

WMNFE 2210

FRONT-END DEVELOPMENT

JavaScript #4 | Arrays





ΠΕΡΙΕΧΟΜΕΝΑ



Περιεχόμενα

- Πίνακες / Arrays
 - Δημιουργία & Ανάθεση
 - Προσπέλαση & Ιδιότητα **length**
 - Χρήσιμες Μέθοδοι



ARRAYS





Arrays

Τι είναι;

Οι πίνακες, στη **JavaScript**, είναι αντικείμενα και μας επιτρέπουν να αποθηκεύσουμε πολλαπλά στοιχεία ίδιου ή και διαφορετικού τύπου.

Τα στοιχεία του πίνακα είναι σε συγκεκριμένη διάταξη, και κάθε στοιχείο του πίνακα έχει συγκεκριμένη θέση, με αρχική τη θέση **0**, στην οποία αναφερόμαστε με τα **[]**, π.χ. **array[0]**, **array[9]**.



Arrays

Σε τι είναι χρήσιμοι;

Αν θέλω να αποθηκεύσω **10** μεταβλητές ίδιου τύπου και με παρόμοια δεδομένα τι θα έκανα;

Αν ήθελα να αποθηκεύσω **1000** ή ακόμα και άγνωστο αριθμό τιμών;

```
let name_1, name_2, name_3, ..., name_100;

const names = new Array();
```



Arrays

Δημιουργία

```
const a = new Array();  
const b = [];  
  
const c = new Array(8, 4, 9, 3, 5, 1);  
const d = [8, 4, 9, 3, 5, 1];
```



Arrays

Δημιουργία

```
// Μπορούν τα στοιχεία του πίνακα να είναι ανομοιογενή  
const a = [1, 'JS', 2.14];  
  
// ή και πίνακες τα ίδια  
const b = [[1, 2], [3, 4]];  
  
// Αλλά δεν θα μας απασχολήσουν ιδιαίτερα
```




Arrays

typeof & Array.isArray()

```
const a = [0, 2, 4, 8];  
  
console.log(typeof a);           // Object  
console.log(Array.isArray(a));   // true
```



Arrays

Προσπέλαση θέσης

```
const things = ['chair', 'watch', 'spoon', 'book'];  
  
console.log(things[0]); // chair  
console.log(things[2]); // spoon  
console.log(things[4]); // error
```



Προσπέλαση θέσης

Arrays

```
const things = ['chair', 'watch', 'spoon', 'book'];

things[0] = 'couch';    // επιτρέπεται σε const!?!
things[4] = 'photo';    // προσθήκη στο τέλος

console.log(things);
// ['couch', 'watch', 'spoon', 'book', 'photo']
```



Arrays

Προσπέλαση μέσω αναφοράς

```
const things = ['chair', 'watch', 'spoon', 'book'];  
  
const new_things = things;  
const new_things[1] = 'clock';  
  
console.log(things);  
// ['chair', 'clock', 'spoon', 'book']
```



Προσπέλαση με επανάληψη

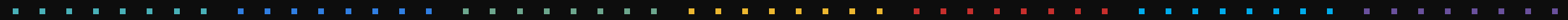
Arrays

```
// In multiple rows
const names = [
  'Albert Einstein',
  'Isaac Newton',
  'Galileo Galilei',
  'Marie Curie',
  'Charles Darwin',
];

for (let i = 0; i < 5; i++) {
  document.write(names[i] + ' <br>');
}
```



CLASSWORK



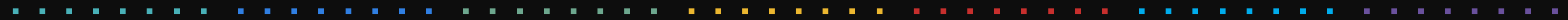


Classwork

Classwork #1

Φτιάξτε ένα **script**, μέσα σε μία σελίδα **HTML** όπου θα έχετε αποθηκευμένα σε μία σταθερά κάποια τυχαία ονόματα.

Στη συνέχεια, με τη χρήση μιας επανάληψης **for** και της **document.write**, δημιουργήστε μια λίστα **** με τα ονόματα αυτά.





Arrays

Ιδιότητα **length**

Με την ιδιότητα **length** μπορώ να ξέρω το "μήκος" του πίνακα, δηλαδή το πλήθος των στοιχείων του.

Όταν ένας πίνακας έχει μήκος **n** τότε το τελευταίο του στοιχείο βρίσκεται στη θέση **n-1**.

```
console.log([].length);           // 0
console.log([1, 2, 3].length);    // 3

const a = [1, 2, 3];

console.log(a[a.length]);         // error
```




Arrays

Ιδιότητα **length**

```
for (let i = 0; i < names.length; i++) {  
  document.write(names[i] + ' <br>');  
}
```



Arrays

Μέθοδος **push**

Με την μέθοδο **push** μπορούμε να προσθέτουμε στοιχεία στο τέλος ενός πίνακα. Θυμίζει τη λειτουργία μίας στοίβας.

```
arr.push(elem1[, elem2, ..., elemN]);
```

```
const things = ['chair', 'watch', 'spoon', 'book'];

things.push('clock');           // Προσθήκη στο τέλος
things[things.length] = 'clock'; // Ισοδύναμα

const a = [];

a.push(1, 2);
console.log(a)                  // [1, 2]
```



Arrays

Μέθοδος **pop**

Με την μέθοδο **pop** μπορούμε να αφαιρούμε στοιχεία από το τέλος ενός πίνακα. Θυμίζει, όπως και η **push**, τη λειτουργία μίας στοίβας.

```
let removed = arr.pop();
```

```
const things = ['chair', 'watch', 'spoon', 'book'];

things.pop();

console.log(things);           // ['chair', 'watch', 'spoon']

let removed = things.pop();    // Αφαίρεση από το τέλος

console.log(removed);          // spoon
console.log(things);           // ['chair', 'watch']
```



Arrays

Μέθοδος **unshift**

Με την μέθοδο **unshift** μπορούμε να προσθέτουμε στοιχεία στην αρχή ενός πίνακα. Θυμίζει τη λειτουργία μίας ουράς.

```
arr.unshift(elem1[, elem2, ..., elemN]);
```

```
const things = ['chair', 'apple', 'spoon', 'book'];

things.unshift('watch');           // Προσθήκη στην αρχή
things[0] = 'watch';               // Ισοδύναμα;

const a = [];

a.unshift(1, 2);
console.log(a);                    // [1, 2]
a.unshift(3);
console.log(a);                    // [3, 1, 2]
```



Arrays

Μέθοδος **shift**

Με την μέθοδο **shift** μπορούμε να αφαιρούμε στοιχεία από την αρχή ενός πίνακα. Θυμίζει τη λειτουργία μίας ουράς.

```
let removed = arr.shift();
```

```
const things = ['chair', 'watch', 'spoon', 'book'];
```

```
things.shift();
```

```
console.log(things); // ['watch', 'spoon', 'book']
```

```
let removed = things.shift(); // Αφαίρεση από την αρχή
```

```
console.log(removed); // watch
```

```
console.log(things); // ['spoon', 'book']
```



Arrays

Μέθοδος **slice**

Η μέθοδος **slice** απομονώνει ένα τμήμα ενός πίνακα και το επιστρέφει, χωρίς να επεμβαίνει στον αρχικό πίνακα.

Δέχεται ως παραμέτρους την αρχή **start** και το τέλος **end**, χωρίς, όμως, η θέση **end** να περιλαμβάνεται στο αποτέλεσμα.

```
let s = arr.slice([start], [end]);
```



Arrays

Μέθοδος **slice**

```
const a = ['a', 'b', 'c', 'd', 'e'];  
  
const b = a.slice(2, 4);  
  
console.log(b);    // ['c', 'd']  
console.log(a);    // ['a', 'b', 'c', 'd', 'e']  
  
// Trick to clone an array  
const c = a.slice();
```



Arrays

Μέθοδος **slice**

```
const phrase = ['I', 'study', 'JS', 'right', 'now'];  
  
// Φέρε μου τα στοιχεία του πίνακα  
// από τη θέση 0 μέχρι τη θέση 2 (3-1)  
let part = phrase.slice(0, 3);  
  
console.log(part);    // ['I', 'study', 'JS']  
console.log(phrase);  // ['I', 'study', 'JS', 'right', 'now']
```




Arrays

Μέθοδος **splice**

Η μέθοδος **splice** μας επιτρέπει να κάνουμε σύνθετες μετατροπές σε έναν πίνακα, είτε αφαιρώντας, είτε αντικαθιστώντας στοιχεία, από οποιοδήποτε σημείο ενός πίνακα.

```
let r = arr.splice(start[, count, elem1, ..., elemN]);
```

```
const things = ['chair', 'watch', 'spoon', 'book'];
```

```
let removed = things.splice(2, 1);
```

```
console.log(removed); // ['watch']
```

```
console.log(things); // ['chair', 'watch', 'book']
```



Arrays

Μέθοδος **splice**

```
const phrase = ['I', 'study', 'JS', 'right', 'now'];

// Αφαίρεσε 3 στοιχεία
// ξεκινώντας από τη θέση 0
// και βάλε στη θέση τους
// τα "Let's", "dance"
phrase.splice(0, 3, "Let's", "dance");

console.log(phrase);
// ["Let's", "dance", "right", "now"]
```



Arrays

Παράδειγμα **slice** & **splice**

```
const a = [1, 2, 3];
const b = a.slice(); // Array copy trick

a.splice(2, 0, 2.5); // Add an item in any place
b[0] = 4;           // a is not affected

console.log(a);      // [1, 2, 2.5, 3]
console.log(b);      // [4, 2, 3]
```



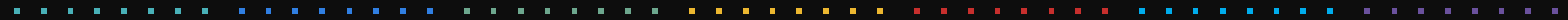
Arrays

Ταξινόμηση - Μέθοδοι **sort** & **reverse**

```
const names = [...];  
  
names.sort();  
console.log(names);  
  
names.reverse();  
console.log(names);
```



CLASSWORK





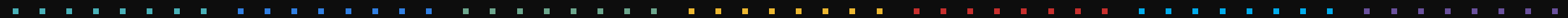
Classwork

Classwork #2

Φτιάξτε ένα **script**, μέσα σε μία σελίδα **HTML**, το οποίο θα ζητά επαναληπτικά ονόματα, ως εξής. Αρχικά εμφανίζεται ερώτηση αν υπάρχει άλλο όνομα για εισαγωγή και αν η απάντηση είναι ναι, τότε να ζητείται το όνομα αυτό.

Τα ονόματα αποθηκεύονται σε πίνακα και η διαδικασία σταματά όταν απαντηθεί αρνητικά η πρώτη ερώτηση.

Στη συνέχεια, και αφού ο πίνακας ταξινομηθεί αλφαβητικά, να δημιουργείται μια λίστα **** με τα ονόματα αυτά.





Εναλλακτικοί τρόποι προσπέλασης

Arrays

```
// Indexes not guaranteed to be in the correct order
for (let index in items) {
  console.log(index, items[index]);
}

// Preferred way for arrays
for (let item of items) {
  console.log(item);
}

// 1337 h4x0r way
items.forEach(console.log);
```



HOMework



Homework

Homework #1

Φτιάξτε ένα script το οποίο κάνει διαδοχικά τις εξής ενέργειες, με τη συγκεκριμένη σειρά:

1. Αρχικοποιεί έναν πίνακα **planets** με τις τιμές **Mars**, **Saturn** και **Uranus**.
2. Προσθέτει στο τέλος του πίνακα τις τιμές **Neptune** και **Pluto**.
3. Προσθέτει στην αρχή του πίνακα τις τιμές **Mercury**, **Venus**, **Earth** και **Earth's Moon**.
4. Παρεμβάλλει την τιμή **Jupiter**, μεταξύ των **Mars** και **Saturn**.
5. Αφαιρεί την τιμή **Earth's Moon**.
6. Αφαιρεί την τιμή **Pluto** :(.
7. Εμφανίζει τα στοιχεία του πίνακα (όπως/όπου θέλετε), είτε με τη χρήση μίας "κλασικής" **for**, είτε με τη χρήση της **for...of**.



Homework

Homework #2

Φτιάξτε ένα script που να ελέγχει αν ένας πίνακας είναι ταξινομημένος!

1. Αρχικά, με τη βοήθεια μίας **while (true)**, ζητάμε επαναλαμβανόμενα στοιχεία από το χρήστη και τα προσθέτουμε με τη σειρά σε έναν πίνακα.
2. Η διαδικασία εισαγωγής σταματά όταν δοθεί για είσοδος το κενό **string** "" (αυτό συμβαίνει όταν ο χρήστης δε γράψει κάτι στο **prompt** και πατήσει το **Enter / OK**).
3. Σε μία νέα επανάληψη **for**, ελέγχουμε κάθε στοιχείο με το επόμενο του. Αν το στοιχείο στη θέση **i** είναι μικρότερο από το στοιχείο στη θέση **i+1**, συνεχίζουμε τον έλεγχο, διαφορετικά εμφανίζεται το μήνυμα **"Not sorted"** και η επανάληψη σταματά.
4. Αν η επανάληψη ολοκληρωθεί, τότε σημαίνει ότι όλοι οι έλεγχοι ήταν επιτυχείς και εμφανίζεται το μήνυμα **"Sorted"**.



Χρήσιμα links

 Array - JavaScript | MDN

[https://developer.mozilla.org/en-US/docs/Web/JavaScript/...](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Arrays)

 Arrays

<https://javascript.info/array>

 Array methods

<https://javascript.info/array-methods>

 JavaScript Arrays

https://www.w3schools.com/js/js_arrays.asp

 splice vs. slice in JavaScript

[https://www.educative.io/edpresso/splice-vs-slice-in-j...](https://www.educative.io/edpresso/splice-vs-slice-in-javascript)



Extra info

📖 javascript - Array() vs new Array() - Stack Overflow
<https://stackoverflow.com/questions/8205691/array-v...>

🔗 Hacks for Creating JavaScript Arrays
<https://www.freecodecamp.org/news/https-medium-...>

📖 Copy an Array in JavaScript - Mastering JS
<https://masteringjs.io/tutorials/fundamentals/copy-array>

📖 for..in versus for..of Loops
<https://bitsofco.de/for-in-vs-for-of/>



THANK YOU!