other words, the changes made to the arguments are not reflected outside of the function.

If function arguments are passed by reference, the changes of variables that you pass into the function will be reflected outside the function. This is *impossible* in JavaScript.

Pass-by-value of primitives values

Let's take a look at the following example.

```
function square(x) {
    x = x * x;
    return x;
}

let y = 10;
let result = square(y);

console.log(result); // 100
console.log(y); // 10 -- no change
```

How the script works.

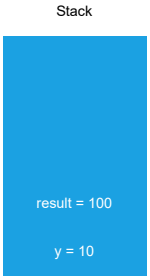
First, define a `square()` function that accepts an argument `x`. The function assigns the square of `x` to the `x` argument.

Next, declare the variable `y` and initialize its value to `10`:

Then, pass the `y` variable into the `square()` function. When passing the variable `y` to the `square()` function, JavaScript copies `y` value to the `x` variable.

After that, the `square()` function changes the `x` variable. However, it does not impact the value of the `y` variable because `x` and `y` are separate variables.

Finally, the value of the `y` variable does not change after the `square()` function completes.



If JavaScript used the pass-by-reference, the variable `y` would change to `100` after calling the function.

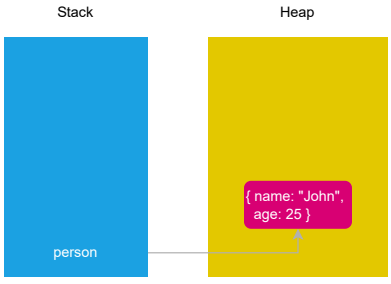
Pass-by-value of reference values

It's not obvious to see that reference values are also passed by values. For example:

```
let person = {
  name: 'John',
  age: 25,
};

function increaseAge(obj) {
  obj.age += 1;
}

increaseAge(person);

console.log(person);
```

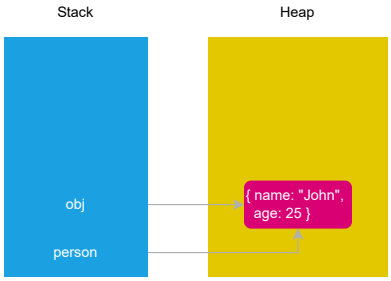
How the script works:

First, define the `person` variable that references an object with two properties `name` and `age` :



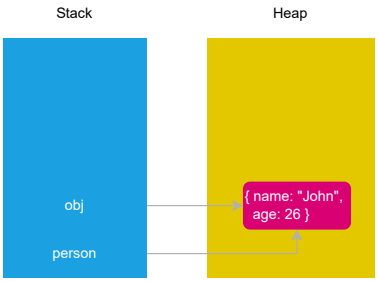
Next, define the `increaseAge()` function that accepts an object `obj` and increases the `age` property of the `obj` argument by one.

Then, pass the `person` object to the `increaseAge()` function:

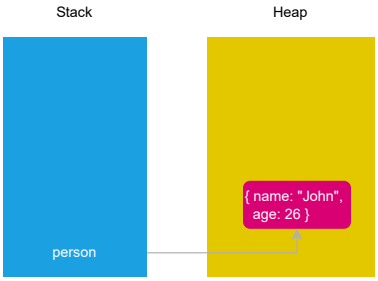


Internally, the JavaScript engine creates the `obj` reference and make this variable reference the same object that the `person` variable references.

After that, increase the `age` property by one inside the `increaseAge()` function via the `obj` variable



Finally, accessing the object via the `person` reference:



It seems that JavaScript passes an object by reference because the change to the object is reflected outside the function. However, this is not the case.

In fact, when passing an object to a function, you are passing the reference of that object, not the actual object. Therefore, the function can modify the properties of the object via its reference.

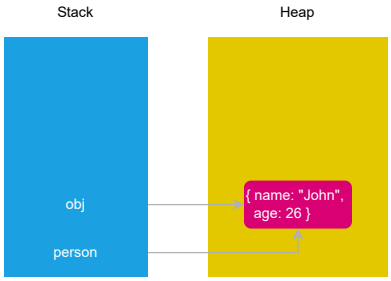
However, you cannot change the reference passed into the function. For example:

```
let person = {
  name: 'John',
  age: 25,
};

function increaseAge(obj) {
  obj.age += 1;

  // reference another object
  obj = { name: 'Jane', age: 22 };
}

increaseAge(person);

console.log(person);
```

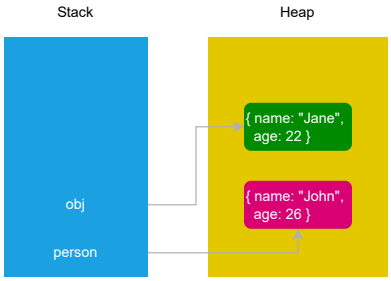
Output:

```
{ name: 'John', age: 26 }
```

In this example, the `increaseAge()` function changes the `age` property via the `obj` argument:



and makes the `obj` reference another object:



However, the `person` reference still refers to the original object whose the `age` property changes to `26`. In other words, the `increaseAge()` function doesn't change the `person` reference.

If this concept still confuses you, you can think of the function arguments as local variables.

Summary

- JavaScript passes all arguments to a function by values.
- Function arguments are local variables in JavaScript.

Quiz