

Τεχνολογία Διαδικτύου

4. Javascript

Γρηγόρης Τζιάλλας

Καθηγητής

Τμήμα Πληροφορικής και Τηλεπικοινωνιών

Σχολή Θετικών Επιστημών

Πανεπιστήμιο Θεσσαλίας

Στατικές ιστοσελίδες



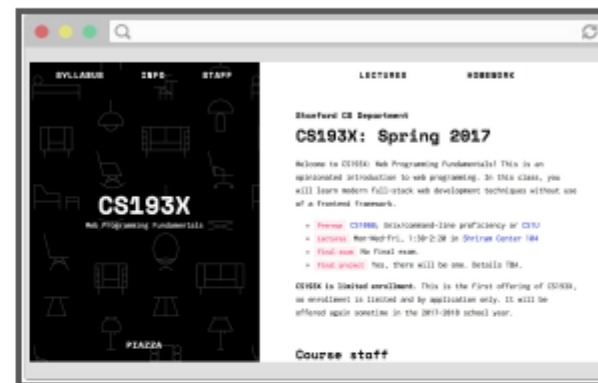
Describes the
content and
structure of
the page

+



Describes the
appearance
and style of
the page

produces



A web page...
that doesn't do
anything

Javascript

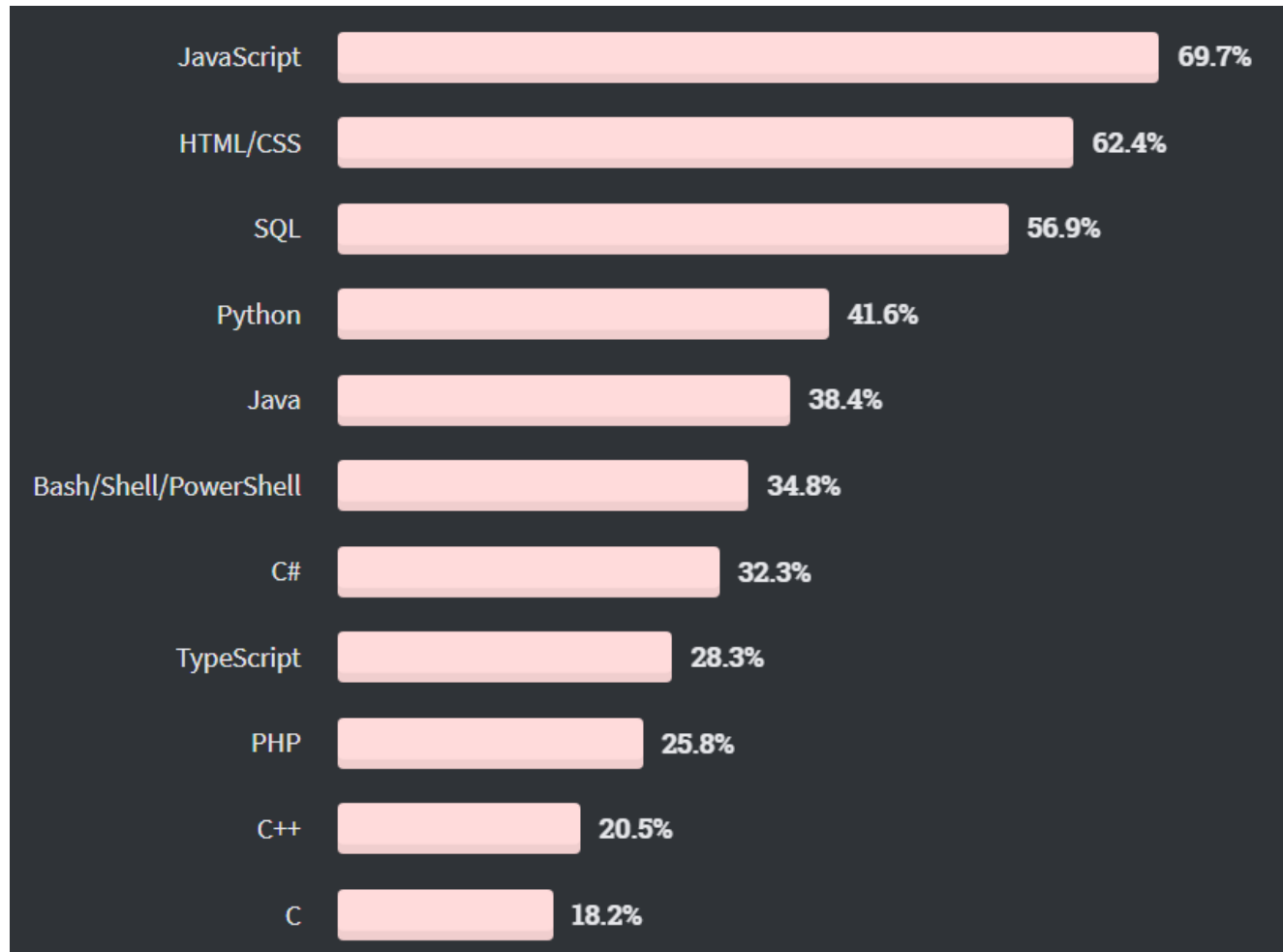


- Δημιουργήθηκε το 1995 από τον Brendan Eich (συνιδρυτή του Mozilla)
- Δεν έχει σχέση με την Java
 - Ονομάστηκε έτσι κυρίως για εμπορικούς λόγους
- Η πρώτη έκδοση της έγινε σε 10 ημέρες !!
- Είναι η μοναδική γλώσσα που εκτελείται από τους φυλλομετρητές και είναι πρακτικά η μοναδική επιλογή για τον προγραμματισμό ιστοσελίδων (γίνονται προσπάθειες να αλλάξει αυτό, πχ. web assembly)

Χαρακτηριστικά της γλώσσας:

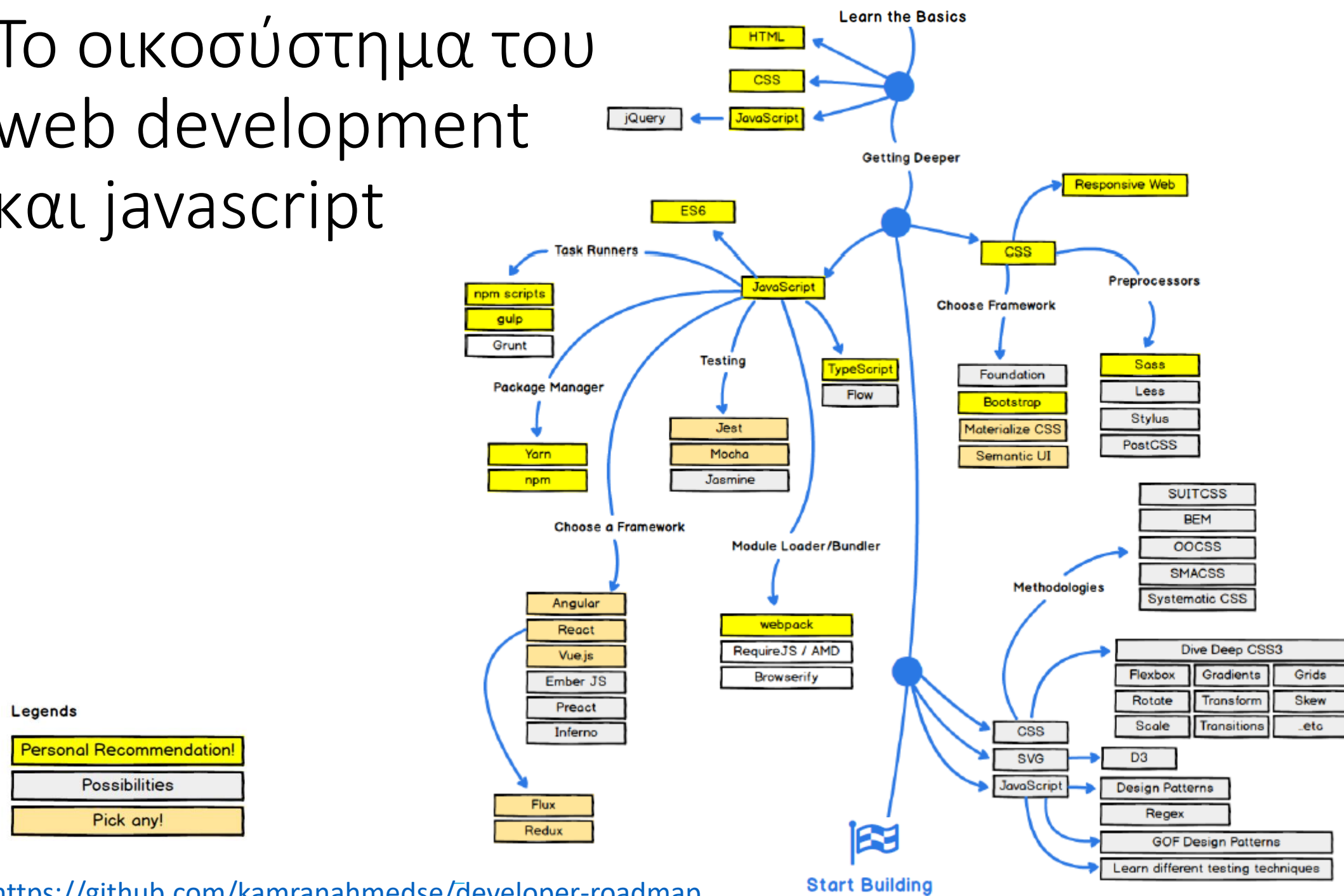
- Υψηλού Επιπέδου
- Διερμηνευόμενη και μεταγλωττιζόμενη δυναμικά (JIT)
- Δυναμική
- Μεταβλητές και συναρτήσεις μπορούν να αλλάξουν ή να δημιουργηθούν νέες οποιαδήποτε στιγμή (κατά το runtime, πχ φόρτωση νέων scripts)

Δημοφιλείς γλώσσες προγραμματισμού



95% των ιστοσελίδων χρησιμοποιούν Javascript

Το οικοσύστημα του web development και javascript



ECMAScript 2015 (ES6)

Νέα χαρακτηριστικά

- Υποστήριξη για κλάσεις
- Iterators και for/of loops
- Promises
 - Fetch API
 - Αντικαθιστά το XMLHttpRequest (AJAX style)
- Let, const αντί για var (εμβέλεια βρόχου)
- Arrow functions (lambdas) =>
- Currying

Σύστημα τύπων γλωσσών προγραμματισμού

- Στατικές γλώσσες προγραμματισμού: Java, C++,...
 - Οι μεταβλητές συνδέονται με ένα συγκεκριμένο τύπο (typed based)
 - Οι έλεγχοι για την ύπαρξη, την ιεραρχία και τον σωστό ορισμό μεθόδων γίνεται κατά την διάρκεια του compile
 - Προκαλούν TypeException στην περίπτωση που συνδεθούν με διαφορετικό τύπο από αυτόν που αρχικοποιήθηκαν
- Δυναμικές γλώσσες προγραμματισμού: JavaScript, python, php,...
 - Οι μεταβλητές δεν συνδέονται με ένα συγκεκριμένο τύπο (untyped based)
 - Οι έλεγχοι για την ύπαρξη, την ιεραρχία και τον σωστό ορισμό μεθόδων γίνεται κατά την διάρκεια του runtime (εκτέλεσης)
 - Μπορούν να συνδεθούν με διαφορετικό τύπο από αυτόν που αρχικοποιήθηκαν

Ενσωμάτωση Javascript σε ιστοσελίδα

- Με την χρήση της ετικέτας `<script>` σε οποιοδήποτε τμήμα της σελίδας μπορούμε να ενσωματώσουμε:
 - Κώδικα javascript
πχ. `<script> alert("Hello") ;</script>`
 - Εξωτερικό αρχείο js
πχ. `<script src="app.js"></script>`
- ή να ενσωματώσουμε κώδικα javascript σε κάποιο συμβάν στοιχείου html
πχ. `<div onclick = "alert("Hello");">Click me!</div>`

Παράδειγμα ιστοσελίδας με javascript

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Hello</title>
  <script>
    alert('Hello');
  </script>
</head>
<body>
  <div>Hello from javascript</div>
</body>
</html>
```

Με την συνάρτηση alert εμφανίζεται ένα παράθυρο διαλόγου με το μήνυμα Hello

Εκτύπωση μηνυμάτων στην κονσόλα

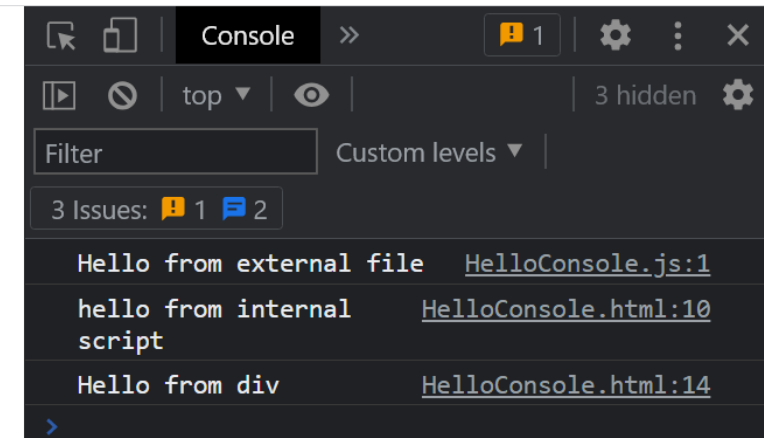
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Hello JS</title>
  <script src="HelloConsole.js"></script>
  <script>
    console.log("hello from internal script");
  </script>
</head>
<body>
  <div onclick="console.log('Hello from div')">Click Me</div>
</body>
```

Η εντολή `console.log` είναι
αντίστοιχη με:
`System.out.println`, `print`,
`printf`, κ.λπ.

Click Me

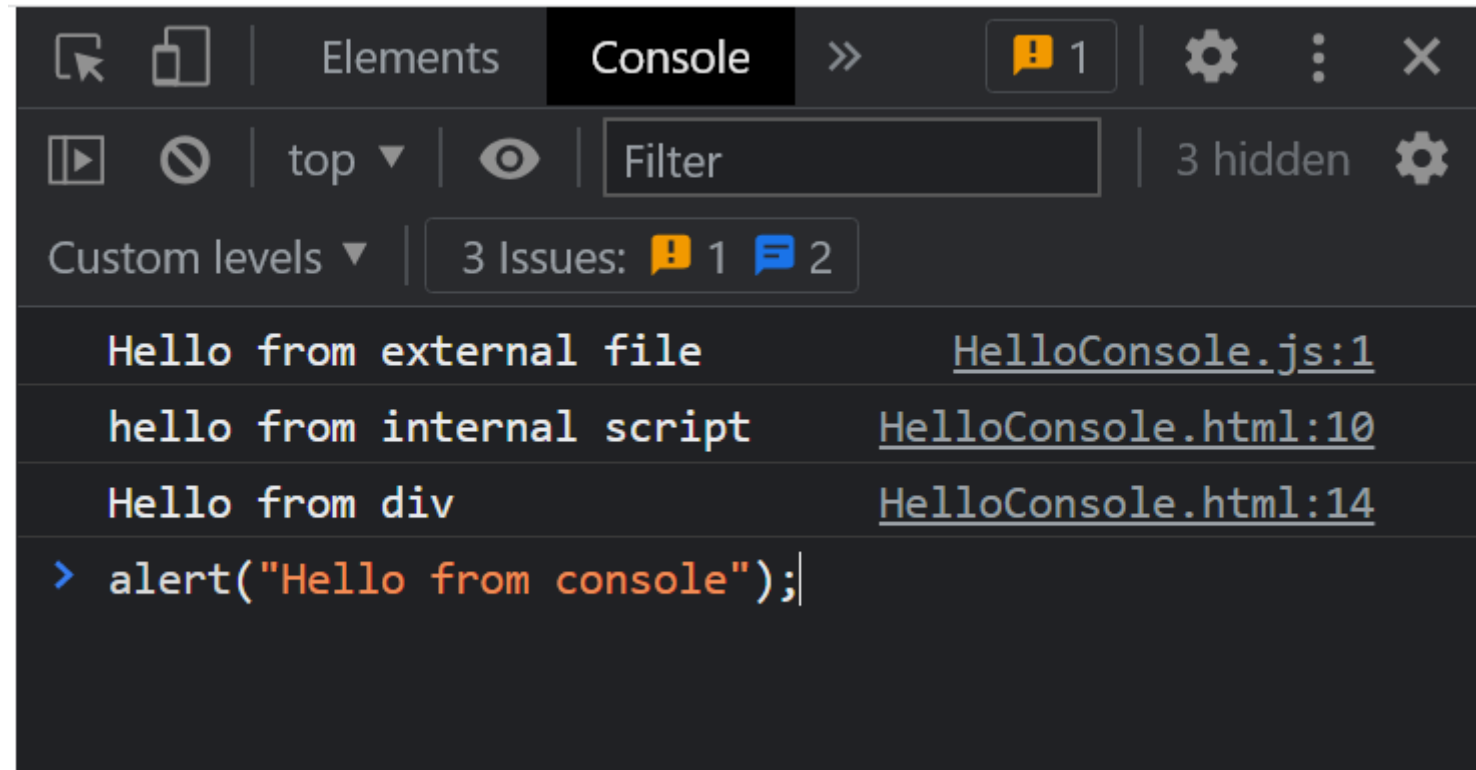
File `helloConsole.js`

```
console.log("Hello from external file");
```



Χρήση της κονσόλας για εκτέλεση εντολών

- Η κονσόλα (Console) μπορεί να χρησιμοποιηθεί και για την εκτέλεση εντολών javascript (interactive language shell)



Εμφάνιση στην ιστοσελίδα

- document.write : Έξοδος στην ιστοσελίδας
- innerHtml : Τροποποιεί το περιεχόμενο ενός στοιχείου HTML

```
<h2 id="output"></h2>
<script>
    document.getElementById("output").innerHTML="Hello";
    document.write("<h3>Hello</h3>");
</script>
```

- Η χρήση τους συνιστάται να γίνεται μόνο για debugging και όχι σε παραγωγικό σύστημα

Εκτέλεση της Javascript

- Δεν υπάρχει κάποια συνάρτηση ή μέθοδος `main`
- Η εκτέλεση ξεκινά από την πρώτη γραμμή και συνεχίζει μέχρι το τέλος
- Η γλώσσα διερμηνεύεται και μεταγλωττίζεται δυναμικά κατά την εκτέλεση (JIT – Just In Time Compilation)

Χαρακτηριστικά της γλώσσας (1)

Η σύνταξη είναι σχεδόν παρόμοια με Java/C++/C/C#

- Εντολές διακλάδωσης (if statements):

```
if (...) {
```

```
    ...
```

```
} else {
```

```
    ...
```

```
}
```

- Εντολή επανάληψης for:

```
for (let i = 0; i < 5; i++) { ... }
```

- Εντολή διακλάδωσης while:

```
while (notFinished) { ... }
```

- σχόλια: // comment or /* comment */

Χαρακτηριστικά της γλώσσας (2)

- Case sensitive
- Τα ονόματα πρέπει να ξεκινούν με :
 - Με χαρακτήρα (A-Z ή a-z)
 - Το σύμβολο του δολλαρίου (\$)
 - Ή την κάτω παύλα (_)
- Χρησιμοποιεί χαρακτήρες Unicode
- Δυναμικοί τύποι μεταβλητών

Δήλωση μεταβλητών: var, let, const

// Δήλωση με εμβέλεια συνάρτησης (Function scope)

var x = 15;

// Δήλωση με εμβέλεια ομάδας εντολών (Block scope)

let fruit = 'banana';

// Δήλωση σταθεράς με εμβέλεια ομάδας εντολών

const isHungry = true;

- Οι μεταβλητές δεν έχουν τύπο. Οι τιμές των μεταβλητών έχουν!
- Η ίδια μεταβλητή μπορεί να συσχετισθεί με διαφορετικούς τύπους τιμών
- Οι παράμετροι συναρτήσεων δεν δηλώνονται

Ορισμός συναρτήσεων

- Ένας από τους τρόπους ορισμού μιας συνάρτησης είναι ο παρακάτω:

```
function name() {  
    statement;  
    statement;  
    ...  
}
```

- Η κλήση μιας συνάρτησης γίνεται με ξεχωριστή εντολή η οποία καλεί την συνάρτηση με την χρήση παρενθέσεων. Η παράληψη των παρενθέσεων δεν καλεί την συνάρτηση αλλά αναφέρεται στην ίδια την συνάρτηση!!!

```
function hello(){  
    document.writeln("Hello from a function");  
}  
hello();
```

Hoisting

- Ο ορισμός μιας συνάρτησης μπορεί να γίνει και μετά την κλήση της όπως φαίνεται στο παρακάτω παράδειγμα:

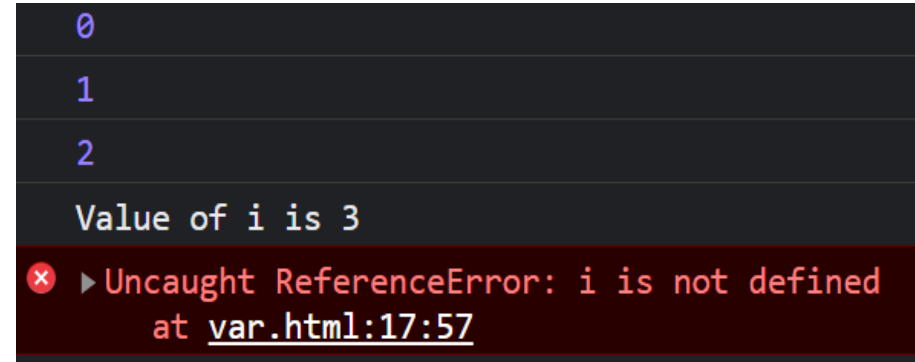
```
hello();  
function hello() {  
    document.writeln('Hello!');  
}
```

- Η Javascript με την τεχνική hoisting διαβάζει πρώτα τους ορισμούς των συναρτήσεων και στην συνέχεια εκτελεί τις εντολές.
- Δεν συνιστάται η χρήση της. Υπάρχουν και άλλοι τρόποι ορισμού συναρτήσεων οι οποίες δεν γίνονται «hoisted»

Η δήλωση μεταβλητής var

- Μια μεταβλητή ή οποία δηλώνεται ως var έχει εμβέλεια στην συνάρτηση την οποία δηλώνεται.
- Στο παρακάτω παράδειγμα το i αν και δηλώνεται στην εντολή επανάληψης for διατηρεί την τιμή του και μετά το τέλος των επαναλήψεων στην συνάρτηση printMessage
- Εκτός τη συνάρτησης η χρήση της μεταβλητής προκαλεί σφάλμα εκτέλεσης

```
function printMessage(message, times) {  
    for (var i = 0; i < times; i++) {  
        console.log(i);  
    }  
    console.log('Value of i is ' + i);  
}  
printMessage('hello', 3);  
console.log('Value of i outside function is ' + i);
```



```
0  
1  
2  
Value of i is 3  
✖ ▶ Uncaught ReferenceError: i is not defined  
   at var.html:17:57
```

Η δήλωση μεταβλητής let

- Μια μεταβλητή ή οποία δηλώνεται με την λέξη κλειδί let έχει εμβέλεια στην ομάδα εντολών (block) στην οποία δηλώνεται.
- Στο παρακάτω παράδειγμα το i έχει ισχύ μόνο στην εντολή επανάληψης.
- Εκτός της εντολής επανάληψης η χρήση του i προκαλεί σφάλμα εκτέλεσης

```
function printMessage(message, times) {  
  for (let i = 0; i < times; i++) {  
    console.log(i);  
  }  
  console.log('Value of i is ' + i);  
}  
printMessage('hello', 3);
```

```
0  
1  
2  
✖ ▶ Uncaught ReferenceError: i is not defined  
   at printMessage (let.html:14:44)  
   at let.html:16:9
```

Η δήλωση μεταβλητής const

- Μια μεταβλητή ή οποία δηλώνεται ως const έχει εμβέλεια στην ομάδα εντολών (block) στην οποία δηλώνεται (όπως και η δήλωση let).
- Μία μεταβλητή που έχει δηλωθεί ως const δεν μπορεί να αλλάξει τιμή, να της ανατεθεί νέα τιμή ή να δηλωθεί εκ νέου στην ίδια ομάδα εντολών.
- Στο παρακάτω παράδειγμα η μεταβλητή k έχει ισχύ μόνο στο τμήμα της if στο οποίο δηλώνεται.
- Η χρήση της k σε άλλο τμήμα προκαλεί σφάλμα εκτέλεσης

```
let x = 45;  
if (x < 50){  
  const k = "Hello";  
  console.log(k);  
}  
console.log(k);
```

Hello

✖ ▶ Uncaught ReferenceError: k is not defined
at const.html:15:21

Συστάσεις για την καλή χρήση των μεταβλητών

- Προτιμήστε την χρήση `const` όπου δεν αλλάζει η τιμή της μεταβλητής
- Χρησιμοποιήστε `let` όταν γίνεται ανάθεση τιμής πολλές φορές στην μεταβλητή
- Αποφύγετε την χρήση `var`

<https://google.github.io/styleguide/jsguide.html#features-use-const-and-let>

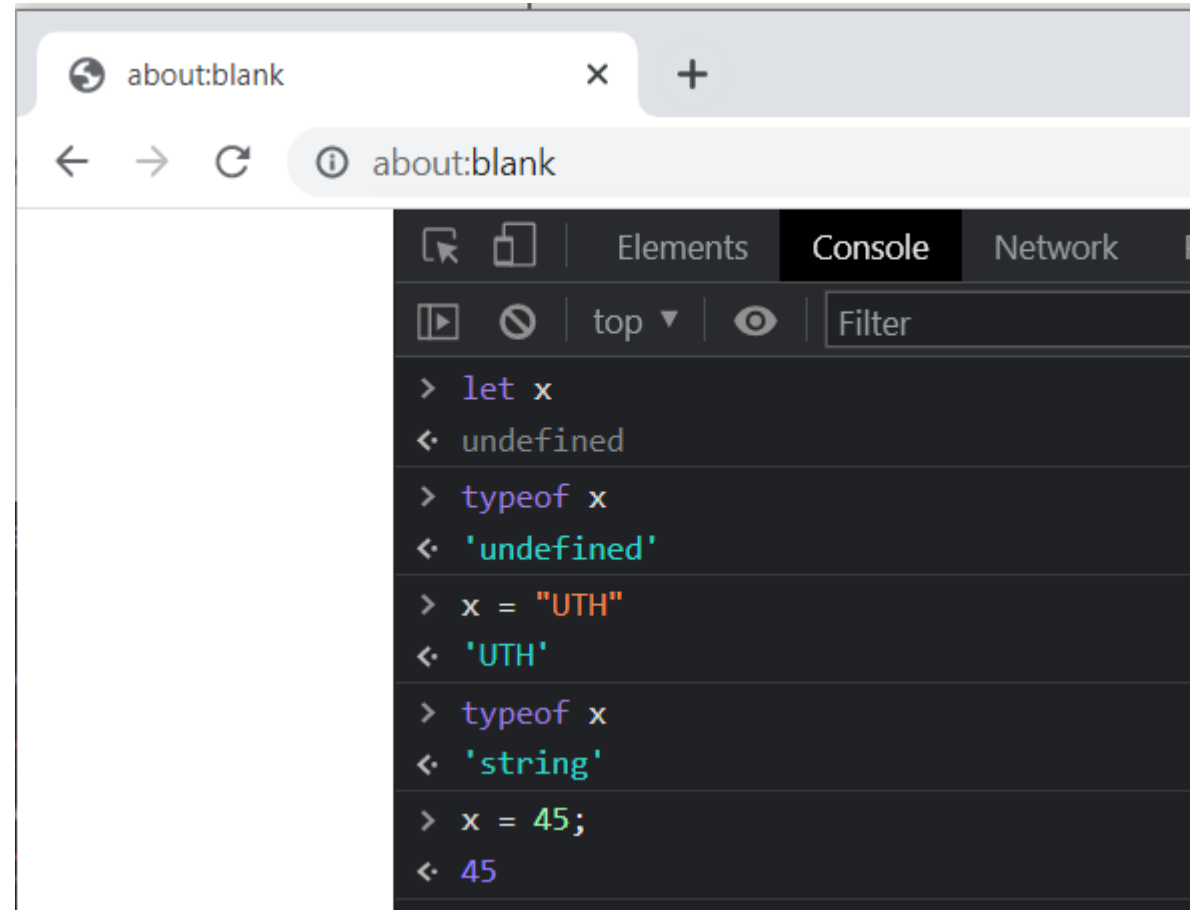
Βασικοί τύποι τιμών δεδομένων

- Οι βασικοί τύποι τιμών δεδομένων (primitives) είναι:
 - Boolean : true ή false
 - Number : όλοι οι αριθμοί είναι τύπου double (δεν υπάρχουν ακέραιοι)
 - BigInt: Μεγάλος ακέραιος
 - String: πχ 'Hello' ή "Hello" (επιτρέπεται η χρήση μονών ή διπλών εισαγωγικών)
 - Symbol
 - Null: Χωρίς τιμή
 - Undefined: Μη καθορισμένη τιμή

Εκτός από τους βασικούς τύπους, όλες οι υπόλοιπες τιμές είναι αντικείμενα όπως Array, Date κ.λπ.

Ο τελεστής typeof

Με την χρήση της κονσόλας του φυλλομετρητή μπορούμε να ορίσουμε μεταβλητές, να τις αναθέσουμε τιμές και να ελέγχουμε τον τύπο τους με τον τελεστή `typeof`



The screenshot shows a web browser window with the address bar displaying 'about:blank'. The developer console is open, showing the following code and output:

```
> let x
< undefined
> typeof x
< 'undefined'
> x = "UTH"
< 'UTH'
> typeof x
< 'string'
> x = 45;
< 45
```


Χρήση αριθμών

- Όλοι οι αριθμοί παριστάνονται ως float (δεν υπάρχουν ακέραιοι).
- Παρόμοιοι τελεστές με Java και C++.
- Προτεραιότητα πράξεων όπως στην Java ή C++.
- Ειδικές τιμές: NaN (not-a-number), +Infinity, -Infinity
- Κλάση Math: Math.floor, Math.ceil, Math.round, κ.λπ.

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Math

```
const project = 6;  
const final = 7;  
const grade =  
    project * 0.3 + final * 0.7;  
console.log("Βαθμός ", Math.round(grade,2));
```

Τελεστές πράξεων

Τελεστής	Περιγραφή
+	Πρόσθεση
-	Αφαίρεση
*	Πολλαπλασιασμός
**	Ύψωση σε δύναμη (ES2016)
/	Διαίρεση
%	Υπόλοιπο ακέραιας διαίρεσης
++	Αύξηση κατά ένα
--	Μείωση κατά ένα

Τελεστές ανάθεσης

Τελεστής	Παράδειγμα	Παρόμοιο με
=	$x = y$	$x = y$
+=	$x += y$	$x = x + y$
-=	$x -= y$	$x = x - y$
*=	$x *= y$	$x = x * y$
/=	$x /= y$	$x = x / y$
%=	$x \% = y$	$x = x \% y$
**=	$x ** = y$	$x = x ** y$

Οι τελεστές της πρόσθεσης μπορούν να χρησιμοποιηθούν και για την σύνδεση συμβολοσειρών

Τελεστές σύγκρισης

Τελεστής	Περιγραφή
==	Ίσο (με μετατροπή τύπου αν χρειάζεται)
===	Ίσο (με ίδια τιμή και τύπο)
!=	Διάφορο (με μετατροπή τύπου αν χρειάζεται)
!==	Διάφορο (τιμή ή τύπος)
>	Μεγαλύτερο
<	Μικρότερο
>=	Μεγάλυτερο ή ίσο
<=	Μικρότερο ή ίσο
?	Τριαδικός τελεστές

Τελεστές σύγκρισης == και !=

- Οι τελεστές == and != δεν λειτουργούν όπως αναμένεται: γίνεται πρώτα μετατροπή των τιμών, πριν τη σύγκριση.

```
' ' == '0' // false
' ' == 0 // true
0 == '0' // true
NaN == NaN // false
[' '] == ' ' // true
false == undefined // false
false == null // false
null == undefined // true
```

Τελεστές σύγκρισης === και !==

- Συνίσταται η χρήση των === και !== αντί των == και !=

```
' ' === '0' // false
' ' === 0 // false
0 === '0' // false
NaN === NaN // still weirdly false
[''] === '' // false
false === undefined // false
false === null // false
null === undefined // false
```

Booleans

- Τιμές true ή false
- Τελεστές: && (σύζευξη), || (διάζευξη), !(άρνηση)
- null, undefined, 0, NaN, "", "" όταν χρησιμοποιούνται σε συγκρίσεις και εντολές ελέγχου μετατρέπονται στην τιμή false

Παράδειγμα ελέγχου ορισμού τιμής

```
let x;  
if (x){  
    console.log(x);  
}  
else{  
    console.log("x is undefined");  
}
```

x is undefined



Strings - Συμβολοσειρές

- Χρήση μονών ή διπλών εισαγωγικών
- Δεν αλλάζουν - immutable
- Δεν ορίζεται τύπος χαρακτήρα, είναι string με μήκος 1
- Χρήση τελεστή πρόσθεση για την συνένωση τους
- Έχουν την ιδιότητα length που αναφέρει το μήκος τους
 - (προσοχή: δεν είναι συνάρτηση)
- Με την χρήση template strings <https://css-tricks.com/template-literals/> μπορούμε να ορίσουμε συμβολοσειρές που καταλαμβάνουν πολλαπλές γραμμές και ενσωματώνουν μεταβλητές ή και εκφράσεις.

```
let age = 32;  
let name = "nick";  
let message = `I am ${name} and my age is ${age}`;
```

Backtick χαρακτήρας

Τελεστές, ιδιότητες και μέθοδοι συμβολοσειρών

- Τελεστές: +, =, +=
- Ιδιότητα: length επιστρέφει το μήκος μιας συμβολοσειράς
- Μέθοδοι:

Μέθοδος	Περιγραφή
charAt(position)	επιστρέφει τον χαρακτήρα μιας θέσης
charCodeAt(position)	επιστρέφει τον κωδικό του χαρακτήρα μιας θέσης
indexOf(), lastIndexOf()	επιστρέφουν την θέση συγκεκριμένου κειμένου σε μια συμβολοσειρά
toUpperCase()	μετατρέπει και επιστέφει την συμβολοσειρά σε κεφαλαία
toLowerCase()	μετατρέπει και επιστέφει την συμβολοσειρά σε πεζά
concat()	συνενώνει δύο ή περισσότερες συμβολοσειρές
trim()	αφαιρεί τα κενά της αρχής και τέλους της συμβολοσειράς
slice()	εξάγει και επιστρέφει ένα τμήμα μιας συμβολοσειράς
substring()	παρόμοια με slice() με την διαφορά είναι ότι δεν μπορεί να δεχθεί αρνητικούς δείκτες.
replace()	αντικαθιστά ένα τμήμα της συμβολοσειράς
split()	επιστροφή τον πίνακα που προκύπτει από την κατάτμηση της συμβολοσειράς με συγκεκριμένο κείμενο (delimiter)

Παράδειγμα μεθόδων string (1)

```
let s = "Τμήμα Πληροφορικής";  
//char at 0  
let r1 = s.charAt(0);  
document.write("<div>char at 0 = " + r1 + "</div>");  
//charCodeAt at 0  
let r2 = s.charCodeAt(0);  
document.write("<div>charCodeAt at 0 = " + r2 + "</div>");  
//indexOf  
let r3 = s.indexOf("ρ");  
document.write("<div>index of 'ρ' = " + r3 + "</div>");  
//lastIndexOf  
let r4 = s.lastIndexOf("ρ");  
document.write("<div>lastIndex of 'ρ' = " + r4 + "</div>");  
let r5 = s.toUpperCase();  
//toUpperCase  
document.write("<div>toUpperCase = " + r5 + "</div>");  
let r6 = s.toLowerCase();  
//toLowerCase  
document.write("<div>toLowerCase = " + r6 + "</div>");
```

char at 0 = T

charCodeAt at 0 = 932

index of 'ρ' = 9

lastIndex of 'ρ' = 13

toUpperCase = ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

toLowerCase = τμήμα πληροφορικής

Παράδειγμα μεθόδων string (1)

slice from 2 to 8 = ήμα Πλ
slice reverse from -8 to -2 = οφορικ
replace = Τμήμα Πληροφορικής και Τηλεπικοινωνιών
split with space = Τμήμα,Πληροφορικής

```
//slice
let r7 = s.slice(2,8)
document.write("<div>slice from 2 to 8 = " + r7 + "</div>");
//slice
let r8 = s.slice(-8, -2)
document.write("<div>slice reverse from -8 to -2 = " + r8 + "</div>");
//replace
let r9 = s.replace("κής", "κής και Τηλεπικοινωνιών")
document.write("<div>replace = " + r9 + "</div>");
//split with delimiter space
let r10 = s.split(" ")
document.write("<div>split with space = " + r10 + "</div>");
```

Χρήση Template Strings

I am nick and my age is 32

Τμήμα Πληροφορικές και Τηλεπικοινωνιών

Τμήμα Πληροφορικές και Τηλεπικοινωνιών

```
//Template strings
let age = 32;
let name = "nick";
let message = `I am ${name} and my age is ${age}`;
document.write(message);
//multiline template string
let x = `Τμήμα Πληροφορικές
και Τηλεπικοινωνιών`;
//use string concatenation to print string with variable
document.writeln('<h2>' + x + "</h2>");
//use template string
document.writeln(`<h2> ${x} </h2>`);
```

Εντολές ελέγχου (1)

- if .. else
 if (condition)
 {statements1;}
 else
 {statements2;}

• switch
 switch (expression) {
 case label1:
 statements1;
 break;
 case label2:
 statements2;
 break;
 default:
 statements;
 }

```
let k = 45;  
document.write("<h2>Έλεγχος περιττού</h2>");  
if (k % 2 === 0) {  
    document.write("<p>Άρτιος</p>");  
}  
else {  
    document.write("<p>Περιττός</p>");  
}
```

```
document.write("<h2>Έλεγχος με switch</h2>");  
let v = 1;  
switch (v) {  
    case 1:  
        document.write("<p>Ένα</p>");  
        break;  
    case 2:  
        document.write("<p>Δύο</p>");  
        break;  
    default:  
        document.write("<p>Άλλος αριθμός</p>")  
}
```

Εντολές ελέγχου (2)

- for

for (initial; condition; incr)
{statements;}

- while

while (condition){
 statements;
}

do {
 statements;
}
while (condition);

Έξοδος από βρόγχο με continue και break.

```
document.write("<h2>Εντολή for</h2>");  
for (let i = 1; i < 15; i++) {  
    if (i % 2 === 0) {  
        continue; //skip even numbers  
    }  
    document.write(i + " ");  
}
```

```
document.write("<h2>Εντολή while</h2>");  
let x = 0;  
while (x < 15) {  
    if (x % 2 !== 0) {  
        document.write(x + " ");  
    }  
    x++;  
}
```

```
document.write("<h2>Εντολή do while</h2>");  
let n = 0, sum = 0;  
do {  
    n++;  
    sum += n;  
    if (sum > 10) break; // Exit from while  
    document.write(sum + " ");  
}  
while (true) //loop unconditionally
```

Πίνακες / Λίστες

- Οι πίνακες / λίστες είναι αντικείμενα της Javascript τα οποία χρησιμοποιούν δείκτες για την αρίθμηση των στοιχείων τους
- Η αρίθμηση τους ξεκινά από το 0 (0-based indexing)
- Μπορούν να τροποποιηθούν (Mutable)
- Η ιδιότητα length (όχι συνάρτηση) επιστρέφει το μήκος τους

Μέθοδοι πινάκων

Μέθοδος	Περιγραφή
push()	Προσθέτει ένα στοιχείο στο τέλος ενός πίνακα και επιστρέφει το νέο μέγεθος του πίνακα
pop()	Αφαιρεί το τελευταίο στοιχείο από ένα πίνακα και επιστρέφει το διαγραφμένο στοιχείο
shift()	Αφαιρεί το πρώτο στοιχείο από ένα πίνακα και επιστρέφει το διαγραφμένο στοιχείο
unshift()	Προσθέτει ένα στοιχείο στην αρχή ενός πίνακα και επιστρέφει το νέο μέγεθος του πίνακα
reverse()	Αντιστρέφει τη σειρά των στοιχείων ενός πίνακα (το πρώτο τελευταίο, το δεύτερο προ-τελευταίο, κ.ο.κ.)
sort()	Ταξινομεί αλφαβητικά τα στοιχεία ενός πίνακα
join()	Δημιουργεί μια συμβολοσειρά που περιέχει όλα τα στοιχεία ενός πίνακα, χωρίζοντάς τα με ένα διαχωριστικό
concat()	Συνενώνει τα στοιχεία 2 ή περισσότερων πινάκων σε ένα νέο πίνακα
slice()	Αντιγράφει στοιχεία του πίνακα
splice()	Προσθέτει ή αφαιρεί στοιχεία από τον πίνακα

Πίνακες - Παράδειγμα

```
let m=["11",1,2,3,"45","nick"];  
console.log(m); //prints array to console  
console.log(m[0]); //prints first item  
m.push("newItem"); //adds a new item to the end  
console.log(m[m.length-1]); //prints last item  
let y = m.shift(); //retrieves and removes first item  
console.log("Poped item ", y); //prints removed item  
console.log( m); //prints final array
```

```
► (6) ['11', 1, 2, 3, '45', 'nick']
```

```
11
```

```
newItem
```

```
Poped item 11
```

```
► (6) [1, 2, 3, '45', 'nick', 'newItem']
```

Παράδειγμα slice και splice

```
Initial list: ▶ (5) [2, 4, 6, 7, 8]
list after add with splice: ▶ (6) [2, 3, 4, 6, 7, 8]
List after remove with splice: ▶ (4) [4, 6, 7, 8]
Removed items: ▶ (2) [2, 3]
List after slice: ▶ (4) [4, 6, 7, 8]
Copied items: ▶ [4]
```

```
let list=[2,4,6,7,8];
console.log("Initial list: ", list);
list.splice(1,0,3); //add element at index 1
console.log("list after add with splice: ", list);
let removed = list.splice(0,2); //remove first two elements
console.log("List after remove with splice: ", list);
console.log("Removed items: ",removed);
let copied = list.slice(0,1); //copies first element
console.log("List after slice: ", list);
console.log("Copied items: ",copied);
```

Εντολές επανάληψης σε πίνακες

```
let l = ["a", "d", "f"];  
for (let i = 0; i < l.length; i++) {  
  console.log(l[i]);  
}
```



For .. με δείκτες του πίνακα

```
for (let anItem of l) {  
  console.log(anItem);  
}
```



For .. of .. για το κάθε στοιχείο του πίνακα

```
let k = [];  
for (let i = 0; i < 5; i++) {  
  k[i] = 2 * i + 1;  
}
```



For .. για την προσθήκη στοιχείων σε πίνακα

```
//Returns values  
console.log("for of loop");  
for (let anItem of k) {  
  console.log(anItem);  
}
```



For .. of .. για τα στοιχεία του πίνακα

```
//Returns keys - indexes  
console.log("for in loop");  
for (let anItem in k) {  
  console.log(anItem);  
}
```



For .. in .. η οποία επιστέφει τους δείκτες του πίνακα

a
d
f
a
d
f
for of loop
1
3
5
7
9
for in loop
0
1
2
3
4

Τα αντικείμενα ως συλλογές ιδιοτήτων

- Στην Javascript ένα αντικείμενο (object) είναι μια συλλογή ιδιοτήτων (key-value pairs)

```
const classes = {};  
const vathmoi = {  
  'giannis': 9,  
  'nikos': 8,  
  'anna': 7.5  
};  
console.log(vathmoi['anna']); // 7.5
```

Καθολικό αντικείμενο (Global object) σε εφαρμογές front- end στους φυλλομετρητές είναι το **window**. Στο nodejs (σε εφαρμογές back end) καθολικό αντικείμενο είναι το **global**.

Δημιουργία αντικειμένων και προσθήκη ιδιοτήτων

```
//Object creation
const giannis={};
giannis.email ="giannis@uth.gr";

//Object creation with new
const nikos = new Object;
nikos["email"]="nikos@uth.gr";

//Object creation with JSON Notation
const anna={
  email:"anna@uth.gr",
  semester: 5,
  vathmos: 7
}
console.log(giannis, nikos, anna);
```

Ορισμός ονόματος ιδιότητας email με dot notation

Ορισμός ονόματος ιδιότητας email με κείμενο. Ο τρόπος αυτός μπορεί να χρησιμοποιηθεί και με μεταβλητή.

Ορισμός αντικειμένου και των ιδιοτήτων του με JSON (Javascript Object Notation)

```
▶ {email: 'giannis@uth.gr'}
▶ {email: 'nikos@uth.gr'}
▶ {email: 'anna@uth.gr', semester: 5, vathmos: 7}
```

Διαγραφή ιδιοτήτων και επανάληψη

```
//delete object property
delete anna.semester;
console.log(anna);

//for in loop fro object properties
console.log("Properties of Anna");
for (let aProperty in anna){
    console.log(aProperty, anna[aProperty]);
}
```

```
► {email: 'anna@uth.gr', vathmos: 7}
```

```
Properties of Anna
```

```
email anna@uth.gr
```

```
vathmos 7
```

- Η επανάληψη for .. in χρησιμοποιείται σε objects.
- Δεν μπορεί να χρησιμοποιηθεί for...of σε objects. Μόνο σε λίστες.

Αναφορές

- <https://developer.mozilla.org/en-US/docs/Learn/JavaScript>
- <https://www.w3schools.com/js/default.asp>
- https://www.w3schools.com/js/js_examples.asp
- <https://www.freecodecamp.org/learn/javascript-algorithms-and-data-structures/#basic-javascript>
- Video Tutorial for CSS
 - https://www.youtube.com/watch?v=OXGznpKZ_sA&ab_channel=freeCodeCamp.org
 - https://www.youtube.com/watch?v=1Rs2ND1ryYc&ab_channel=freeCodeCamp.org
- CSS How to / Examples
 - https://www.w3schools.com/css/css_examples.asp