Τεχνολογίες Διαδικτύου

Εισαγωγή στη javascript
Μεταβλητές και Συναρτήσεις

Καθηγητής Χρήστος Δουληγέρης
Δρ. Ρόζας Μαυροπόδη

Δυναμικές Ιστοσελίδες

Μία ιστοσελίδα είναι δυναμική όταν:

- αλληλεπιδρά με το χρήστη (π.χ. αλλάζει η εμφάνιση ενός μενού επιλογών όταν ο δείκτης του ποντικιού τοποθετείται πάνω σε αυτό),
- αλλάζει η μορφή της (π.χ. μετακινούνται λέξεις και αντικείμενα ή αλλάζουν δυναμικά εικόνες, γράμματα και χρώματα),
- μεταβάλλεται το περιεχόμενό της (π.χ. αλλάζουν τα περιεχόμενα ενός πίνακα).

Δυναμικές Ιστοσελίδες

- Η HTML επιτρέπει τη δημιουργία στατικών ιστοσελίδων, δηλαδή ιστοσελίδων των οποίων το περιεχόμενο και η μορφή παραμένουν σταθερά. Παρέχει περιορισμένη επικοινωνία με τους χρήστες, η οποία βασίζεται στη δυνατότητα μετάβασης από τη μία ιστοσελίδα στην άλλη μέσω των συνδέσμων.

- Περισσότερες δυνατότητες αλληλεπίδρασης με τους χρήστες υλοποιούνται με τη χρήση επιπρόσθετων τεχνολογιών όπως:

  - Γλώσσες σεναρίων (scripting languages), όπως η **JavaScript**, οι οποίες επιτρέπουν τη διασύνδεση διαφορετικών αντικειμένων και τεχνολογιών στην ίδια ιστοσελίδα, τεχνολογίες όπως AJAX και JQuery (που βασίζονται στη JavaScript). Λειτουργούν στη πλευρά του φυλλομετρητή
  - Σενάρια (scripts) CGI (c++, perl), γλώσσες σεναρίων όπως **PHP** και τεχνολογίες όπως node.js (που βασίζεται στη JavaScript). Λειτουργούν στη πλευρά του εξυπηρετητή Ιστού (Web server).

Τί είναι;

Είναι μία γλώσσα σεναρίων (χαλαρή με απλή δομή και σύνταξη), της οποίας ο κώδικάς ενσωματώνεται σε αυτόν της HTML και τρέχει στην πλευρά του πελάτη (φυλλομετρητή).

Η **JavaScript** δεν έχει σχέση με τη **Java.**

1995: Στη Netscape, Brendan Eich δημιούργησε τη "JavaScript".

1996: Microsoft δημιουργεί "JScript", για τον IE3.

1997: JavaScript καθιερώθηκε ως πρότυπο "ECMAScript"

(www.ecma-international.org).

2005: "AJAX" δημιουργήθηκε και η εποχή του web 2.0 ξεκινά. (interactive web)

2006: jQuery 1.0 δημιουργήθηκε.

2010: Node.JS δημοσιεύτηκε.

2017: ECMAScript 8, aka ECMAScript 2017 οριστικοποιήθηκε τον Ιούνη του 2017.

Πώς μπορώ να τη χρησιμοποιήσω;

- ✓ Δημιουργία αυτοδύναμων εφαρμογών, καθώς είναι ενσωματωμένη στην εφαρμογή του φυλλομετρητή.

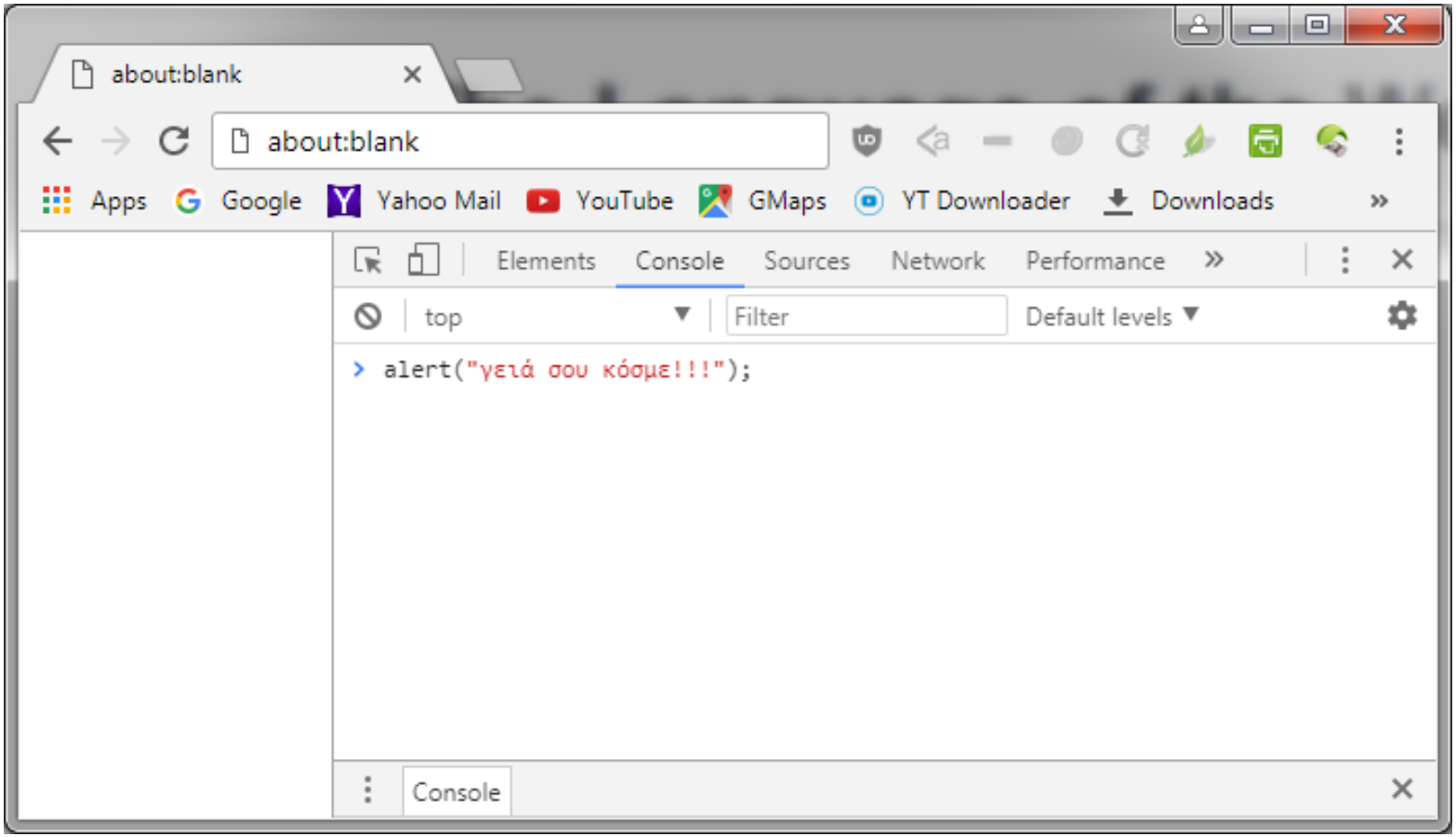
Compiled vs Interpreted

Πρώτα μεταγλώττιση μετά εκτέλεση

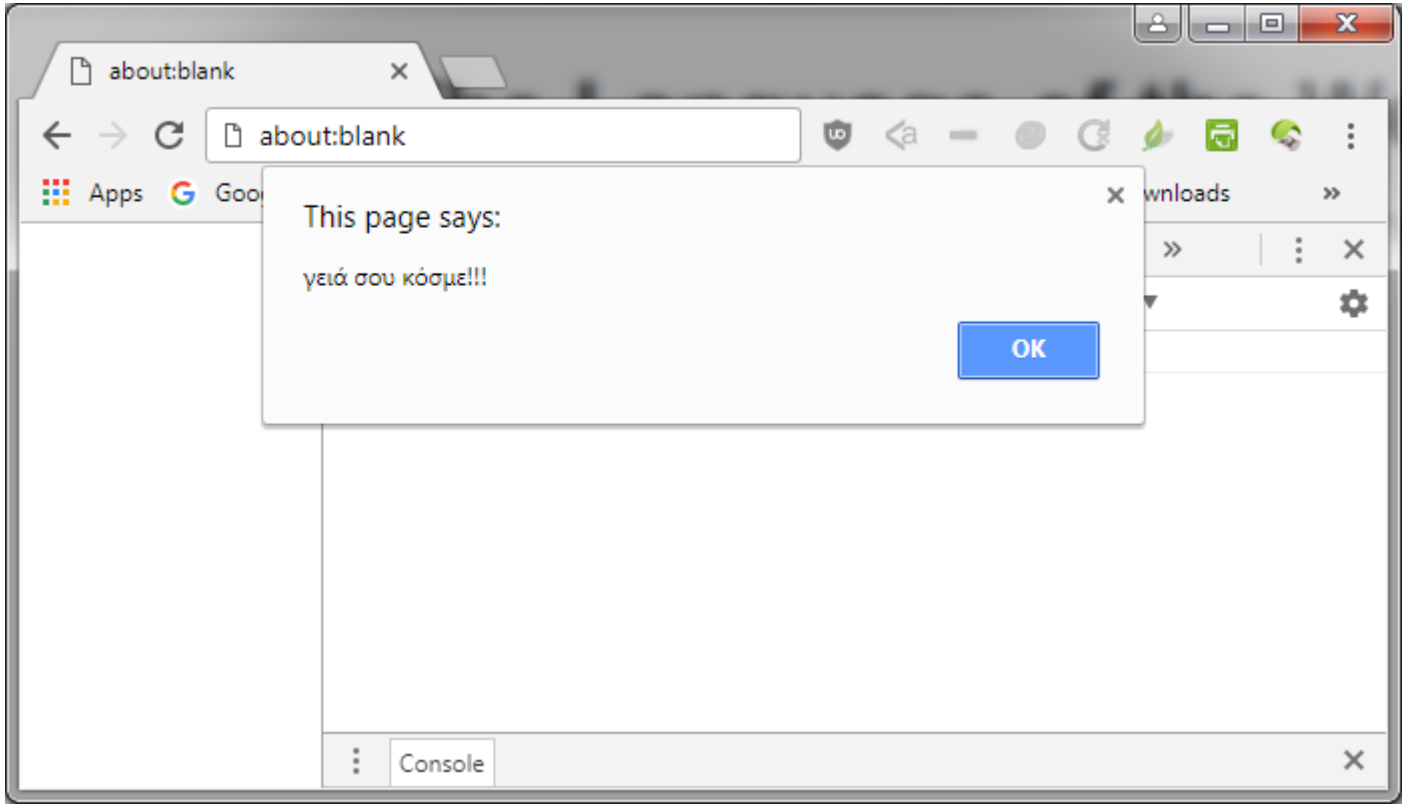
Ταυτόχρονη μεταγλώττιση και εκτέλεση

- ✓ Σε συνδυασμό με κάποια ιστοσελίδα.

Ενσωματωμένη στο φυλομετρητή



Ενσωματωμένη στο φυλομετρητή



Σε συνδυασμό με κάποια ιστοσελίδα;

Τι μπορεί να γίνει με τη χρήση javascript.

- Παρακολουθεί τα γεγονότα που συμβαίνουν σε μία ιστοσελίδα.
- Διαχειρίζεται στοιχεία και αντικείμενα της σελίδας HTML.
- Χρησιμοποιείται για τον έλεγχο της ορθότητας των στοιχείων που εισάγει ο χρήστης σε μία φόρμα.
- Μπορεί να ελέγξει τον τύπο και την έκδοση του φυλλομετρητή του χρήστη.
- Μπορεί να χρησιμοποιηθεί για τη δημιουργία cookies – αποθήκευση και ανάκτηση πληροφορίας από τον υπολογιστή του χρήστη.
- Μπορεί να χρησιμοποιηθεί για τη διαχείριση συνόδων (sessions).
- Σχεδιασμό (<canvas>) και animation

Τοποθέτηση στην ιστοσελίδα

Οι εντολές JavaScript που θα εκτελέσει ο φυλλομετρητής περιλαμβάνονται στο head ή στο body της σελίδας html. Περιέχονται στο στοιχείο <script>, π.χ.

```
<script>
...εδώ γράφουμε τις εντολές του Script...
</script>
```

Σε ένα έγγραφο HTML μπορούν να υπάρχουν περισσότερα του ενός σενάρια, σε περισσότερα του ενός μέρη.

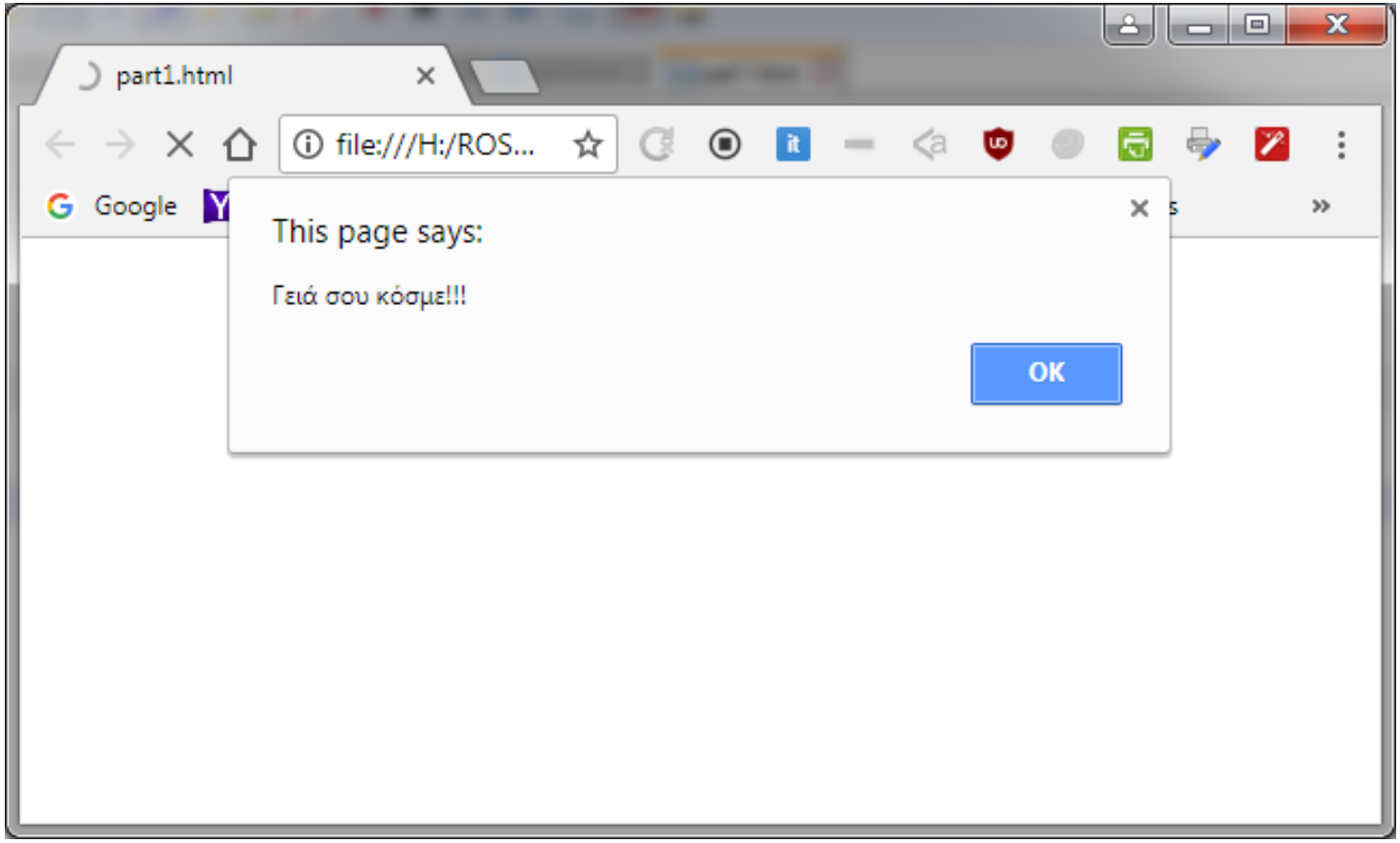
Τοποθέτηση στην ιστοσελίδα - head

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<script>
     alert ("Γειά σου κόσμε!!!");
</script>
</head>
<body> … </body>
</html>
```

Τοποθέτηση στην ιστοσελίδα - body

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
</head>
<body>
<script>
      alert ("Γειά σου κόσμε!!!");
</script>
</body>
</html>
```

Τοποθέτηση στην ιστοσελίδα - αποτέλεσμα



Τοποθέτηση στην ιστοσελίδα – head vs body

Εξαρτάται την εφαρμογή, π.χ. εάν χρειάζεται να συμβεί κάτι πριν εμφανιστεί ολόκληρη η σελίδα στον επισκέπτη, τότε η θέση είναι στο head.

Σε διαφορετική περίπτωση η θέση είναι στο body.

Γενικά η τοποθέτηση σεναρίου της Javascript στο τέλος του body, δηλαδή πριν το </body>, βελτιώνει την ταχύτητα φόρτωσης της ιστοσελίδας, καθώς η επεξεργασία του από τον φυλλομετρητή καθυστερεί την εμφάνιση της ιστοσελίδας.

Τοποθέτηση στην ιστοσελίδα – σε εξωτερικό αρχείο

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
</head>
<body>
<script src="myScript.js">
     </script>
</body>
</html>
```

/*περιεχόμενα του αρχείου myScript.js */

```
alert ("Γειά σου κόσμε!!!");
```

Τοποθέτηση στην ιστοσελίδα – σε εξωτερικό αρχείο

Πολλαπλά αρχεία:

```
<script src="myScript1.js"></script>
<script src="myScript2.js"></script>
```

Απομακρυσμένη κλήση:

```
<script
src="https://www.unipi.com/js/myScript
1.js"></script>
```

Τοποθέτηση στην ιστοσελίδα – σε εξωτερικό αρχείο

Πλεονεκτήματα:

- ✓ Διαχωρίζεται ο κώδικας από τις ετικέτες της HTML
- ✓ Η λογική της εφαρμογής παρακολουθείται και συντηρείται καλύτερα
- ✓ Τα συχνά χρησιμοποιούμενα αρχεία javascript μπορεί να αποθηκευτούν στην προσωρινή μνήμη (casched) και έτσι κάθε μελλοντική επίσκεψη στην ιστοσελίδα να πραγματοποιηθεί γρηγορότερα

Χαρακτηριστικά δομής της JavaScript (1/2)

- Υπάρχει διαφορά ανάμεσα στα **πεζά** και τα **κεφαλαία** γράμματα (case sensitive).
- Υπάρχει **ευελιξία (χαλαρότητα) στις δηλώσεις των μεταβλητών**. Οι δηλώσεις μπορούν να καταλαμβάνουν περισσότερες από μία γραμμές, αλλά και σε μία γραμμή μπορούν να τοποθετούνται περισσότερες από μία δηλώσεις.
- Ανάμεσα στις διαδοχικές δηλώσεις μιάς γραμμής πρέπει απαραιτήτως να μπαίνει ερωτηματικό.

- Τα άγκιστρα (**{** και **}**) χρησιμοποιούνται για να ομαδοποιούν εντολές, όπως το να διαχωρίζουν τις εντολές μίας συνάρτησης, τις εντολές ενός βρόχου for ή while.

Χαρακτηριστικά δομής της JavaScript - Παράδειγμα

- ALERT (“Γειά σου κόσμε"); vs alert (“Γειά σου κόσμε");
- temp="Γειά σου κόσμε"; alert (temp); Αλλά και αυτό είναι σωστό:

```
temp="Γειά σου κόσμε"
alert (temp)
```

- function sayMyName(name) {

```
    console.log('Hi, ' + name);
    alert('Hi, ' + name);
}
```

- temp="Γειά σου κόσμε";

Χαρακτηριστικά δομής της JavaScript (2/2)

Οι εφαρμογές JavaScript απαρτίζονται από:

- Δηλώσεις οι οποίες ορίζουν τις ενέργειες που θα εκτελέσει ο φυλλομετρητής

```
alert (“Hello”);
```

- δηλώσεις συναρτήσεων

```
function sayMyName(name) {
        alert('Hi, ' + name);
}
```

- αναφορές σε αυτές τις συναρτήσεις (κλήσεις συναρτήσεων)

```
sayMyName(“Roza”) ;
```

Μεταβλητές

Χρησιμοποιούμε της μεταβλητές για την αποθήκευση πληροφοριών. Διάφοροι τρόποι ορισμού και αρχικοποίησης.

- Ορισμός, και στη συνέχεια αρχικοποίηση σε δυο διαφορετικές δηλώσεις:

```
var x;
x = 5;
```

- Ορισμός και αρχικοποίηση σε μια δήλωση:

```
var y = 2;
```

- Ορισμός χωρίς το var δηλαδή όπως παρακάτω:

```
my_number = value;
```

- Οι τιμές που παίρνουν οι μεταβλητές μπορούν να αλλάξουν κατά τη διάρκεια του προγράμματος.

  Επανάθεση τιμών μεταβλητής:

```
var x = 5;
x = 1;
```

Μεταβλητές – ονοματολογία

Τα ονόματα των μεταβλητών μπορούν να

- ✓ αρχίζουν με γράμμα ή τα σύμβολα: $, _
- ✓ περιέχουν μόνο γράμματα, αριθμούς ή τα σύμβολα $, _

- Υπάρχει διάκριση πεζών κεφαλαίων

- Αποφυγή δεσμευμένων λέξεων (break, class, switch)

- Επιλογή ονόματος σύμφωνα με τη χρήση τους

- Επιλογή ονοματολογίας (camelCase) για πολλαπλές λέξεις ως: multipleWords και όχι multiple_words

- Μείνετε σταθεροί στην επιλογή του τρόπου ονοματολογίας

- ✓ Σωστό:
  var numPeople, $mainHeader, _num, _Num;
- ✓ Λάθος:

  var 2coolForSchool, soHappy!

Μεταβλητές – τύποι

- **string:** σειρά από γράμματα, αριθμούς και σύμβολα:

```
var greeting = 'Hello Kitty';
var restaurant = "Pamela's Place";
```

- **number:** ακέραιοι (6, -102) δεκαδικοί (5.8737):

```
var myAge = 28;
var pi = 3.14;
```

- **boolean:** δέχονται λογικές τιμές true ή false:

```
var catsAreBest = true;
var dogsRule = false;
```

- **undefined:** αντιπροσωπεύει μια μεταβλητή η οποία έχει οριστεί αλλά όχι αρχικοποιηθεί, δεν έχει οριστεί τιμή.

```
var notDefinedYet;
```

- **null:** αντιπροσωπεύει μια μεταβλητή η οποία έχει οριστεί και αρχικοποιηθεί με την τιμή null

```
var goodPickupLines = null;
```

Μεταβλητές – αλφαριθμητικά (string)

Μια μεταβλητή string κρατά τη θέση των χαρακτήρων:

```
var alphabet="abcdefghijklmnopqrstuvwxyz";
```

Η ιδιότητα length παρέχει το πλήθος των χαρακτήρων:

```
console.log(alphabet.length); // 26
```

Κάθε χαρακτήρας έχει ένα αναγνωριστικό της θέσης του.

Ο πρώτος χαρακτήρας είναι πάντα στη θέση 0.

Ο τελευταίος στη length-1:

```
console.log(alphabet[0]); // 'a'
console.log(alphabet[1]); // 'b'
console.log(alphabet[2]); // 'c'
console.log(alphabet[alphabet.length]); // undefined
console.log(alphabet[alphabet.length-1]); // 'z'
console.log(alphabet[alphabet.length-2]); // 'y'
```

Μεταβλητές – αποτελέσματα πράξεων

Μπορεί να περιέχουν αποτελέσματα αριθμητικών πράξεων:

```
var x = 2 + 2;
var y = x * 3;
var name = 'Claire';
var greeting = 'Hello ' + name;
var title = 'Baroness';
var formalGreeting = greeting + ', ' + title;
```

Ευελιξία δηλώσεων - Loose typing

Ορισμός μεταβλητών στην JavaScript (loose typing)

```
var a = 13; // Number
var b = "thirteen"; // String
```

Ορισμός μεταβλητών σε γλώσσα με (strong typing)

```
int a = 13; // int
String b = "thirteen"; // String
```

Ευελιξία δηλώσεων - Loose typing

Μετατροπή σε string.

```
7 + 7 + 7; // = 21
7 + 7 + "7"; // = 147
"7" + 7 + 7; // = 777
```

Εφόσον αντί για + υπάρχει κάτι άλλο π.χ. * ή / τότε συμβαίνει το αντίθετο:

```
"35" - 3; // = 32
30 / "3"; // = 10
"3" * 3; // = 9
```

Αυτή η ‘σιωπηρή’ μετατροπή εξαρτάται τόσο από τον τελεστή όσο και από τη θέση των ορισμάτων.

Ευελιξία δηλώσεων - Loose typing

Ένας τύπος μεταβλητής

```
var y = 2 + "string"
console.log(typeof y);
```

Σχόλια

Τα σχόλια αποτελούν κείμενο το οποίο διαβάζεται από τον άνθρωπο και αγνοείται από τον υπολογιστή

```
// Σχόλιο μιας ολόκληρης γραμμής
    var x = 4; // Σχόλια μετά τη δήλωση

    /*
    Σχόλια τα οποία επεκτείνονται σε
πολλαπλές γραμμές.
    */
```

Τελεστές πράξεων στη Javascript

- Μαθηματικοί τελεστές (+, -, *, /, %).

Το (%) επιστρέφει το υπόλοιπο μιας διαίρεσης. (π.χ. 10%3 επιστρέφει 1)

- Τελεστές αυτόματης αύξησης (++) και αυτόματης μείωσης (--), οι οποίοι αυξάνουν ή μειώνουν αντίστοιχα την τιμή μιας μεταβλητής κατά 1.

Έστω ότι έχουμε τη μεταβλητή x, το **x++** θα είναι ισοδύναμο με **x+1** ενώ το **x--** με **x-1**.

- Εκχώρηση τιμής γίνεται με τον τελεστή “=” (π.χ. το x=2 δίνει στη μεταβλητή x την τιμή 2).
- Σύνθετοι τελεστές ανάθεσης (+=, -=, *=, /=, %=), όπου το **x+=y** είναι ισοδύναμο με **x=x+y**, το **x-=y** με **x=x-y**, κ.λ.π.

Τελεστές σύγκρισης στη JavaScript

Τελεστής	Επεξήγηση	Παράδειγμα
= =	Ελέγχει εάν δύο τιμές είναι ίσες	Το 5=='5' επιστρέφει true
= = =	Ελέγχει εάν δύο τιμές είναι ίσες και του ίδιου τύπου	Το 5===‘5’ επιστρέφει false
!=	Ελέγχει εάν δύο τιμές είναι διαφορετικές	Το 5!=2 επιστρέφει true
>	Ελέγχει εάν μία τιμή είναι μεγαλύτερη από μία άλλη	Το 5>2 επιστρέφει true
>=	Ελέγχει εάν μία τιμή είναι μεγαλύτερη ή ίση από μία άλλη	Το 5>=2 επιστρέφει true
<	Ελέγχει εάν μία τιμή είναι μικρότερη από μία άλλη	Το 5<2 επιστρέφει false
<=	Ελέγχει εάν μία τιμή είναι μικρότερη ή ίση από μία άλλη	Το 5<=2 επιστρέφει false

Τελεστές σύγκρισης: =, ==, ===, στη JavaScript

Χρειάζεται προσοχή στη χρήση των τελεστών: =, = =, ===.

Εκχώρηση τιμής: θέσε τη μεταβλητή a=1;

Σύγκριση ισότητας τιμής: είναι η μεταβλητή a==1;

Σύγκριση ισότητας τιμής αλλά και τύπου: είναι η μεταβλητή αριθμός και είναι a= = =1;

Για παράδειγμα

"1"==1 θα επιτύχει ενώ το

"1"===1 θα εποτύχει.

Λογικοί τελεστές στη JavaScript

Τελεστής	Επεξήγηση	Παράδειγμα
&&	και (and)	Εάν x=2 και y=3 το (x==2 && y>1) επιστρέφει true
||	ή (or)	Εάν x=2 και y=3 το (x<2 || y>1) επιστρέφει true
!	όχι (not)	Εάν x=2 το !(x= =3) επιστρέφει true

Συναρτήσεις: Ορισμός – έτοιμες συναρτήσεις

Οι συναρτήσεις περιέχουν τμήματα κώδικα τα οποία μπορούν να ξαναχρησιμοποιηθούν σε διάφορα σημεία της εφαρμογής.

Η JavaScript υποστηρίζει δικές της συναρτήσεις / μεθόδους όπως την **alert, confirm**

```
confirm ("Είναι καλή μέρα;");
```

Συναρτήσεις: Ορισμός από τον χρήστη

Υπάρχουν και συναρτήσεις τις οποίες δημιουργεί ο προγραμματιστής. Ορίζονται με το λεκτικό `function`.

Χρησιμοποιούν `{ }` ώστε να καθορίσουν τα όρια τους/ περικλύσουν τον κώδικά τους.

Ορισμός:

```
function sayMyName() {
    console.log('Η Javascript είναι πρώτη!');
}
```

Κλήση / χρήση:

```
sayMyName();
```

Συναρτήσεις: παράμετροι (1/?)

Οι παράμετροι αποτελούν τον τρόπο με τον οποίο παρέχονται πληροφορίες/ δεδομένα στη συνάρτηση.

```
function sayMyName(name) {
        console.log('Γειά σου, ' + name);
    }

    sayMyName('Ρόζα');
    sayMyName('Γιώργο Καραγιώργο');
```

Συναρτήσεις: παράμετροι (2/?)

Οι συναρτήσεις μπορεί να έχουν μία ή περισσότερες παραμέτρους, οι οποίες χωρίζονται με κόμμα.

```
function addNumbers(num1, num2) {
        var result = num1 + num2;
        console.log(result);
    }
```

Συναρτήσεις: παράμετροι (3/?)

Κατά την κλήση μιας συνάρτησης μπορεί να χρησιμοποιηθούν τιμές ή μεταβλητές ως ορίσματα..

```
addNumbers(7, 21);
addNumbers(3, 10);
```

```
var number = 10;
addNumbers(number, 2);
addNumbers(number, 4);
```

Συναρτήσεις: παράμετροι ή ορίσματα

Το όρισμα είναι μια τιμή που παρέχεται στη συνάρτηση και

παράμετρος είναι η μεταβλητή εντός της συνάρτησης στην οποία εκχωρείται η τιμή αυτή.

Χρησιμοποιούνται πανομοιότυπα.

Συναρτήσεις: κυκλική κλήση συναρτήσεων

```
function chicken() {
        egg();
    }

    function egg() {
        chicken();
    }

    egg();
```

Συναρτήσεις: επιστροφή τιμής (1/?)

Υπάρχουν δύο είδη συναρτήσεων. Οι συναρτήσεις που εκτελούν ενέργειες χωρίς να επιστρέφουν τιμή και οι συναρτήσεις που επιστρέφουν κάποια τιμή.

Οι τελευταίες στον ορισμό τους χρησιμοποιούν το λεκτικό `return`.

```
function addNumbers(num1, num2) {
      var result = num1 + num2;
      return result; /* οτιδήποτε μετά αυτή τη  γραμμή
                        κώδικα θα αγνοηθεί */
}

var sum = addNumbers(5, 2);
console.log(sum);
```

Συναρτήσεις: Κλήση / χρήση

Μια συνάρτηση μπορεί να κληθεί ως εξής:

ανάλογα τον ορισμό της: χωρίς ορίσματα, με ορίσματα.

Ανάλογα τον τύπο των ορισμάτων: τα ορίσματα μπορεί να είναι τιμές ή μεταβλητές.

Με κάποιον από τους παρακάτω τρόπους:

Ως τμήμα μιας πράξης/ δήλωσης:

```
var biggerSum = addNumbers(2, 5) +
addNumbers(3, 2);
```

Ως όρισμα άλλων συναρτήσεων:

```
var hugeSum = addNumbers(addNumbers(5,
2), addNumbers(3, 7));
```

Εμβέλεια μεταβλητών

- Όταν μία μεταβλητή δηλωθεί μέσα σε μία συνάρτηση ονομάζεται **τοπική** μεταβλητή και έχει ισχύ μόνο μέσα στη συνάρτηση αυτή. Όταν ολοκληρωθεί η εκτέλεση της συνάρτησης η μεταβλητή αυτή καταστρέφεται. Επομένως μπορούμε σε διαφορετικές συναρτήσεις υπάρχουν τοπικές μεταβλητές με το ίδιο όνομα.
- Εάν μία μεταβλητή δηλωθεί εκτός των συναρτήσεων της JavaScript μπορεί να κληθεί και να χρησιμοποιηθεί από όλες τις συναρτήσεις (**καθολικές**).

Σημαντικό: Η μεταβλητή ξεκινάει να έχει ισχύ από τη στιγμή που δηλώνεται (δηλαδή από τη στιγμή που φορτώνεται η σελίδα HTML) και σταματάει να έχει ισχύ όταν κλείσει η σελίδα.

Εμβέλεια μεταβλητών: τοπικές μεταβλητές

```
function addNumbers(num1, num2) {
      var localResult = num1 + num2;
      console.log("The local result
is: " + localResult);
    }
    addNumbers(5, 7);
    console.log(localResult);
```

Εμβέλεια μεταβλητών: καθολικές μεταβλητές

```
var globalResult;
function addNumbers(num1, num2) {
    globalResult = num1 + num2;
    console.log("The global result is: " +
globalResult);
    }
addNumbers(5, 7);
console.log(globalResult);
```

Συναρτήσεις παράδειγμα

- Δημιουργήστε μια συνάρτηση η οποία μετατρέπει τη θερμοκρασία από την κλίμακα Celsius σε Fahrenheit. Δίνεται ότι ο τύπος F=C*9/5 +32

  Η συνάρτηση ονομάζεται celsiusToFahrenheit:
  - Αποθηκεύστε τη θερμοκρασία σε μια μεταβλητή.
  - Μετατρέψτε το σε Fahrenheit και η έξοδος θα εμφανίζει μήνυμα όπως: "NN ° C είναι NN ° F".
- Δημιουργήστε μια συνάρτηση η οποία ονομάζεται fahrenheitToCelsius:
  - Τώρα αποθηκεύστε τη θερμοκρασία του Fahrenheit σε μια μεταβλητή.
  - Μετατρέψτε το σε celsius και η έξοδος εμφανίζει μήνυμα όπως "NN ° F είναι NN ° C.

Συναρτήσεις παράδειγμα

```
function celsiusToFahrenheit(celsius) {
        var celsiusInF = (celsius*9)/5 + 32;
        console.log(celsius + '°C is ' + celsiusInF + '°F ');
}

function fahrenheitToCelsius(fahrenheit) {
 var fahrenheitInC = ((fahrenheit - 32)*5)/9;
console.log(fahrenheit + '°F is ' + fahrenheitInC + '°C’);
}
```