# Value and Reference Types

An entry-level introduction to value and reference types in JavaScript.

## Introduction

JavaScript does not give you full access to your data structures in memory. However, reference types still exist in the language. Mixing value and reference types comes with unwanted side-effects and bugs. Understanding the difference between value and reference types plays a vital role in writing robust programs.

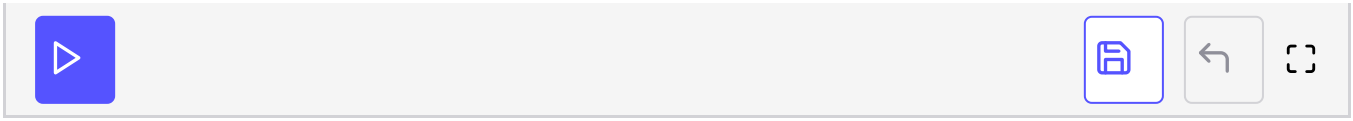
> Numbers, booleans, strings, null and undefined are primitive types. All primitive types are passed by value. Objects, arrays, and functions are passed by reference.

Strings are somewhat special in JavaScript. Unlike many other languages, strings are not defined as arrays of characters. What is more, the character type does not even exist in JavaScript. Strings are immutable data types, with an interface that makes them look like an array of characters. In practice, any modifications to a string result in a new, immutable string value.

After the introduction of value and reference types, let's continue with a riddle. Determine the result of the console logs.

```
var people = [
    { name: 'John Hill', age: 22 },
    { name: 'Jack Chill', age: 27 }
];

var getInitials = function( name ) {
    // Reusing the name argument makes little sense in general.
    // We are making this assignment here for demonstrating
    // the difference between value types and reference types.
    name = name.split( ' ' ).map( function( word ) {
        return word.charAt( 0 );
    } ).join( '' );
```

```
    console.log( name );

    return name;
}

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

var addPerson = function( people, name, age ) {
    people.push( { name: name, age: age } );

}

// Part 1: getInitials
console.log( getInitials( people[0].name ) );
console.log( people[0].name );

// Part 2: increaseAge
increaseAge( people[1] );
console.log( people[1].age );

// Part 3: addPerson
addPerson( people, 'Jim Gordon', 32 );
console.log( people );
```



**Question**:

```
What is logged to the console?
```

**Answer:**

- In the first part, the initials of a name are constructed by the `getInitials` function. The string value passed to the function is copied. This copy is accessed and modified inside the function. These modifications have nothing to do with the original value `people[0].name`.
- The `increaseAge` function accepts an object. As objects are reference types, only the reference to the passed `person` is copied. When members of the referenced object are changed, the changes are kept even after executing the function. These changes are accessible via each reference.
- Arrays are also reference types. Adding an element to an array is a permanent change.

```
JH
JH
John Hill
```

```
28
(index) name         age
0       "John Hill"  22
1       "Jack Chill" 28
2       "Jim Gordon" 32
```