

Note Web Application

Brendon Mendicino

February 28, 2023

Contents

1	Javascript Introduction	3
----------	--------------------------------	----------

1 Javascript Introduction

Javascript is backward compatible, to be able to use the previous features is use the directive:

```
1 "use strict";
```

JS has primitive types and non-primitive types, JS is also and strongly typed language, the primitive types are: string, number, boolean, null, undefined. The non-primitive are the objects, which can be: array, function, user-defined.

The all possible false values in JS: 0, -0, NaN, undefined, null, '', in JS there are two main comparison operators:

```
1 a == b    // equal, convert types and compare
2 a === b   // strict equal, inhibits automatic type conversion
```

In JS you can create variable with:

```
1 // modern
2 let a = 10;    // can be changed
3 const b = 'a'; // constant
4
5 // old
6 var k = 9;
7 j = 30;
```

The difference between null and undefined, is that variable with null they old a value which is null, on the other way if a variable is declared and nothing is associated with it the value olds by default undefined.

A scope is defined by a **block**, which is created with ...

There two kinds of **foreach** in JS, using **in** allows iterating over objects, while **of** allows iterating over iterable objects:

```
1 for (let a in object) {
2   ...
3 }
4
5 for (let b of iterable) {
6   ...
7 }
```

Using arrays:

```
1 let a = [1, 2, 'ok', false];
2 let b = Array.of(1, 2, true);
3 a.push(5);    // append an element
4 b.unshift(2); // insert at the beginning
5
6 let copy = Array.from(a); // shallow copy, it does not deep copy
```

The **destructuring assignment** can be done, it extracts the values from the mast left-hand side:

```
1 let [x, y] = [1, 2];
2 [x, y] = [y, x]      // swap
```

The **spread operator** (...) expands on iterable object into it's values:

```
1 let [x, ...y] = [1, 2, 3, 4];      // y == [2, 3, 4]
2
3 const a = [1, 2];
4 const b = [0, ...a, 3]; // [0, 1, 2, 3]
```

Spreading can be from the left or from the right, usually the spread operator is used for copying array:

```
1 const a = [1, 2];
2 const b = [...a];
```

A **string** in JS is an immutable type (like python) encoded in Unicode. The **template literals** can be done with the **tick** operator `` (expression like Kotlin):

```
1 let name = 'Bre';
2 let sur = 'Mend';
3 // Template literal
4 let fullName = `${name} ${sur}`;
```