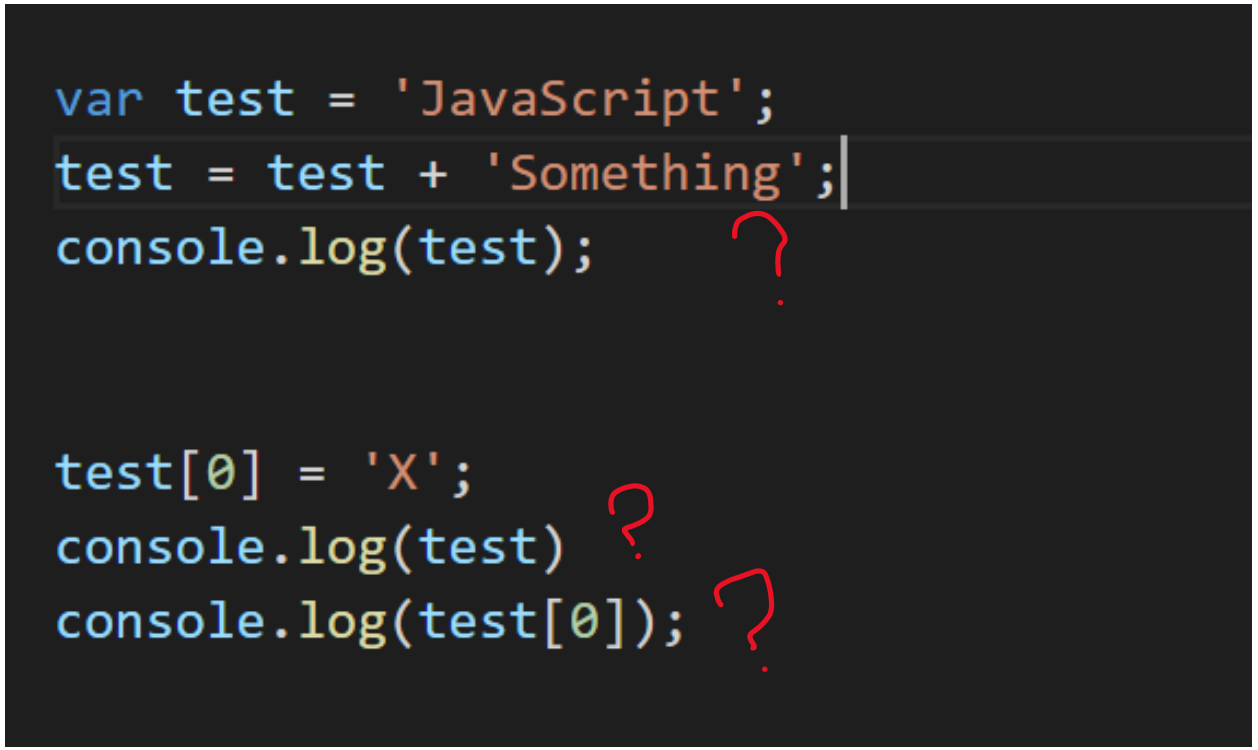
JavaScript variables

Strings είναι immutable

Οι συμβολοσειρές στη JavaScript είναι αμετάβλητες

- δεν μπορούν να τροποποιηθούν μόλις δημιουργηθούν.
- μπορείτε να δημιουργήσετε μια νέα συμβολοσειρά από μια υπάρχουσα συμβολοσειρά.

Strings είναι immutable



Objects

- Η JavaScript υποστηρίζει την δημιουργία objects, δηλαδή αντικειμένων
- Ένα αντικείμενο στη JavaScript είναι μια **συλλογή ονοματισμένων τιμών**
- Στα αντικείμενα λοιπόν έχουμε ζευγάρια key-value (name-value)
- Τα αντικείμενα ορίζονται με τις λέξεις κλειδιά var-let-const

Objects

```
const cat = {
  name: "Javie",
  age: "5",
  color: "white",
  eyeColor: "blue"
};

console.log(cat["name"]);
console.log(cat.age);
```

Objects

- Για να αποκτήσουμε πρόσβαση σε μια ιδιότητα ενός αντικειμένου, χρησιμοποιούμε:

- dot (.)
  - objectName.propertyName

- Array-like notation ([])
  - objectName['propertyName’]

- objectName[expression] // x = " 'propertyName "; objectName[x]

- Αν προσπαθήσουμε να διαβάσουμε μια ιδιότητα που δεν υπάρχει: undefined
- Μπορούμε να διαγράψουμε μια ιδιότητα μέσω του delete operator
- in operator: υπάρχει μια ιδιότητα σε ένα αντικείμενο?

Objects

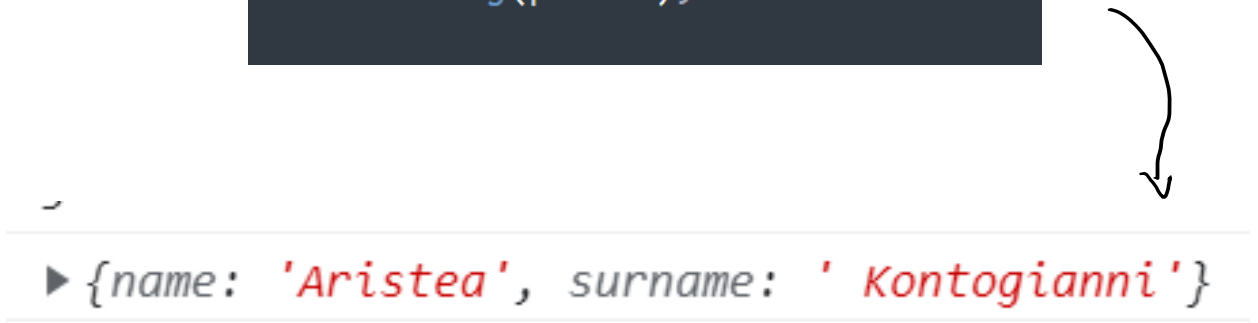
- Επιπλέον μπορούν να χρησιμοποιηθούν μεταβλητές ως ιδιότητες

```
var name="Aristea";
var surname = " Kontogianni";
var person = { name, surname };
console.log(person);
```

Objects

- Επιπλέον μπορούν να χρησιμοποιηθούν μεταβλητές ως ιδιότητες

```
var name="Aristea";
var surname = " Kontogianni";
var person = { name, surname };
console.log(person);
```



Objects

- Τρόποι για να αποκτήσουμε πρόσβαση στις ιδιότητες ενός αντικειμένου :

```
var name="Aristea";
var surname = " Kontogianni";
var person = { name, surname };
console.log(person.name+" "+person["surname"]);
```

```
Aristea  Kontogianni
```

Objects

- Επίσης μπορούμε να προσθέσουμε εμφωλευμένο αντικείμενο μέσα σε ένα object
- Κάθε αντικείμενο στη JavaScript μπορεί να μετατραπεί σε συμβολοσειρά με τη συνάρτηση JSON.stringify ():

```
var name="Aristea";
var surname = " Kontogianni";
var person = { name, surname };
console.log(person.name+" "+person["surname"]);

var city ="Athens";
var country="Greece";
person.address={city,country};
console.log(person);
document.body.innerHTML = JSON.stringify(person);
```

{"name":"Aristea","surname":" Kontogianni","address":{"city":"Athens","country":"Greece"}}

Objects

- JSON.parse()->string becomes a JavaScript object

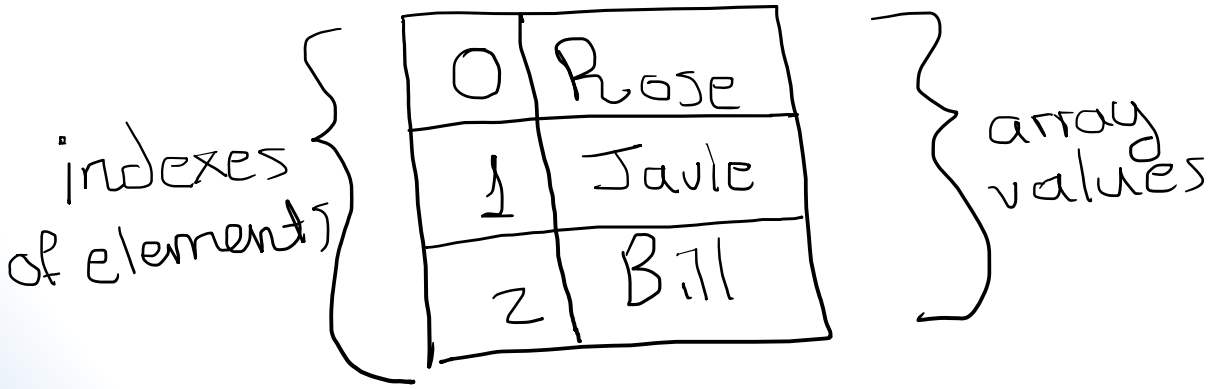
```
var str= '{"name":"John", "age":30, "city":"New York"}';
console.log(typeof(str));
console.log(str);

const obj = JSON.parse(str);
console.log(typeof(obj));
console.log(obj);
```

Objects

- Η μέθοδος Object.freeze() παγώνει ένα αντικείμενο.
- Το πάγωμα ενός αντικειμένου καθιστά τις υπάρχουσες ιδιότητες **μη εγγράψιμες** και **μη διαμορφώσιμες**.
- Δεν είναι πλέον δυνατή η αλλαγή ενός παγωμένου αντικειμένου: δεν μπορούν να προστεθούν νέες ιδιότητες, δεν είναι δυνατή η κατάργηση υπαρχουσών ιδιοτήτων, ούτε να αλλάξουν.
- Η freeze() επιστρέφει το ίδιο αντικείμενο
- Note that: η μέθοδος αυτή δεν “Παγώνει” nested objects/arrays.

Arrays

- Ένα array (πίνακας) αποτελεί μια μεταβλητή που περιέχει περισσότερες από μία τιμές κάθε φορά
- Στη μνήμη, ένα array αντιπροσωπεύει μια ομάδα τοποθεσιών μνήμης
- Κάθε μεμονωμένη τοποθεσία ονομάζεται στοιχείο (element)



Arrays

- Cats -> Δημιουργούμε ένα array με τέσσερα στοιχεία, τα οποία ορίζονται κατά τη δήλωση του πίνακα

```
var cats = ["Ragdoll", "Bengal", "Ocicat","Toyger"];
document.getElementById("demo").innerHTML = cats;
```

Arrays

- Πόσα στοιχεία περιέχουν οι παρακάτω πίνακες;

```
<script>
var cats = ["Ragdoll", "Bengal", "Ocicat","Toyger"];
var cats1 = ["Ragdoll",,,,"Toyger"];
```

Arrays

- Cats -> Δημιουργούμε ένα array με **τέσσερα** στοιχεία, τα οποία **ορίζονται κατά τη δήλωση** του πίνακα

- Cats1-> Δημιουργεί ένα array με **5** στοιχεία, **3** από τα οποία δεν έχουν καθοριστεί ακόμη

```
var cats1 = ["Ragdoll", , , ,"Toyger"];
cats1[1]="Bengal";
document.body.innerHTML =cats1;
```

Ragdoll,Bengal,,,Toyger

typeof Array

- Οι πίνακες είναι στην πραγματικότητα ένας ειδικός τύπος αντικειμένων
- Έτσι, typeof Array επιστρέφει “object"
- Ωστόσο, τα arrays στη JavaScript είναι καλύτερο να περιγράφονται ως arrays

Arrays’ Elements

Ένα array μπορεί να περιέχει :

- μεταβλητές διαφορετικών τύπων
- αντικείμενα
- συναρτήσεις
- άλλα arrays

Stacks/queues

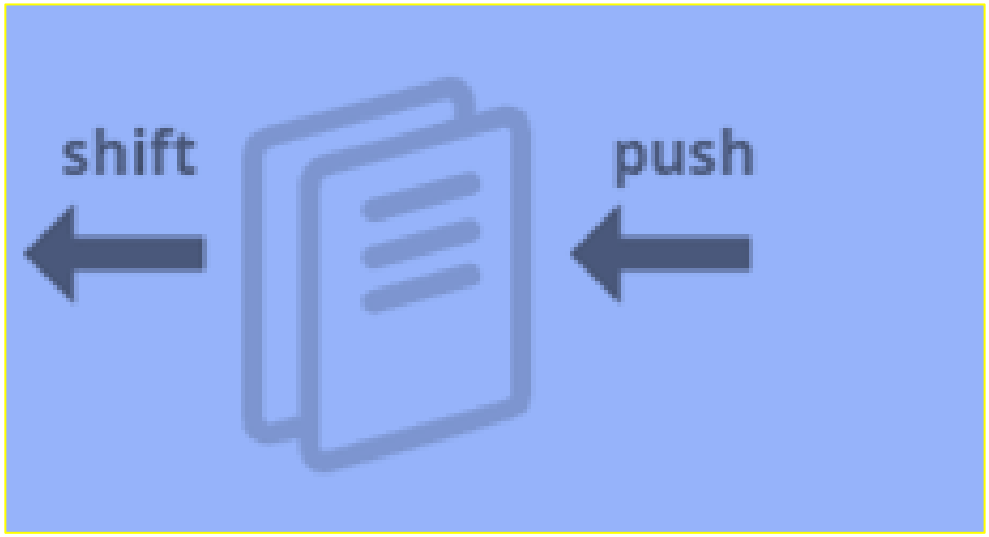
- Τα arrays στη JavaScript μπορούν να λειτουργήσουν τόσο ως **queues** όσο και ως **stacks**
- Επιτρέπουν έτσι την πρόσθεση/αφαίρεση στοιχείων **και από την αρχή και το τέλος**

- **stacks**, latest pushed item is received first LIFO (Last-In-First-Out)
- **queues**, FIFO (First-In-First-Out).

Queue

- **push** προσθέτει ένα στοιχείο στο τέλος
- **shift** εξάγει ένα στοιχείο από την αρχή
  - Έτσι το 2ο στοιχείο γίνεται 1ο
- **unshift** προσθέτει στοιχείο στην αρχή του πίνακα

Queue



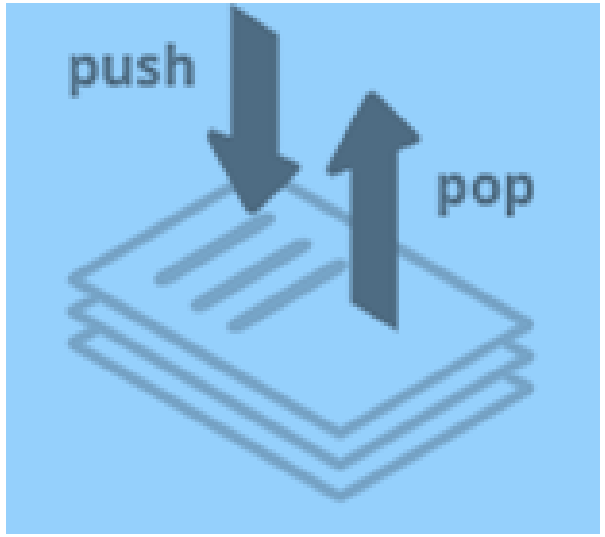
Queue

```
//push
var fruits = ["Apple", "Orange"];

fruits.push("Banana");

alert( fruits ); // Apple, Orange, Banana

//shift

alert( fruits.shift() ); // remove Apple and alert it
alert( fruits ); // Orange, Banana

//unshift

fruits.unshift('Apple');
fruits.unshift('pear');

alert( fruits ); // pear, Apple, Orange, Banana
</script>
```

stack

- push : προσθέτει ένα στοιχείο στο τέλος της στοίβας
- pop : εξάγει ένα στοιχείο από το τέλος στοίβας

stack



stack

```
//pop
var fruits = ["Apple", "Orange","Banana"];

alert( fruits.pop() ); // remove "Banana" and alert it

alert( fruits ); // Apple, Orange

//push

fruits.push("Banana");

alert( fruits ); // Apple, Orange, Banana
```

JS Arithmetic Operators

- Arithmetic Operators : + , - , * , / , % , ++ , --
- Comparison Operators: ==, !=, >, <, >=,<=
- Logical (or Relational) Operators : &&, ||, !
- Assignment Operators: =, +=, -=, *=, /= , %=
- **Conditional (or ternary) Operators: ? :**

  If Condition true? Then value X : Otherwise value Y

  Lets see an example

Lets understand operators better…

X=5

==	equal to	x == 8	false
		x == 5	true
		x == "5"	true
===	equal value and equal type	x === 5	true
		x === "5"	false
!=	not equal	x != 8	true
!==	not equal value or not equal type	x !== 5	false
		x !== "5"	true
		x !== 8	true
>	greater than	x > 8	false
<	less than	x < 8	true
>=	greater than or equal to	x >= 8	false
<=	less than or equal to	x <= 8	true

Source w3schools

Assignment Operators

Operator	This is the same as :	this
=	x = y	x = y
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y
^=	x ^= y	x = x ^ y

To be continued…

Interesting sources

https://javascript.plainenglish.io/why-does-javascript-have-both-null-and-undefined-6a42fcca9301

Check deepFreeze function: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/freeze#what_is_shallow_freeze

https://medium.com/@ashfaqueahsan61/time-complexities-of-common-array-operations-in-javascript-c11a6a65a168