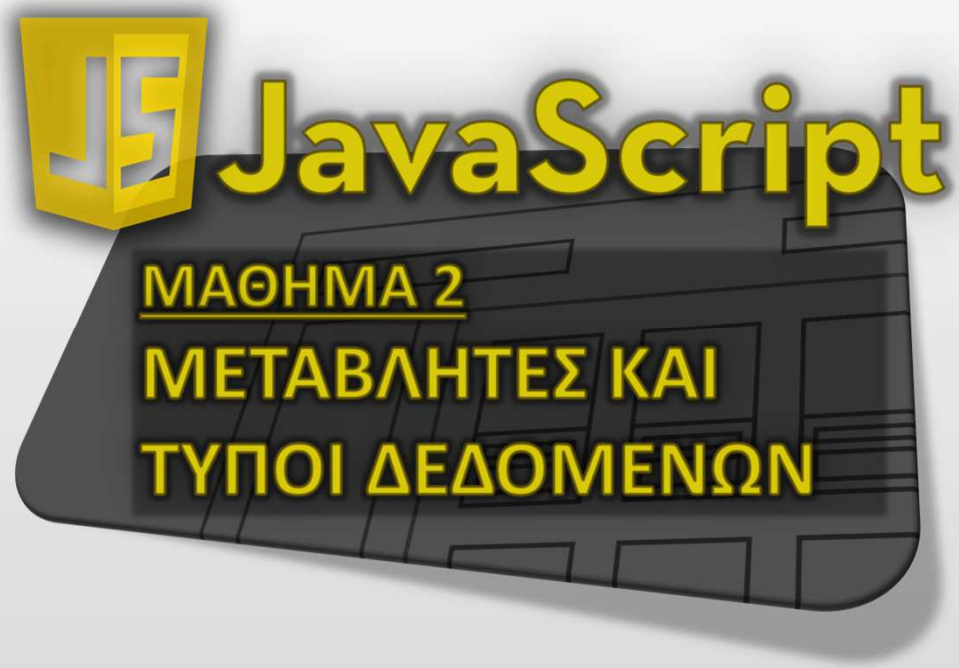


ΠΕΡΙΕΧΟΜΕΝΑ:

Νικόλαος Βασιλειάδης

Σμαραγδένιος Χορηγός Μαθήματος

ΜΑΘΗΜΑ 2: Μεταβλητές και Σταθερές

1. Τιμές και Ονόματα

JavaScript psounis

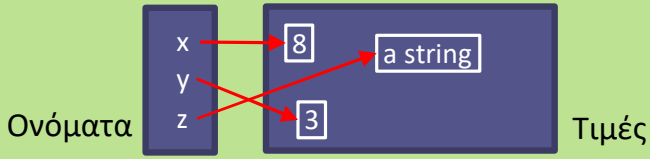
- Στη JS αποθηκεύουμε **τιμές (values)**. Μία τιμή:
  - Είτε ανήκει σε έναν από τους **7 πρωτογενείς τύπους δεδομένων (primitive data types)**:
    - Αριθμοί (**number**) και μεγάλοι ακέραιοι (**bigint**)
    - Συμβολοσειρές (**string**)
    - Λογικές Μεταβλητές (**boolean**)
    - Τους ειδικούς τύπους-τιμές: **null** και **undefined**
    - Σύμβολα (**symbol**), προχωρημένο, εισήχθη στην ES6
  - Είτε είναι **αντικείμενο(object)**
- Η τιμή έχει **έναν χώρο αποθήκευσης στη μνήμη**
- Μπορούμε να δώσουμε ένα **όνομα (name)** σε μία τιμή:
  - Καλό θα είναι να είναι περιγραφικό (όχι ξερά x και y)
  - Ακολουθείται το «camelCase» (όχι κενά, πρώτη λέξη να ξεκινά με μικρό και επόμενες με κεφαλαίο)
  - Ξεκινά με χαρακτήρα (ή τα $,_) και ακολουθούν χαρακτήρες, τα $,_ και αριθμητικά ψηφία.
  - Δεν πρέπει να είναι λέξη-κλειδί (π.χ. for)
  - Είναι case-sensitive
- Ονοματοδοσία:
  - Ένα όνομα πρέπει να δηλωθεί, πριν χρησιμοποιηθεί. Γράφουμε:
    - **let όνομα;** ή
    - **let όνομα = τιμή;**
  - Για να θέσουμε όνομα σε μία τιμή, γράφουμε
    - **όνομα = τιμή** (έκχώρηση, καταχώρηση, ανάθεση)

Παράδειγμα 1: values.html

```
let x;
let y = 3;
x=8;
console.log(x);
let z;
z=1;
console.log(z);
z="a string";
console.log(z);
```

Παρατήρηση:

- Για μία εντολή τύπου: let x = 3;
  - Αντί να λέμε «Δίνουμε στην τιμή 3, το **όνομα** x»
  - Θα λέμε «Αναθέτουμε στη **μεταβλητή** x την τιμή 3»
- Προσοχή όμως (ιδίως αν υπάρχει εμπειρία σε C/C++/Java)
  - Τα ονόματα (μεταβλητές) δεν έχουν τύπο δεδομένων
  - Έτσι μπορούμε να αναθέσουμε το ίδιο όνομα, πρώτα σε ακέραια τιμή, έπειτα σε συμβολοσειρά κ.ο.κ.)
- Σχηματική Αναπαράσταση Μνήμης (βλ. και βίντεο):



- Λέμε ότι οι τύποι της JavaScript καθορίζονται δυναμικά (**dynamically typed language**)

ΜΑΘΗΜΑ 2: Μεταβλητές και Τύποι Δεδομένων

1.1. Αριθμοί

JavaScript psounis

Η JS έχει έναν κοινό τύπο δεδομένων για αριθμούς (ακέραιους ή πραγματικούς)

- Έτσι **ο τύπος δεδομένων αριθμός (number)**:
  - Ακολουθεί το πρότυπο IEEE754 που αποθηκεύει πραγματικούς αριθμούς σε 64bit (το ίδιο ακριβώς με τους double στις C/C++/Java)
  - ως **ακέραιος** μπορεί να αποθηκεύσει αριθμούς από το -2^{53} έως το $2^{53}-1$ (~ **$9x10^{15}$**)
  - ως **πραγματικός**, αποθηκεύει σωστά **16-17 ψηφία** με εύρος ακρίβειας από $4.9x10^{-324}$ έως $1.8x10^{308}$

Παράδειγμα 2: number.html

```
let f1, f2;
f1 = 15;
f2 = 132.939;
console.log(f1*f2);
```

Πρόσθετα, μπορούμε να ορίσουμε έναν αριθμό χρησιμοποιώντας και τις εξής, όχι τόσο συνηθισμένες, **μορφές** (βλ. παράδειγμα 3)

Παράδειγμα 3: number_literals.html

```
let x = 0xFF; //hexadecimal
let y = 0o77; //octal
let z = 0b111; //binary
console.log(x, y, z);
```

```
let a = 5.115e2; //scientific
let b = 5.115E2; //scientific
let c = 100_0000.000_001; //separators
console.log(a, b, c);
```

Ειδικές τιμές (μέσω καθολικών σταθερών)

- Δεν υπάρχει υπερχείλιση. Το αποτέλεσμα είναι **Infinity**
- Δεν υπάρχει υποχειλιση. Το αποτέλεσμα είναι **-Infinity**
- Δεν προκαλεί σφάλμα π.χ. το 0/0. Το αποτέλεσμα είναι **NaN** (not a number)

Παράδειγμα 4: number_global_constants.html

```
console.log(-1/0);
console.log(0/0);
console.log(Math.pow(10,309));
console.log(Infinity+Infinity)
```

Ο τύπος δεδομένων μεγάλος ακέραιος (bigint):

- Αναπαριστά μεγάλους ακεραίους, με πρακτικά απεριοριστο πλήθος ψηφίων.
- Εισήχθηκε στην ES2020
- Τα literals αναπαρίστανται όπως οι ακέραιοι, αλλά πρέπει να ακολουθούνται από «n», π.χ. 15n, 28n
- (Καλό θα είναι να μην μπλέκονται με τους συνήθεις ακεραίους σε παραστάσεις)

Παράδειγμα 5: bigint.html

```
let x=15123345342341221 23n;
console.log(x*x);
console.log(x*x*x);
```

ΜΑΘΗΜΑ 2: Μεταβλητές και Τύποι Δεδομένων 1.2. Συμβολοσειρές και Λογικές Μετ/τές

JavaScript psounis

Συμβολοσειρές (string):

- Μία συμβολοσειρά ορίζεται σε διπλά (ή μονά) εισαγωγικά:
  - Μπορούμε να έχουμε μονά εισαγωγικά, μέσα σε διπλά εισαγωγικά (και το αντίστροφο)
  - Με το + μπορούμε να συνενώσουμε συμβολοσειρές.

Παράδειγμα 6: string.html

```
let s1 = "Hello";
let s2 = 'World';
let s = s1 + " '" + s2 + "' ";
console.log(s);
console.log(s+s1);
console.log(s+s2);
```

- Οι συμβολοσειρές έχουν πλούσια λειτουργικότητα που θα μελετήσουμε αναλυτικά σε επόμενο μάθημα. Εισαγωγικά:
  - όνομα.length: Μήκος συμβολοσειράς
  - όνομα[i]: Χαρακτήρας στη θέση i
  - Είναι immutable (δεν τροποποιείται)
- **typeof:** Μονοθέσιος προθεματικός τελεστής που επιστρέφει τον τύπο δεδομένων μίας τιμής.

Παράδειγμα 7: string2.html

```
let s = "Hello World";
console.log(s.length);
console.log(s[0], s[s.length-1], typeof s[0]);
// s[0]="a" //error
```

Λογικές Μεταβλητές (boolean):

- Αναπαριστούν τις λογικές τιμές αλήθεια/ψέματα:
  - Αλήθεια: Αναπαρίσταται ως **true**
  - Ψέματα: Αναπαρίσταται ως **false**

Παρατήρηση:

- Σε λογικους ελέγχους (π.χ. στη συνθήκη της if) οι τιμές:
  - 0, undefined, null, NaN και η κενή συμβολοσειρά “”, ερμηνεύονται ως false
  - Αναφέρονται ως **falsy**
- Κάθε άλλη τιμή ερμηνεύεται ως true
  - και αναφέρονται ως **truthy**

Παράδειγμα 8: boolean.html

```
let x = true;
x = false;
console.log(x, typeof(x))
if (NaN)
  console.log("NaN");
```

Άσκηση 1: exercise01.html

Παρατηρήστε ότι ο τελεστής + είναι υπερφορτωμένος:

- Αποθηκεύστε μία συμβολοσειρά (έστω s)
- Αποθηκεύστε έναν ακέραιο (έστω n)
- Υπολογίστε το n+s και τυπώστε το

Κωδικοποίηση:

- Η JS χρησιμοποιεί UTF-16.
- Άρα ένας χαρακτήρας θα πιάνει 1 ή 2 bytes στη μνήμη.
- Χρησιμοποιούμε χαρακτήρες διαφυγής για την ενσωμάτωση σε συμβολοσειρές:

Χαρακτήρας Διαφυγής	Επεξήγηση
\xNN	N: 16-αδικό ψηφίο
\uNNNN	N: 16-αδικό ψηφίο
\u{NNNNNN}	N: 16-αδικό ψηφίο

Παράδειγμα 9: unicode.html

```
let symbols = "\u{1f923}😄\u27FE"
document.write(symbols);
```

🤣😄⟾

Άλλοι Χαρακτήρες Διαφυγής (Escape Characters):

Χαρακτήρας Διαφυγής	Επεξήγηση
\t	Tab
\n	Newline
\'	Μονό Εισαγωγικό
\"	Διπλό Εισαγωγικό
\\	Backslash
\r	Carriage Return
\f	Form Feed
\b	Backspace

Παράδειγμα 10: escape_characters

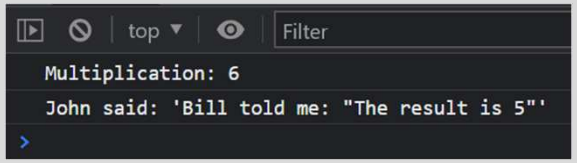
```
console.log("quotes: \", \'")
console.log("slashes: / \\")
console.log("tabs:\n\t|\t|a\t|aa\t|aaa\t|")
console.log("change line: \n newline \r carriage return")
console.log("Backspace: bb\baa\bc\b\baa\b")
```

Template Strings:

- Εξαιρετικά χρήσιμα είναι και τα template strings (ES6)
  - (Θα τα δούμε και πιο αναλυτικά σε επόμενο μάθημα)
  - Η συμβολοσειρά εντίθεται σε **backticks** (` => κουμπί στο πληκτρολογίο αριστερά από το 1)
  - Μέσα στη συμβολοσειρά, μπορούμε να ενσωματώσουμε παραστάσεις **${…}** όπου η παράσταση εντός των αγκίστρων αντικαθίσταται από το αποτέλεσμα του υπολογισμού της.

Παράδειγμα 11: template_strings

```
let x = 2;
let y = 3;
let str = `Multiplication: ${x*y}`;
console.log(str);
console.log(`John said: 'Bill told me: "The result is ${x+y}"'`);
```



1.3. null και undefined

null:

- Είναι μία δεσμευμένη λέξη-κλειδί (keyword) της JS
- **Συμβολίζει την ανυπαρξία τιμής**
- Εσωτερικά αναπαρίσταται με ένα αντικείμενο που έχει μία μοναδική τιμή (την null).

Παράδειγμα 12: null.html

```
let firstName = "George";
let middleName = "W";
let lastName = "Bush";
console.log(firstName + " " + middleName + " " + lastName);

firstName = "Joe";
middleName = null;
lastName = "Biden";
console.log(firstName + " " + lastName);
console.log(middleName);
```

Παρατήρηση (αν και πολλοί διαφωνούν, βλ. βίντεο)

- Είναι προγραμματιστικά συνηθισμένο να χρησιμοποιούμε το null για να συμβολίσουμε την ανυπαρξία τιμής, όπως στο παραπάνω παράδειγμα: Κάποιοι έχουν μεσαίο όνομα και κάποιοι δεν έχουν. Συμβολίζουμε με null την απουσία μεσαίου ονόματος.

undefined:

- Είναι μία καθολική σταθερά
- **Επίσης συμβολίζει την ανυπαρξία τιμής.**
- Εσωτερικά αναπαρίσταται με έναν ειδικό τύπο, τον undefined.

Παρατήρηση:

- Το undefined χρησιμοποιείται από την ίδια τη γλώσσα για να συμβολίσει την ανυπαρξία τιμής, π.χ.:
  - όταν δηλώνουμε μία μεταβλητή και δεν την αρχικοποιούμε με τιμή
  - Όταν αποθηκεύουμε την επιστροφή κλήσης συνάρτησης που δεν επιστρέφει τιμή, κ.α.

Παράδειγμα 13: undefined.html

```
let x;
console.log(x);
console.log(typeof x);
```

Παρατήρηση:

- Το **Symbol** έχει μόνο εξειδικευμένες χρήσεις και θα το μελετήσουμε σε ένα από τα τελευταία μαθήματα της σειράς.

JS beef: (beef: Σημείο έντονης διαφωνίας - βλ. βίντεο)

- Η ύπαρξη και των δύο είναι σχεδιαστικό λάθος της JS.
- Το null το χρησιμοποιούμε σε δικές μας μεταβλητές για την ανυπαρξία τιμής, ενώ το undefined το αφήνουμε στη γλώσσα για τις ειδικές περιπτώσεις χρήσης (βλ. παραπάνω)

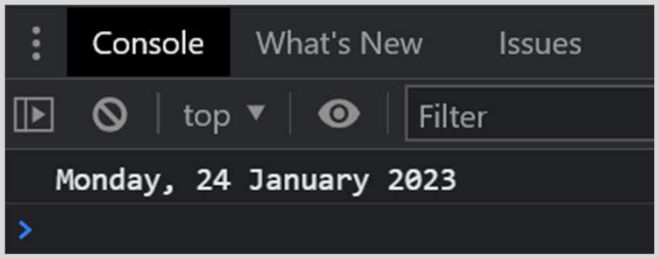
2. Ασκήσεις

Άσκηση 2: Εκτύπωση ημερομηνίας

Ορίστε τις μεταβλητές:

- weekDay: Να απεικονίζει την μέρα της εβδομάδας (π.χ. Monday, Tuesday, …)
- day: Να απεικονίζει την ημερολογιακή μέρα του μήνα (π.χ. 17, 24, …)
- month: Να απεικονίζει το πλήρες όνομα του μήνα (π.χ. February, March,…)
- year: Να απεικονίζει το έτος (π.χ. 1999, 2009, 2022, …)

Έπειτα να τυπώνει μορφοποιημένα την ημερομηνία που απεικονίζεται στις 4 μεταβλητές ως εξής:



(Χρησιμοποιήστε ένα template string για την τελική εκτύπωση)

Άσκηση 3: Μήνυμα καλωσορίσματος

- Κατασκευάστε μια μεταβλητή με όνομα firstName
- Αρχικοποιήστε την με το όνομα σας
- Κατασκευάστε μια μεταβλητή με όνομα surname
- Αρχικοποιήστε την με το επώνυμό σας

Το πρόγραμμα να τυπώνει Hello, έπειτα το ονοματεπώνυμό σας και τελικά ένα θαυμαστικό, π.χ.: