Εργαστήριο #12

Από τα προηγούμενα εργαστήρια:

Το εργαστήριο αυτό είναι ανεξάρτητο από τα προηγούμενα· επειδή όμως ασχολείται με τη γλώσσα JavaScript, βεβαιωθείτε ότι έχετε διαβάσει το εισαγωγικό **Παράρτημα Γ** του 11ου εργαστηρίου.

Οδηγίες

Στο σημερινό εργαστήριο θα εξασκηθείτε στη χρήση της Javascript για τον προγραμματισμό στην πλευρά του browser του νέου στοιχείου HTML που ονομάζεται **canvas**. Το στοιχείο αυτό μας επιτρέπει να "ζωγραφίσουμε" σχήματα και άλλα στοιχεία δύο διαστάσεων (2D) μέσα στην ιστοσελίδα – η προσθήκη σχημάτων γίνεται αποκλειστικά μέσω JavaScript.

Ακολουθήστε τα παρακάτω βήματα:

⇒ *Βήμα 1ο*.

Στο σημερινό εργαστήριο θα χρησιμοποιήσετε δύο αρχεία:

- ένα αρχείο html (**κατάληξη .html)**
- ένα αρχείο Javascript (**με κατάληξη .js**)

Ξεκινήστε από ένα "φρέσκο" αρχείο HTML· μπορείτε να χρησιμοποιήσετε το υπόδειγμα από το site του εργαστηρίου. Δημιουργήστε ένα νέο (κενό προς το παρόν) αρχείο κειμένου για τον κώδικα JavaScript.

Διαβάστε στο **Παράρτημα Α.1** πώς θα συνδέσετε τα δύο αρχεία, χρησιμοποιώντας το στοιχείο **<script>** της HTML.

⇒ *Βήμα 2ο*.

Διαβάστε το **Παράρτημα Β.1** και εισάγετε ένα στοιχείο **<canvas>** με διαστάσεις **600x400** pixels στο <body> της ιστοσελίδας. Μην παραξενευτείτε που δεν βλέπετε τίποτα στον browser! Το στοιχείο <canvas> είναι αρχικά κενό και διαφανές!

⇒ *Βήμα 3ο*.

Πριν αρχίσετε να αλληλεπιδράτε μέσω της JavaScript με τα στοιχεία HTML της ιστοσελίδας, θα **πρέπει η ιστοσελίδα να έχει φορτωθεί πλήρως και να έχουν σχηματιστεί όλα τα στοιχεία της**. Διαβάστε το **Παράρτημα Α.2** και προσθέστε μια συνάρτηση για τον χειρισμό του γεγονότος **onload** της ιστοσελίδας.

⇒ *Βήμα 4ο*.

Προσθέστε μια φόρμα για να μπορεί ο χρήστης να αλληλεπιδρά με την εφαρμογή σας. Το ζητούμενο είναι:

1. Να μπορεί ο χρήστης να επιλέξει από μια drop-down λίστα μεταξύ 3 σχημάτων (square, circle, text).
2. Να υπάρχει επίσης drop-down λίστα για επιλογή 3 χρωμάτων (red, green, blue).
3. Να υπάρχει ένα πλαίσιο για την εισαγωγή κειμένου, όταν ο χρήστης επιλέξει στη drop-down λίστα το "text".
4. Τέλος, να υπάρχει ένα κουμπί για την εκτέλεση της επιλογής του χρήστη.

Η φόρμα αυτή δεν θα στέλνει τίποτα στον web server (προορίζεται μόνο για αλληλεπίδραση με τον χρήστη) και πρέπει να τη χειριστείτε μέσω JavaScript. Διαβάστε το **Παράρτημα Γ.1** για να δείτε πώς θα το επιτύχετε.

⇒ *Βήμα 5ο*.

Προσθέστε στο αρχείο JavaScript μια συνάρτηση χειρισμού για το γεγονός **onclick** του κουμπιού της φόρμας. Προς το παρόν η συνάρτηση αυτή θα είναι κενή. Δείτε πώς θα το κάνετε στο **Παράρτημα Γ.2**.

⇒ *Βήμα 6ο*.

Διαβάστε το **Παράρτημα Β.2** για να μάθετε πώς να σχεδιάζετε πάνω στην επιφάνεια του στοιχείου <canvas>.

⇒ *Βήμα 7ο*.

Διαβάστε το **Παράρτημα Γ.3** για να μάθετε πώς μπορείτε μέσω JavaScript να ανακτήσετε τις τιμές των στοιχείων εισόδου της φόρμας. Στη συνέχεια, προσθέστε κώδικα στη συνάρτηση χειρισμού **onclick** του κουμπιού, έτσι ώστε να σχεδιάζετε σε κάθε πάτημα ανάλογα με την επιλογή του χρήστη.

Το μέγεθος και η θέση κάθε σχήματος θα υπολογίζονται δυναμικά με τη βοήθεια των εξής συναρτήσεων της JavaScript:

Math.random()	επιστρέφει έναν float ψευδοτυχαίο αριθμό r, $0 \le r < 1$
Math.floor(f)	επιστρέφει τον μέγιστο ακέραιο αριθμό μικρότερο ή ίσο με τον float f

Για να βρείτε έναν ψευδοτυχαίο ακέραιο αριθμό μεταξύ mink και maxk, γράφετε:

```
Math.floor(Math.random()*(maxk-mink+1))+min
```

Επίσης: για να βρείτε τα μέγιστα όρια, μπορείτε να χρησιμοποιήσετε τις ιδιότητες `canvas.width` και `canvas.height` του αντικειμένου Javascript που αντιπροσωπεύει το στοιχείο <canvas>.

⇒ *Βήμα 8ο (προαιρετικό)*.

Διαβάστε στο **Παράρτημα Β.3** πώς θα αποθηκεύσετε το περιεχόμενο του <canvas> σε ένα αρχείο. Προσθέστε ένα δεύτερο κουμπί στη φόρμα, καθώς και τον κατάλληλο κώδικα για να μπορεί ο χρήστης να αποθηκεύει το αποτέλεσμα της σχεδίασης.

Παράρτημα Α: JavaScript στον browser

Στη συνέχεια δίνονται ορισμένα στοιχεία για τη σχέση της JavaScript με τον προγραμματισμό του browser.

Το παρόν **δεν είναι μια αναλυτική περιγραφή της Javascript**· για την ίδια τη γλώσσα θα δοθούν κατά περίπτωση ορισμένα στοιχεία, όπου αυτό είναι αναγκαίο. Αρκεί να γνωρίζετε ότι η σύνταξη είναι όμοια με της C (στις εντολές διακλάδωσης και επανάληψης, τα block εντολών, τα σχόλια, τον τερματισμό κάθε εντολής με το ;).

Η σύνταξη στη χρήση των μεταβλητών είναι επίσης παρόμοια με της C. Υπάρχουν όμως **σημαντικές διαφορές στον τύπο και τη δήλωση των μεταβλητών**:

- Οι μεταβλητές μπορούν να αλλάζουν δυναμικά τύπο κατά τη διάρκεια της εκτέλεσης. Π.χ. ενώ μια μεταβλητή έχει περιεχόμενο integer, μπορεί αργότερα να αντιπροσωπεύει ένα string.
- Οι μεταβλητές δηλώνονται, χωρίς όμως προσδιορισμό τύπου, μέσω της λέξης-κλειδιού **var**. Μπορείτε να έχετε δήλωση και αρχικοποίηση μαζί:

```
// δήλωση μεταβλητής – η τιμή της είναι "undefined"
var i;


// δήλωση και αρχικοποίηση μεταβλητής
var str = 'some text';  // μπορείτε να χρησιμοποιήσετε και "
```

- Η εμβέλεια των μεταβλητών είναι είτε σφαιρική (global), είτε τοπική σε συναρτήσεις (local). **Δεν υπάρχει εμβέλεια block όπως στη C**.

Α.1. Πού θα προσθέσετε τον κώδικα JavaScript.

Ο κώδικας Javascript μπορεί να προστεθεί:

- είτε μέσα στη δήλωση ενός στοιχείου HTML, ως κώδικας χειρισμού γεγονότος (βλ. 11ο εργαστήριο)
- είτε μέσα στην ιστοσελίδα, μέσα σε ένα στοιχείο <script>
- είτε τέλος σε ξεχωριστό αρχείο, όπως και οι οδηγίες CSS, πάλι με τη χρήση του στοιχείου <script>.

Ο τελευταίος τρόπος είναι και ο πλέον πρότυπος και θα τον ακολουθήσετε στο εργαστήριο. Μέσα στο στοιχείο <head> εισάγετε το εξής:

```
<script src="code.js"></script>
```

υποθέτοντας φυσικά ότι το αρχείο JavaScript ονομάζεται `code.js` και βρίσκεται στον ίδιο φάκελο με την ιστοσελίδα. Παρατηρήστε ότι υπάρχει και η ετικέτα τέλους </script>, παρόλο που δεν υπάρχει περιεχόμενο.

Α.2. Το αντικείμενο window μιας ιστοσελίδας.

Από το 11ο εργαστήριο γνωρίζετε ήδη το αντικείμενο **document**. Το αντικείμενο αυτό μας επιτρέπει την πρόσβαση στα στοιχεία HTML της ιστοσελίδας.

Υπάρχει όμως ένα **κεντρικό αντικείμενο**, πατέρας όλων των αντικειμένων της ιστοσελίδας – το **περιβάλλον εκτέλεσης** του κώδικα Javascript της ιστοσελίδας – το οποίο ονομάζεται **window** (η ονομασία είναι κληρονομιά μιας εποχής όπου ένας browser είχε μόνο ένα παράθυρο και απεικόνιζε μία και μοναδική ιστοσελίδα).

Το αντικείμενο **window** παρέχει το χώρο όπου ορίζονται όλα τα άλλα αντικείμενα της JavaScript για τη συγκεκριμένη ιστοσελίδα. Ένα από αυτά το γνωρίζετε ήδη:

```
το document είναι "παιδί" του αντικειμένου window:
window.document


επειδή το window είναι το πρωταρχικό αντικείμενο, μπορούμε να
περιγράψουμε τα "παιδιά" του χωρίς τη λέξη window. Έτσι λέμε
απλά:
document
```

Ιδιότητες του αντικειμένου window που θα χρησιμοποιήσετε.

window.onload

Εάν οριστεί, παρέχει τη συνάρτηση που θα εκτελεστεί όταν η ιστοσελίδα και όλα τα αντικείμενα που περιέχει φορτωθούν πλήρως. Μόνο τότε έχει σχηματιστεί το δέντρο DOM των στοιχείων της ιστοσελίδας!

Η συνάρτηση που θα δώσετε κάνει συνήθως τις αρχικοποιήσεις και συνδέει τα στοιχεία HTML με τον κώδικα χειρισμού των γεγονότων τους. Εισάγετε τη συνάρτηση αυτή στο αρχείο JavaScript ως εξής:

```
window.onload = function() {
  // ...ο κώδικας της συνάρτησης εδώ...
};
```

Σας παραξενεύει ο παραπάνω κώδικας; Στην JavaScript οι συναρτήσεις είναι αντικείμενα, όπως οτιδήποτε άλλο. Έτσι, μπορείτε να **δημιουργήσετε** μια (ανώνυμη) συνάρτηση όπως στο παράδειγμα και να την **αναθέσετε** σε μια άλλη μεταβλητή.

Φυσικά μπορείτε να ορίσετε μια συνάρτηση κατά τον παραδοσιακό τρόπο, όπως θα δείτε αργότερα.

window.location

Περιέχει πληροφορίες διεύθυνσης (URL) της ιστοσελίδας που είναι φορτωμένη στον browser:

```
// διαβάζοντας το αντικείμενο location παίρνετε το URL
// της τρέχουσας ιστοσελίδας
var currentURL = window.location;

// γράφοντας στο location οδηγείτε τον browser σε νέα
// ιστοσελίδα
window.location = "http://di.ionio.gr ";
```

Παράρτημα Β: Το στοιχείο <canvas> της HTML5

Το νέο στοιχείο **<canvas>** επιτρέπει τη σχεδίαση γραφικών σε μια ιστοσελίδα. Η σχεδίαση αυτή γίνεται μέσω JavaScript, με τη χρήση μεθόδων που παρέχει το προγραμματιστικό αντικείμενο του στοιχείου **<canvas>**.

Όταν εισάγετε γραφικά σε ένα στοιχείο <canvas>, τα πάντα μετατρέπονται σε pixels. Αυτό σημαίνει ότι δ**εν μπορείτε** αργότερα να πείτε π.χ. "εκείνη η γραμμή (που πρόσθεσα προηγουμένως) να αλλάξει χρώμα ή θέση". Απλούστατα, "εκείνη η γραμμή" δεν υπάρχει ως ξεχωριστό αντικείμενο!

Αν θέλετε παρόμοια λειτουργικότητα, υπάρχει η αντίστοιχη τεχνολογία: ονομάζεται SVG αλλά δεν θα καλυφθεί στο εργαστήριο.

Β.1. Πώς εισάγεται ένα στοιχείο <canvas> στην ιστοσελίδα.

Ακολουθήστε το παράδειγμα, το οποίο εισάγει ένα στοιχείο **<canvas>** με διαστάσεις 320x200 "pixels ζωγραφικής":

```
<canvas id="testcanvas" width="320" height="200">
    Canvas not supported...
</canvas>
```

Η ιδιότητα **id** δίνει ένα αναγνωριστικό όνομα στο στοιχείο <canvas>. Το όνομα αυτό θα χρησιμοποιηθεί αργότερα από την JavaScript.

Το περιεχόμενο του στοιχείου θα φανεί μόνο στην περίπτωση που ο browser δεν υποστηρίζει το στοιχείο <canvas>!

Β.2. Σχεδιάζοντας πάνω σε ένα στοιχείο <canvas>.

Πριν κάνετε οτιδήποτε άλλο, πρέπει να προσπελάσετε το **context** σχεδίασης (δύο διαστάσεων – 2D) του στοιχείου <canvas>:

```
// επιλέγουμε το αντικείμενο του στοιχείου canvas
var canvas = document.getElementById("testcanvas");
// ανάκτηση του context για σχεδίαση
var context = canvas.getContext("2d");
```

Στη συνέχεια, μπορείτε να σχεδιάσετε ένα γεμάτο (filled) ορθογώνιο παραλληλόγραμμο ως εξής:

```
// χρώμα γεμίσματος για τις επόμενες ενέργειες,
// έστω ότι η μεταβλητή drawcolor περιέχει ένα string
// με την περιγραφή του χρώματος, π.χ. "red"
context.fillStyle = drawcolor;
// δημιουργία παραλληλογράμμου, έστω ότι οι μεταβλητές
// startx, starty, width, height περιέχουν ακέραιες τιμές
context.fillRect(startx,starty,width,height);
```

Η σχεδίαση ενός κύκλου είναι λίγο πιο περίπλοκη: πρέπει πρώτα να δημιουργήσετε ένα καμπύλο μονοπάτι (**path**) και στη συνέχεια να ζητήσετε την εμφάνισή του με γέμισμα:

```
//  χρώμα γεμίσματος για τις επόμενες ενέργειες
context.fillStyle = drawcolor;
// έναρξη path
context.beginPath();
// προσθήκη στο path καμπύλης με κέντρο τα tx,ty (ακέραιοι),
// ακτίνα radius (ακέραια τιμή), αρχική γωνία από τον άξονα x,
// τελική γωνία (σε ακτίνια, στο παράδειγμα 2π).
// Η τελευταία παράμετρος δείχνει αν η καμπύλη θα διαγραφεί
// στη φορά των δεικτών του ρολογιού (false) ή
// αντίστροφα (true).
context.arc(tx,ty,radius,0,2*Math.PI,true);
// τέλος path
context.closePath();
// εμφάνιση και γέμισμα path
context.fill();
```

Τέλος, για να σχεδιάσετε κείμενο:

```
//  χρώμα γεμίσματος για τις επόμενες ενέργειες
context.fillStyle = drawcolor;
// font για σχεδίαση
context.font = "bold 5em sans-serif";
// εμφάνιση κειμένου με γέμισμα, text είναι ένα string
context.fillText(text,tx,ty);
```

Β.3. Αποθήκευση περιεχομένου στοιχείου <canvas>.

Μπορείτε να μετατρέψετε σε εικόνα, να δείτε στον browser και στη συνέχεια να αποθηκεύσετε το περιεχόμενο του στοιχείου <canvas> ως εξής:

```
// ανάκτηση canvas
var canvas = document.getElementById("testcanvas");
// μετατροπή σε εικόνα (data URL)
var imgdata = canvas.toDataURL("image/png");
// μετάβαση στην εικόνα
window.location = imgdata;
```

Παράρτημα Γ: Φόρμες HTML και JavaScript

Στο σημερινό εργαστήριο θα δουλέψετε με μια φόρμα HTML μέσω της JavaScript και μόνο.

Γ.1. Φόρμα HTML χωρίς αποστολή δεδομένων στον web-server.

Εισάγετε τη φόρμα (form) και τα στοιχεία της (select, input κλπ) όπως γνωρίζετε από τα προηγούμενα εργαστήρια. Οι διαφορές είναι οι εξής:

- Δεν θα βάλετε τις ιδιότητες **method** και **action** στην ετικέτα του form (εφόσον δεν στέλνετε δεδομένα).
- Φροντίστε κάθε στοιχείο εισόδου (select, input κλπ) να έχει ένα ξεχωριστό όνομα, μέσω τις ιδιότητας **id**. Το id αυτό θα χρησιμοποιηθεί στην JavaScript για να επιλέξετε το συγκεκριμένο στοιχείο.
- Για να υλοποιήσετε το κουμπί της φόρμας, χρησιμοποιήστε το στοιχείο button, όπως στο εξής παράδειγμα:

```
<input id="drawbutton" type="button" value="Draw!">
```

Δηλώνοντας `type="button"` παίρνετε ένα κουμπί που δεν κάνει τίποτα (εκτός κι αν φροντίσετε διαφορετικά μέσω JavaScript!).

Γ.2. Συναρτήσεις χειρισμού γεγονότων (event handlers).

Κάθε στοιχείο εισόδου της φόρμας δημιουργεί διάφορα γεγονότα, ανάλογα με το αν ο χρήστης το επιλέγει, αλλάζει το περιεχόμενό του ή πιέζει το ποντίκι πάνω του.

Στο σημερινό εργαστήριο θα χρειαστείτε το εξής:

στοιχείο εισόδου	γεγονός (event)
button	**onclick** – όταν ο χρήστης πιέζει το κουμπί

Θα πρέπει να εισάγετε μια συνάρτηση χειρισμού γεγονότος, π.χ.:

```
function drawButtonHandler() {
  // εδώ γράφετε τις λειτουργίες που θέλετε να γίνουν
  // μόλις συμβεί το γεγονός
}
```

και στη συνέχεια, μέσα στην αρχικοποίηση του **window.onload** να συνδέσετε το στοιχείο εισόδου που θέλετε με τη συνάρτηση αυτή, π.χ.:

```
window.onload = function() {
  // επιλέγουμε πρώτα το στοιχείο εισόδου, μέσω του id του
  var drawbutton = document.getElementById("drawbutton");
  // στη συνέχεια συνδέουμε τη συνάρτηση χειρισμού στο γεγονός
  // onclick
  drawbutton.onclick = drawButtonHandler;
};
```

Στο προηγούμενο παράδειγμα, παρατηρήστε την πολύ σημαντική μέθοδο **getElementById()** του αντικειμένου **document**. Η μέθοδος αυτή δέχεται ως όρισμα το όνομα (id) ενός στοιχείου HTML και επιστρέφει το αντίστοιχο αντικείμενο της JavaScript που το αντιπροσωπεύει.

Γ.3. Ανάκτηση τιμών από στοιχεία εισόδου φόρμας HTML.

Ανάκτηση τιμής από στοιχείο select

Παράδειγμα:

```
// ανάκτηση αντικειμένου Javascript του στοιχείου
var drawcolorselect = document.getElementById("drawcolor");
// ανάκτηση του αριθμού (index) της τρέχουσας επιλογής
var sel = drawcolorselect.selectedIndex
// ανάκτηση της τιμής της τρέχουσας επιλογής
var drawcolor = drawcolorselect[sel].value;
```

Ανάκτηση τιμής από πλαίσιο κειμένου

Παράδειγμα:

```
// ανάκτηση αντικειμένου Javascript του στοιχείου
var drawtextbox = document.getElementById("drawtext");
// ανάκτηση του τρέχοντος κειμένου
var text = drawtextbox.value;
```