



# **Responsive, Fluid, Mobile First Design and Media Queries**

**Αθανάσιος Ανδρούτσος**



# Responsive Design (1)

Προγραμματισμός στο Web

- **Responsive design** ονομάζονται ο σχεδιασμός της σελίδας μας με τρόπο ώστε να εμφανίζεται ορθά σε όλες τις συσκευές είτε με μεγάλο μέγεθος οθόνης όπως **laptop, desktop** (πάνω από 992px) ή σε μικρές οθόνες όπως **κινητά τηλέφωνα, tablets** (κάτω από 992px)



# Responsive Design (2)

Προγραμματισμός στο Web

- Πρόκειται για μία προσέγγιση στο Web Design και development που εξαλείφει τη διάκριση μεταξύ mobile-friendly εκδόσεων των web σελίδων μας καθώς και εκδόσεων για laptop και desktop
- Γενικά εξαλείφεται η διάκριση μεταξύ διαφορετικών μεγεθών οθονών και οι σελίδες μας εμφανίζονται με ορθό τρόπο παντού



# Μέγεθος οθόνης

Προγραμματισμός στο Web

- Το μέγεθος της οθόνης έχει να κάνει με τις φυσικές της διαστάσεις, πλάτος και ύψος της οθόνης
- Οι μικρές οθόνες όπως σε κινητά τηλέφωνα και tablets δημιουργούν επιπλέον απαιτήσεις στον σχεδιασμό των σελίδων μας, ώστε να εμφανίζονται ορθά



# Hardware Pixel (1)

Προγραμματισμός στο Web

- Το Hardware Pixel είναι το πλάτος και ύψος ενός σημείου της οθόνης μιας συσκευής.
- **Ανάλυση οθόνης μιας συσκευής** σημαίνει το **πλήθος των hardware pixels οριζοντίως και καθέτως**, κάτι που έχει να κάνει με την πυκνότητα των pixels ανά ίντσα (pixels per inch – ppi)
- Μπορεί μικρές οθόνες κινητών τηλεφώνων να έχουν μεγαλύτερη ανάλυση και πιο μεγάλες οθόνες όπως tablet ή laptop να έχουν μικρότερη. Αυτό εξαρτάται όπως αναφέραμε από το ppi



# Hardware Pixel (2)

Προγραμματισμός στο Web

- Στα κινητά τηλέφωνα μπορεί δύο οθόνες με ίδιες φυσικές διαστάσεις να έχουν διαφορετική φυσική ανάλυση (πόσα hardware pixels δηλαδή περιέχει η οθόνη)
- Για παράδειγμα το Apple iPhone 14 έχει ανάλυση οθόνης 1179x2556 στα 460 ppi. Έχει πλάτος 2.81 inches ( $= 2.81 * 2.54\text{cm} = 7,15\text{cm}$ )
- Ενώ το Samsung Redmi Note 11 έχει ανάλυση 1080x2400 στα 410 ppi. Έχει πλάτος 2.91 inches ( $2.91 * 2.54\text{cm} = 7,4\text{cm}$ ).
- Βλέπουμε πως το Samsung έχει μεγαλύτερο φυσικό πλάτος αλλά μικρότερη ανάλυση. Ο λόγος είναι ότι το Apple έχει μεγαλύτερο ppi



# CSS pixel

- Εμείς ιδανικά θα θέλαμε να προγραμματίζουμε με βάση τις φυσικές διαστάσεις της συσκευής άρα με βάση μία σταθερή τιμή pixel (CSS pixel), που να βασίζεται στο πραγματικό φυσικό πλάτος της συσκευής και όχι στα hardware pixels που μπορεί να διαφέρουν σε κάθε συσκευή
- Επομένως, θα πρέπει να υπάρχει μία αναλογία μεταξύ hardware pixels και CSS pixels, που ονομάζεται device pixel ratio (dpi)



# Συνήθειες Αναλύσεις Οθονών

Προγραμματισμός στο Web

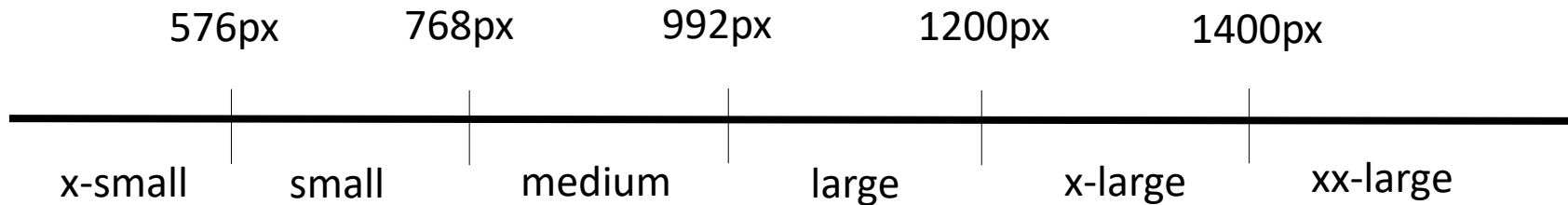
- **Breakpoints**

- (max-width: 576px) x-small mobile
- (min-width: 576px) mobile portrait– landscape
- (min-width: 768px) tablet
- (min-width: 992px) laptop/tablet/desktop
- (min-width: 1200px) large desktops
- (min-width: 1400px) extra large desktops





# Breakpoints

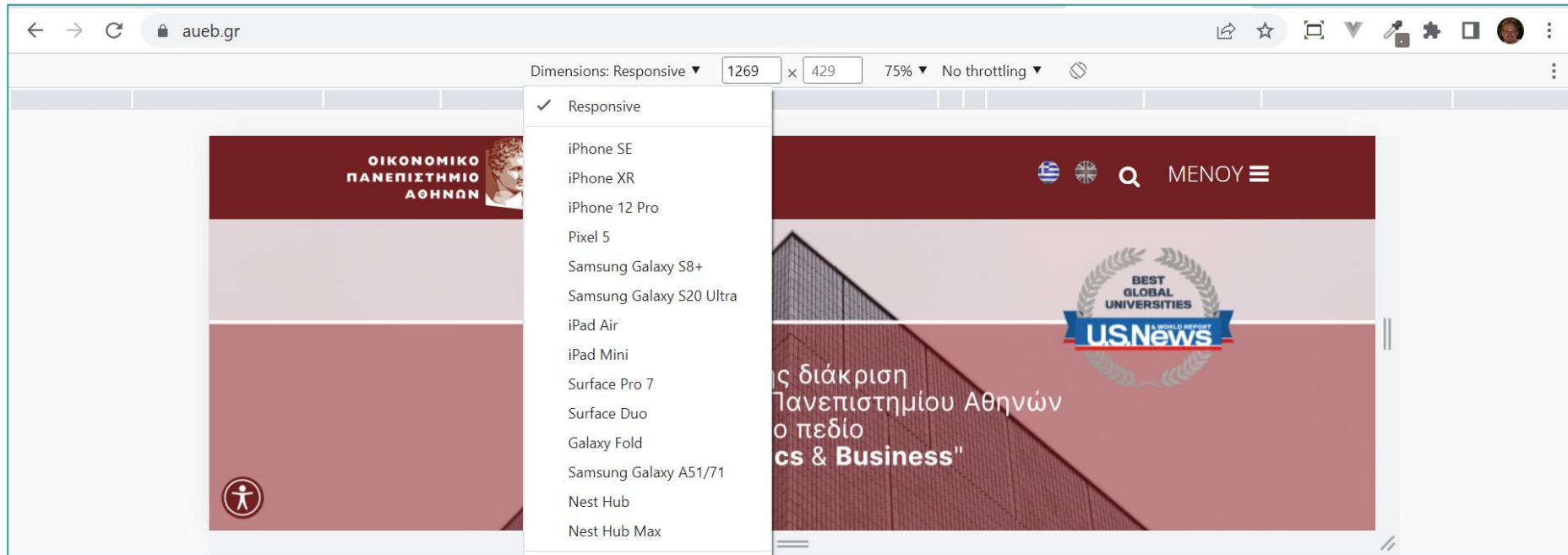


- Παρατηρούμε ότι υπάρχει ένας βασικός διαχωρισμός στο large, στα 992px.
- Και μετά πιο κάτω στα 768px στο small



# Έλεγχος Responsiveness (1)

Προγραμματισμός στο Web

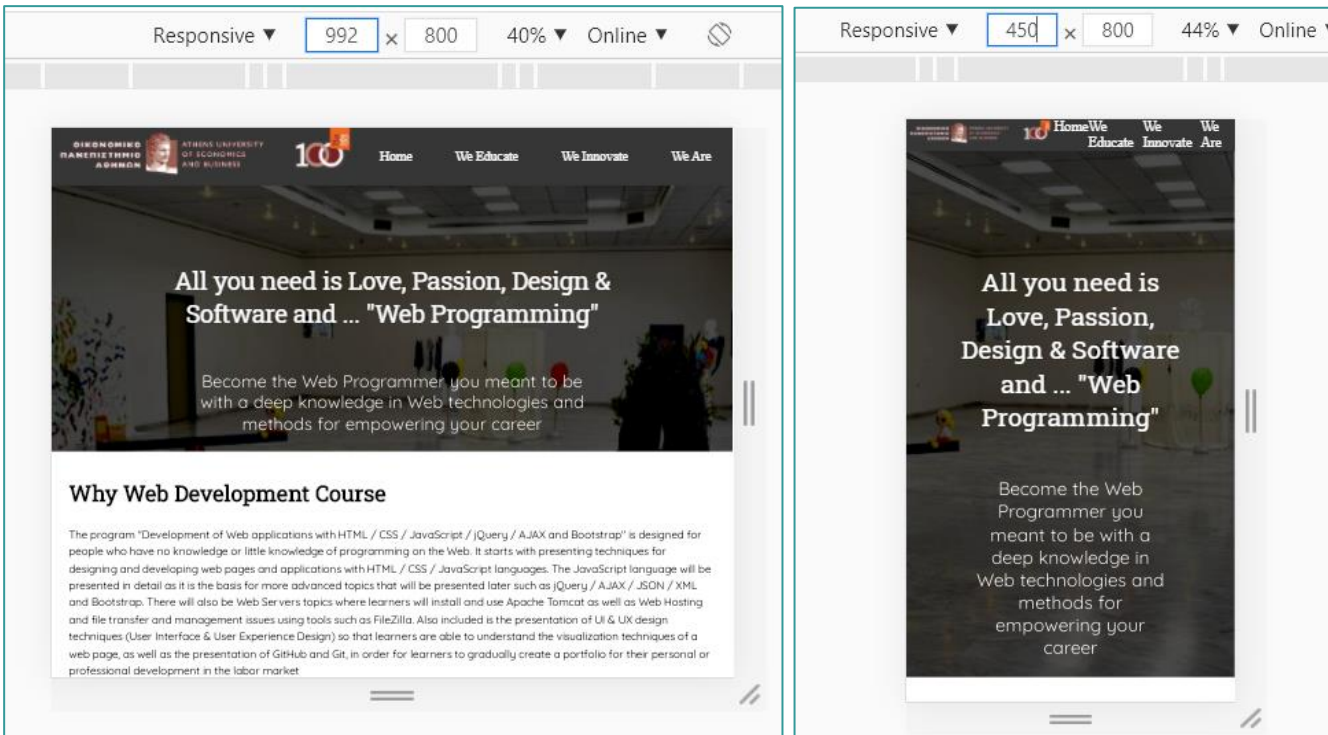


- Με δεξί κλικ και έλεγχος (inspect) στον Chrome μπορούμε να ελέγξουμε πως φαίνεται μία σελίδα σε διάφορα μεγέθη συσκευών



# Έλεγχος Responsiveness (2)

Προγραμματισμός στο Web

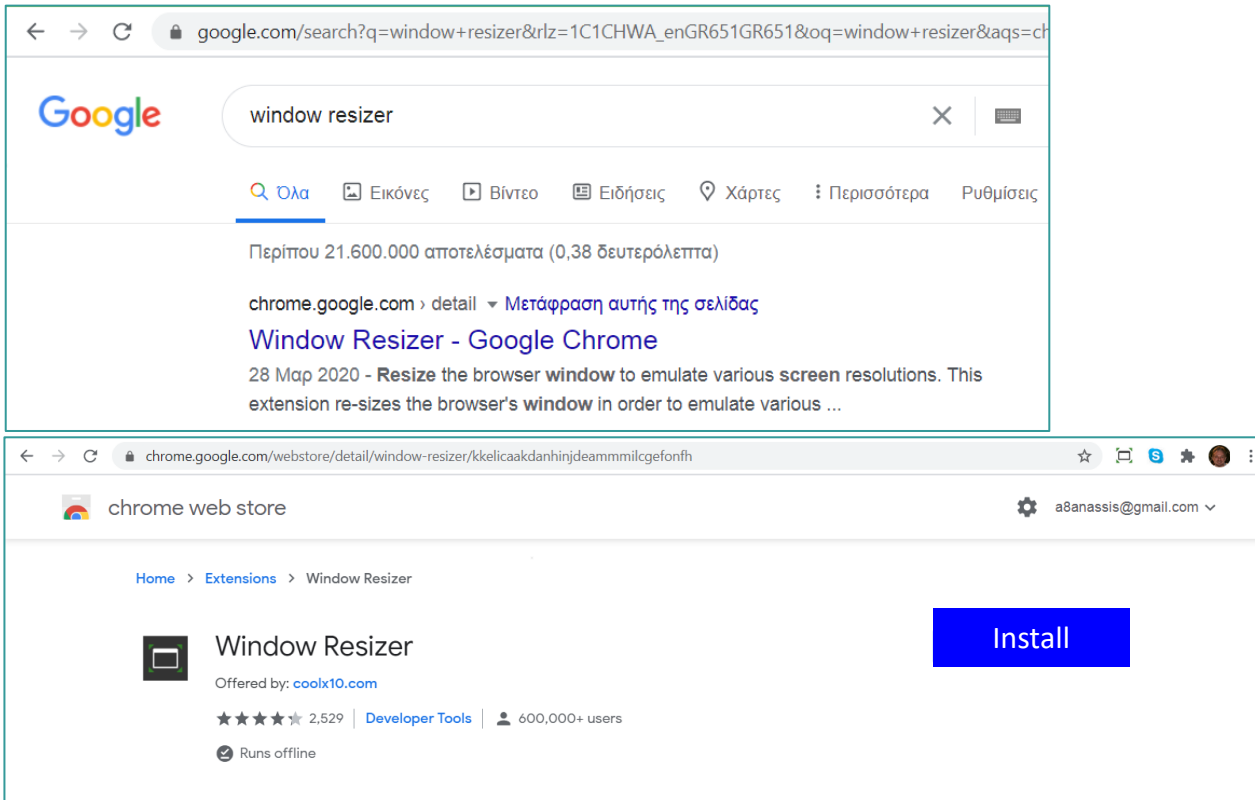


- Παρατηρούμε πως στη σελίδα με το μενού που είχαμε αναπτύξει, σε ανάλυση 450px το μενού δεν εμφανίζεται καλά



# Window Resizer

## Προγραμματισμός στο Web



- Αναζητούνε στο Google με Window Resizer και εγκαθιστούμε το Window Resizer plugin για chrome / Mozilla / MS Edge ή το Resize για Safari

- Πιο εύχρηστο εργαλείο ελέγχου, είναι ο *Window Resizer* για να μπορούμε εύκολα να βλέπουμε τις διαστάσεις του παραθύρου καθώς μικραίνουμε / μεγαλώνουμε το πλάτος του παραθύρου



# Toggle the resize tooltip

Προγραμματισμός στο Web

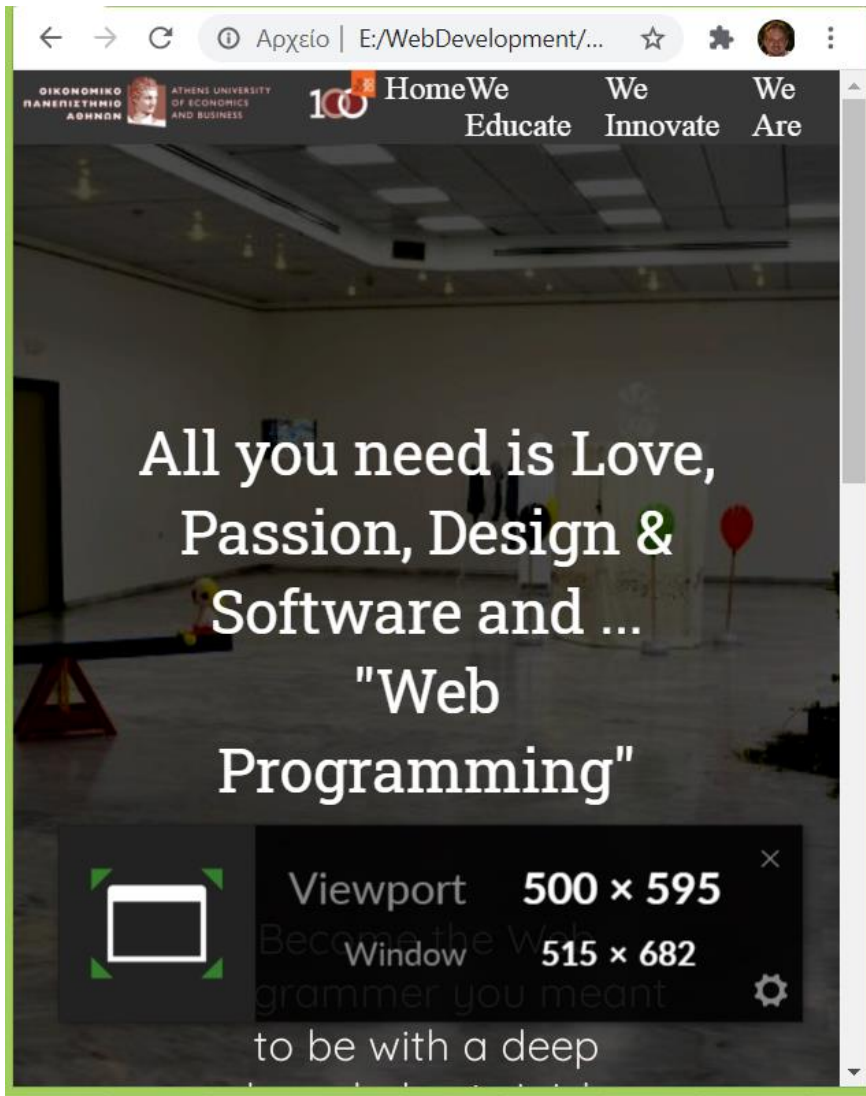


- Επιλέγουμε *Toggle the resize tooltip* για να εμφανίζονται οι διαστάσεις την ώρα που μετακινούμε με το ποντίκι τις διαστάσεις του παραθύρου



# Εμφάνιση διαστάσεων

Προγραμματισμός στο Web



- Παρατηρούμε πως σε πλάτος μικρότερο από 992px το Μενού αρχίζει και συρρικνώνεται ενώ κάτω από 550px χαλάει εντελώς
- Επίσης τα μηνύματα παραμένουν στο αρχικό μέγεθος που είναι μεγάλο για μικρές συσκευές





# Fixed-width Design (1)

- Το Responsive Design δεν είναι ιστορικά η 1<sup>η</sup> προσέγγιση στο σχεδιασμό σελίδων
- Στις αρχές του 1990-1995 οι περισσότερες οθόνες είχαν διαστάσεις 640x480 pixels, οπότε οι web σελίδες σχεδιάζονταν με πλάτος 640px
- Αντίστοιχα και λίγο αργότερα πριν το 2000, οι οθόνες είχαν διάσταση 800x600 pixels, οπότε ο σχεδιασμός των σελίδων βασίζονταν σε αυτές τις διαστάσεις
- Στη συνέχεια το μέγεθος των οθονών αυξήθηκε σε 1024x768 pixels
- Επομένως το fixed-width design λειτουργούσε ως sensible default



# Fixed-width Design (2)

- Στην περίπτωση του fixed-width design η σελίδα μας θα φαίνεται καλά μόνο σε συγκεκριμένες οθόνες με το αντίστοιχο πλάτος
- Αν μία οθόνη έχει μεγαλύτερο πλάτος θα υπάρχει κενός χώρος στο δεξί μέρος της οθόνης και ακόμα και αν στοιχίσουμε τη σελίδα μας στο κέντρο, θα υπάρχει κενός ανεκμετάλλευστος χώρος
- Αν μία οθόνη έχει μικρότερο πλάτος, τότε το περιεχόμενο της σελίδας μας δεν θα ταιριάζει και θα κόβεται. Οι browsers σε αυτές τις περιπτώσεις, δημιουργούν αυτόματα μία μπάρα οριζόντιας κύλισης για τη μετακίνηση της σελίδας δεξιά και αριστερά





# Fluid (Liquid) Design

Προγραμματισμός στο Web

- Για να αντιμετωπίσουμε το προηγούμενο πρόβλημα με τα fixed-width designs, θα μπορούσαμε να κάνουμε το layout μας flexible, χρησιμοποιώντας ποσοστά για το πλάτος των container (columns) της σελίδας μας
- Αυτός ο τρόπος σχεδιασμού ονομάζεται fluid ή liquid design. Και ενώ γενικά το fluid design δουλεύει καλά σε ένα μέσο όρο εύρους οθονών, σε αρκετά μεγάλες ή πολύ μεγάλες οθόνες το layout γίνεται stretched και επεκτείνεται σε όλο το πλάτος της οθόνης ενώ σε πολύ μικρές οθόνες, το layout συρρικνώνεται
- Σε κάθε περίπτωση, η εμφάνιση των περιεχομένων μας δεν είναι ιδανική



# Μικρές Οθόνες

Προγραμματισμός στο Web

- Με την έλευση των smartphones που περιείχαν web browsers, στον 21<sup>ο</sup> αιώνα, και ταυτόχρονα με νέες μεγάλες οθόνες, αλλά και με την ολοένα και αυξανόμενη χρήση του Web, οι σχεδιαστές σελίδων χρειάζονταν μία μεθοδολογία σχεδιασμού, ώστε οι σελίδες τους να εμφανίζονται το ίδιο καλά και σε πολύ μικρές οθόνες και σε πολύ μεγάλες οθόνες
- Επομένως το fluid design δεν ήταν επαρκές εργαλείο για κάτι τέτοιο μιας και δεν δούλευε καλά στα extremes



# Separate pages vs adaptive

- Μία λύση θα ήταν να κάνουμε sniffing το κινητό από όπου έρχεται το request και να έχουμε διαφορετικές εκδόσεις της σελίδας μας για διαφορετικά μεγέθη οθονών
- Κάτι τέτοιο όμως συνιστά παραβίαση της ασφάλειας του user-agent ενώ επίσης ακόμα κι αν κάτι τέτοιο ήταν δυνατό θα ήταν δύσκολο να συντηρούμε δύο ή περισσότερα διαφορετικά code bases και designs



# Adaptive Layout

- Μία λύση στο πρόβλημα αυτό προήλθε από την τεχνική των CSS media queries που επέτρεψαν να δημιουργούμε διάφορα layouts με fixed-width που το κάθε ένα να φαίνεται καλά σε ένα συγκριμένο πλάτος συσκευής
- Ωστόσο, δεδομένου του πλήθους των διαφορετικών συσκευών, σε διαστάσεις συσκευών ανάμεσα στα διάφορα fixed-width layouts, πάλι θα υπάρχουν τα προβλήματα που αναφέρθηκαν σχετικά με τα fixed-width layouts



# Responsive Design (1)

Προγραμματισμός στο Web

- Ενώ το Adaptive design είναι ένα σύνολο media queries και fixed-width layouts,
- Το Responsive Design είναι ένα σύνολο media queries και fluid layouts
- Ο Ethan Marcotte, το 2010, ο οποίος εισήγαγε τον όρο Responsive Design, όρισε τρία κριτήρια επίτευξής του
  - Fluid Layouts
  - Fluid Media (εικόνες)
  - Media Queries



# Responsive Design (2)

Προγραμματισμός στο Web

- Αν ένα website είναι responsive θα πρέπει το layout (columns/div) και οι εικόνες να εμφανίζονται ορθά σε όλες τις οθόνες
- Οι βασικοί τρόποι κατασκευής layouts είναι ιστορικά τα floats και σήμερα τα CSS Flexbox και Grid layouts



# Ορθή προβολή στοιχείων

Προγραμματισμός στο Web

- Οι browsers στα mobile phones για να επιλύσουν κάπως το πρόβλημα των fixed-width layouts θεώρησαν ένα min-width στα 980px και κάτω από αυτό το πλάτος κάνουν zoom-out την σελίδα, γεγονός που οδηγεί σε σμίκρυνση των γραμμάτων
- Στην HTML5 μπορούμε να επιλύσουμε το θέμα αυτό θέτοντας στο <head> την οδηγία

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

- Γιαυτό σε όλες τις σελίδες πρέπει να έχουμε αυτή την οδηγία ώστε: 1) να μην θεωρούνται τα 980 pixels ως min-width αλλά το ίδιο το πλάτος της σελίδας (width=device-width) και 2) να μην γίνεται zoom-out αλλά να διατηρείται το zoom στο 1 (initial-scale=1.0)



# Media Types (1)

- Μπορούμε να χρησιμοποιούμε το `@media` (at-rule) για να επιλέγουμε media στα οποία θα εφαρμόζουμε ένα CSS κανόνα
- Τα media μπορεί να είναι `screen` (που αναφέρεται σε οθόνες), `print` (σε εκτυπωτές), `speech` (screen readers)






# Media Types (2)

```
testbed > ch6-responsive > css > # media1.css > ...  
1  body {  
2      background-color: ☐ black;  
3  }  
4  
5  @media print {  
6      body {  
7          background-color: ☐ grey;  
8      }  
9  }
```

- Για παράδειγμα, στον παραπάνω κανόνα επιλέγουμε μόνο κατά την εκτύπωση, το background να είναι grey ώστε να μην χρειάζεται πολύ μαύρο μελάνι



# HTML Template

```
testbed > ch6-responsive > <> media1.html >  html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Media Queries</title>
8      <link rel="stylesheet" href="./css/media1.css">
9  </head>
10 <body>
11
12 </body>
13 </html>
```

- Παραπάνω είναι ένα HTML template για την εισαγωγή media types μέσω CSS, με <link>



- Αν δεν προσδιορίσουμε media type, αυτόματα θεωρείται ως media type το all. Οι δύο παρακάτω κανόνες είναι ισοδύναμοι

```
1  body {  
2      background-color: grey;  
3  }  
4  
5  @media all {  
6      body {  
7          background-color: grey;  
8      }  
9  }
```



# Media Queries

Προγραμματισμός στο Web

- Μπορούμε να προσθέσουμε συνθήκες (conditions) στα @media types. Αυτές οι συνθήκες ονομάζονται media features
- Τα media types μαζί με τα media features ονομάζονται media queries και έχουν τη μορφή
- **@media *type* and (*feature*)**



# Responsive Design (1)

Προγραμματισμός στο Web

- Στο responsive design ένα από τα πιο χρήσιμα media features είναι το width του browser viewport
- Για να εφαρμόσουμε styling σε οθόνες μεγαλύτερες από ένα συγκεκριμένο width, χρησιμοποιούμε το min-width, ενώ για styling σε οθόνες μικρότερες από ένα συγκεκριμένο width, χρησιμοποιούμε το max-width



# Responsive Design (2)

Προγραμματισμός στο Web

```
testbed > ch6-responsive > css > # media2.css > {} @m
1  @media (min-width: 992px) {
2      body {
3          background-color: ■ green;
4      }
5  }
6
7  @media (max-width: 992px) {
8      body {
9          background-color: ■ yellow;
10     }
11 }
```

- Ορίζουμε δύο media queries με media features τα min-width και max-width



# Responsive Design (3)

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Media Queries</title>
8      <link rel="stylesheet" href="./css/media2.css">
9  </head>
10 <body>
11
12 </body>
13 </html>
```

- Με `<link>` εισάγουμε το CSS στην HTML σελίδα μας



# Length Units

Προγραμματισμός στο Web

- Μπορούμε να χρησιμοποιούμε όλα τα γνωστά CSS length units
- Αν χρησιμοποιούμε images, θα ήταν προτιμότερο να χρησιμοποιούμε pixels
- Αν χρησιμοποιούμε κείμενο θα ήταν καλύτερο να χρησιμοποιούμε relative length units, όπως em, rem,





# Συνδυασμός media queries

Προγραμματισμός στο Web

```
1  /* Large */
2  @media (min-width: 992px) and (max-width: 1200px) {
3      body {
4          background-color: green;
5      }
6  }
7
8  /* Extra large */
9  @media (min-width: 1201) and (max-width: 1400px) {
10     body {
11         background-color: red;
12     }
13 }
14
15 /* xx-large */
16 @media (min-width: 1401) {
17     body {
18         background-color: blue;
19     }
20 }
21
22 /* x-small, small, medium */
23 @media (max-width: 992px) {
24     body {
25         background-color: yellow;
26     }
27 }
```

- Με τον τελεστή **and** μπορούμε και συνδυάζουμε media features σε ένα media query



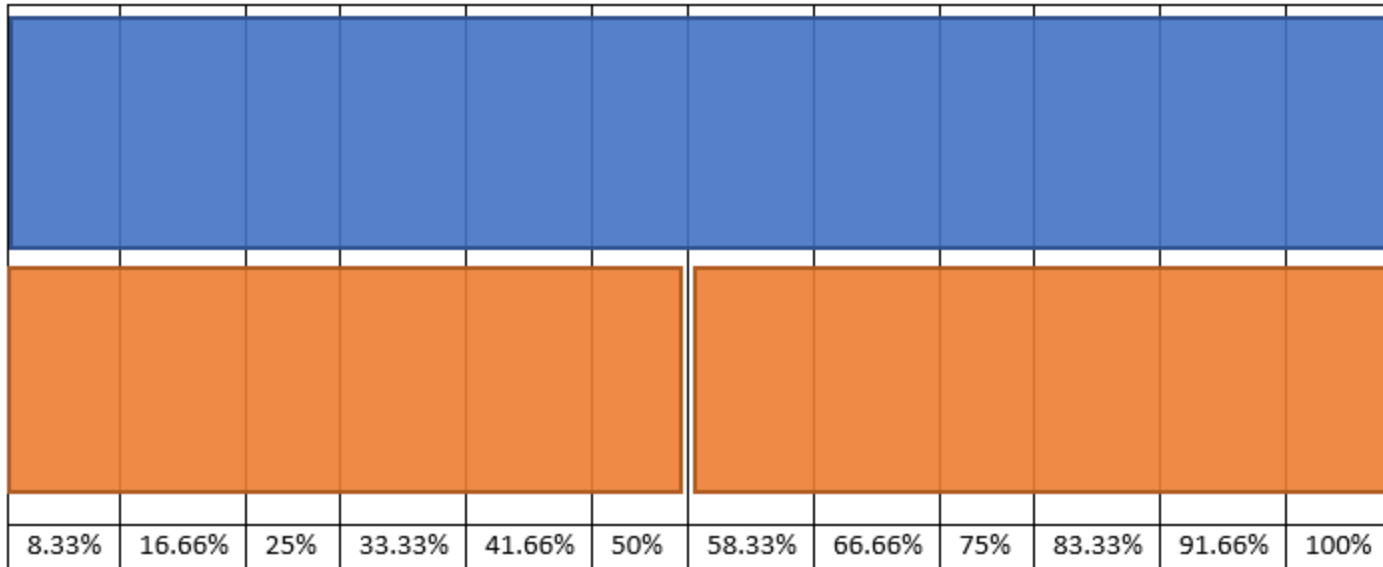
# Breakpoints

- Το σημείο στο οποίο ένα media feature γίνεται true (π.χ. 992px) ονομάζεται breakpoint
- Ιδανικά επιλέγουμε τα breakpoints των σελίδων μας με βάση το περιεχόμενό τους και όχι με βάση τα μεγέθη των συσκευών, τα οποία αλλάζουν
- Εκτός από το width, μπορούμε να χρησιμοποιήσουμε και άλλες CSS ιδιότητες ως conditions, όπως το height



# Responsive Grid System

Προγραμματισμός στο Web



- Το **responsive fluid grid system** που υποστηρίζεται τόσο από το W3.CSS όσο και από το Bootstrap αποτελείται από **12 στήλες**, όπου η κάθε στήλη είναι **8.33%** του viewport
- Έστω ότι θέλουμε να φτιάξουμε το παραπάνω layout με τρία <div>, το μπλε και τα δύο πορτοκαλί. Άρα έχουμε δύο γραμμές, όπου η 1<sup>η</sup> γραμμή περιέχει ένα <div> μπλε και η 2<sup>η</sup> γραμμή δύο <div> πορτοκαλί
- Το μπλε <div> πιάνει 12 στήλες και τα δύο πορτοκαλί από 6 στήλες



# Αρχείο HTML

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>12-column grid system</title>
8   <link rel="stylesheet" href="./css/grid-system.css">
9 </head>
10 <body>
11
12   <div class="container">
13
14     <div class="header box col-12 col-sm-12">
15       Lorem ipsum dolor sit amet consectetur adipisicing elit. Asperna
16       ipsa qui accusantium omnis vero, laboriosam culpa animi dolorum.
17     </div>
18     <div class="side box col-6 col-sm-12">
19       Lorem ipsum dolor sit amet consectetur adipisicing elit. Cumque
20       dolores dicta, ducimus exercitationem voluptatum laudantium modi
21     </div>
22     <div class="side box col-6 col-sm-12">
23       Lorem ipsum dolor sit amet consectetur, adipisicing elit.
24       Corrupti, quibusdam! Esse excepturi corporis, vitae soluta non
25     </div>
26   </div>
27 </body>
28 </html>
```

- Έχουμε τρία <div>
- Το 1ο <div> έχει class-name **header**, **col-12** και **col-sm-12**
- Τα επόμενα δύο <div> έχουν class-name **side**, **col-6** και **col-sm-12**
- Επίσης όλα έχουν class **box**



# Αρχείο CSS (1)

Προγραμματισμός στο Web

```
chapter6 > css > # grid-system.css > {} @media (min-width:
1  * {
2      box-sizing: border-box;
3      margin: 0;
4      padding: 0;
5  }
6
7  .container {
8      width: 100%;
9      padding: 0 15px;
10 }
11
12 .header {
13     background-color: cornflowerblue;
14 }
15
16 .side {
17     background-color: orange;
18 }
19
20 .box {
21     height: 150px;
22     border: 1px solid black;
23     float: left;
24 }
```

- Δίνουμε τις βασικές ρυθμίσεις καθώς και τις κλάσεις **container**, **header**, **side**, **box**
- Το **box** έχει `float: left` για οριζόντια αριστερή στοίχιση των **boxes**



# Αρχείο CSS (2)

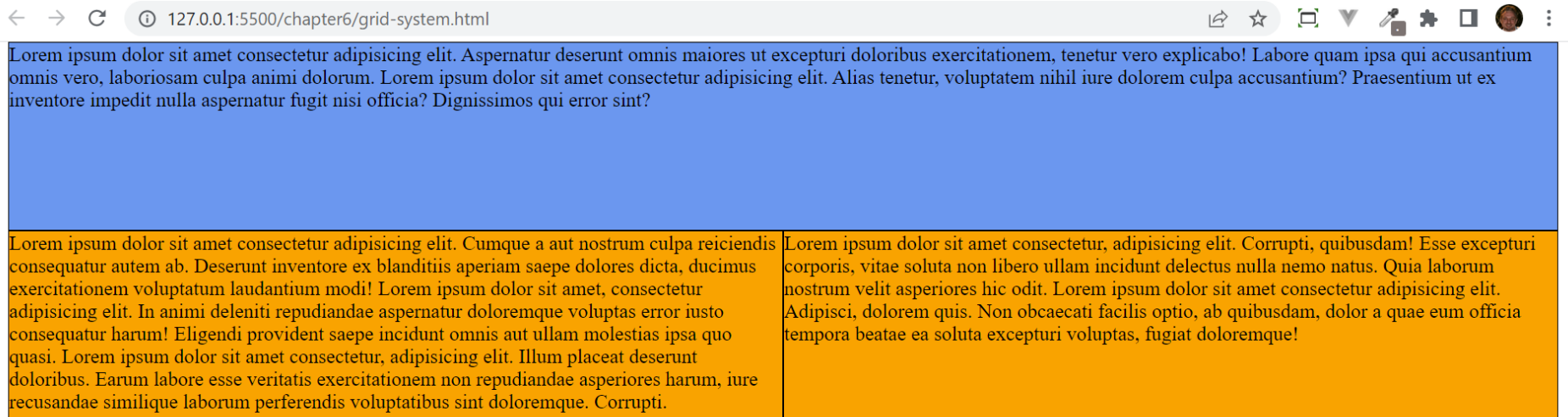
```
26 .col-1 { width: 8.33%; }
27 .col-2 { width: 16.66%; }
28 .col-3 { width: 25%; }
29 .col-4 { width: 33.33%; }
30 .col-5 { width: 41.66%; }
31 .col-6 { width: 50%; }
32 .col-7 { width: 58.33%; }
33 .col-8 { width: 66.66%; }
34 .col-9 { width: 75%; }
35 .col-10 { width: 83.33%; }
36 .col-11 { width: 91.66%; }
37 .col-12 { width: 100%; }
```

- Ορίζουμε **12**  
**κλάσεις** με  
κλιμακούμενο  
width ξεκινώντας  
από το 8.33%



# Οπτικό αποτέλεσμα

Προγραμματισμός στο Web



- Θα θέλαμε σε medium και μικρές οθόνες, κάτω από 992px τα δύο πορτοκαλί <div> να στοιχίζονται κάθετα, το ένα κάτω από το άλλο με 100% width το καθένα ώστε να φαίνονται πιο καλά
- Διαφορετικά θα φαίνονται μικρά σε μικρές οθόνες και τυχόν περιεχόμενό τους δεν θα είναι ευδιάκριτο



# Media Query

```
41 @media (min-width: 992px) {  
42     .col-1 { width: 8.33%; }  
43     .col-2 { width: 16.66%; }  
44     .col-3 { width: 25%; }  
45     .col-4 { width: 33.33%; }  
46     .col-5 { width: 41.66%; }  
47     .col-6 { width: 50%; }  
48     .col-7 { width: 58.33%; }  
49     .col-8 { width: 66.66%; }  
50     .col-9 { width: 75%; }  
51     .col-10 { width: 83.33%; }  
52     .col-11 { width: 91.66%; }  
53     .col-12 { width: 100%; }  
54 }
```

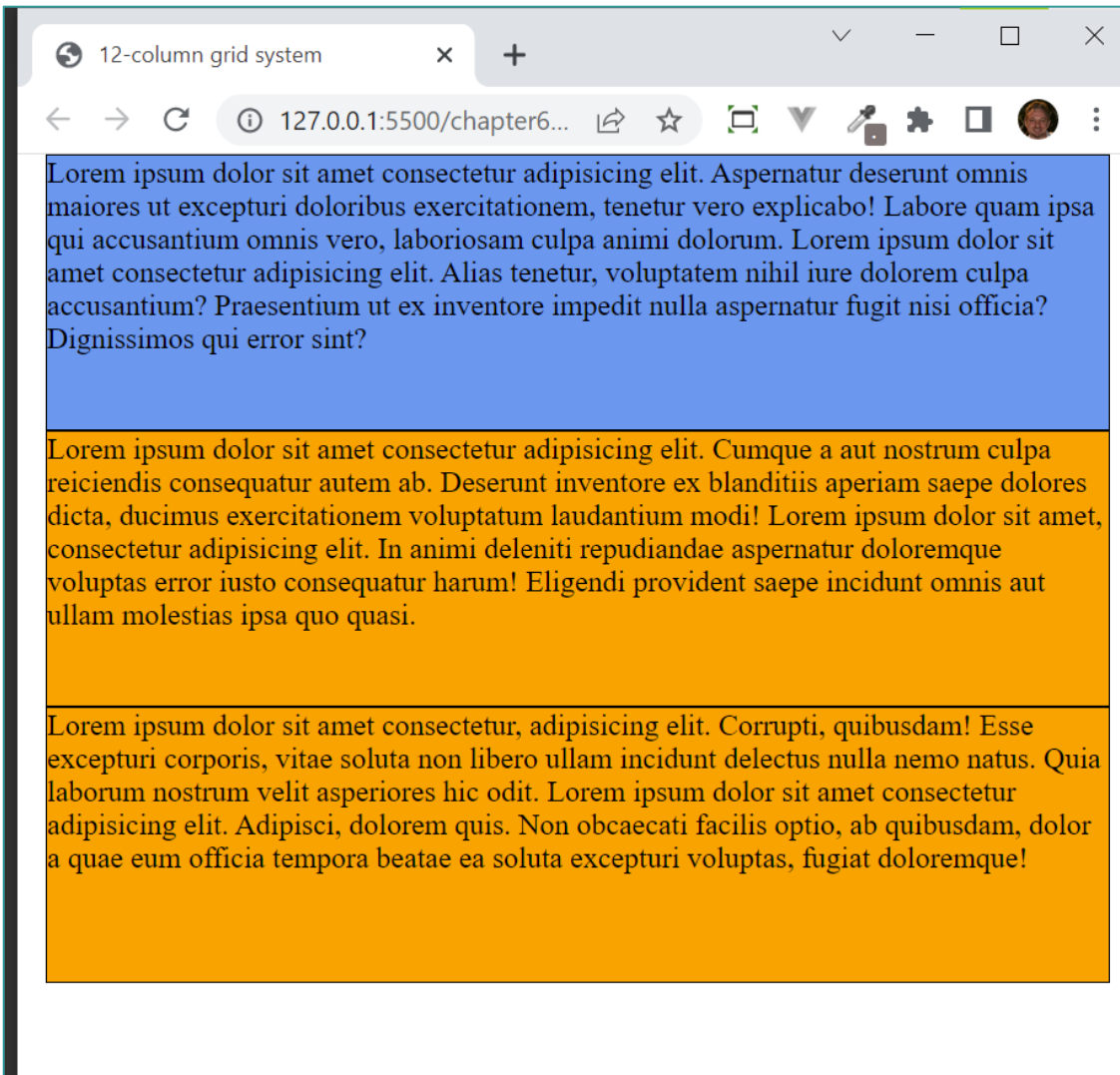
- Εισάγουμε στο CSS ένα media query για οθόνες μικρότερες από 992px
- Πάλι 12 κλάσεις με ονόματα που περιέχουν το `-sm` (small) και αντανakλούν τις μικρές οθόνες
- Στο HTML αρχείο έχουμε εισάγει την κλάση **col-sm-12** σε όλα τα `<div>` εφόσον θέλουμε όλα να έχουν πλάτος 100% σε μικρές και μεσαίες οθόνες





# Οθόνες κάτω από 992px

Προγραμματισμός στο Web



- Παρατηρούμε πως σε οθόνες μικρότερες από 992px τα boxes έχουν στοιχηθεί κάθετα
- Επίσης, τα περιεχόμενα των boxes εμφανίζονται ορθά



# Τελικό CSS

- Παρατηρούμε ότι κατ' αυτό τον τρόπο **πρώτα φορτώνονται οι κλάσεις για μεγάλες συσκευές** και μετά για mobile/tablet, οπότε η ταχύτητα εμφάνισης είναι καλύτερη σε μεγαλύτερες συσκευές από κινητά

```
28
29 .col-1 { width: 8.33%; }
30 .col-2 { width: 16.66%; }
31 .col-3 { width: 25%; }
32 .col-4 { width: 33.33%; }
33 .col-5 { width: 41.66%; }
34 .col-6 { width: 50%; }
35 .col-7 { width: 58.33%; }
36 .col-8 { width: 66.66%; }
37 .col-9 { width: 75%; }
38 .col-10 { width: 83.33%; }
39 .col-11 { width: 91.66%; }
40 .col-12 { width: 100%; }
41
42 @media (min-width: 992px) {
43 .col-sm-1 { width: 8.33%; }
44 .col-sm-2 { width: 16.66%; }
45 .col-sm-3 { width: 25%; }
46 .col-sm-4 { width: 33.33%; }
47 .col-sm-5 { width: 41.66%; }
48 .col-sm-6 { width: 50%; }
49 .col-sm-7 { width: 58.33%; }
50 .col-sm-8 { width: 66.66%; }
51 .col-sm-9 { width: 75%; }
52 .col-sm-10 { width: 83.33%; }
53 .col-sm-11 { width: 91.66%; }
54 .col-sm-12 { width: 100%; }
55 }
```



# Mobile First design (1)

Προγραμματισμός στο Web

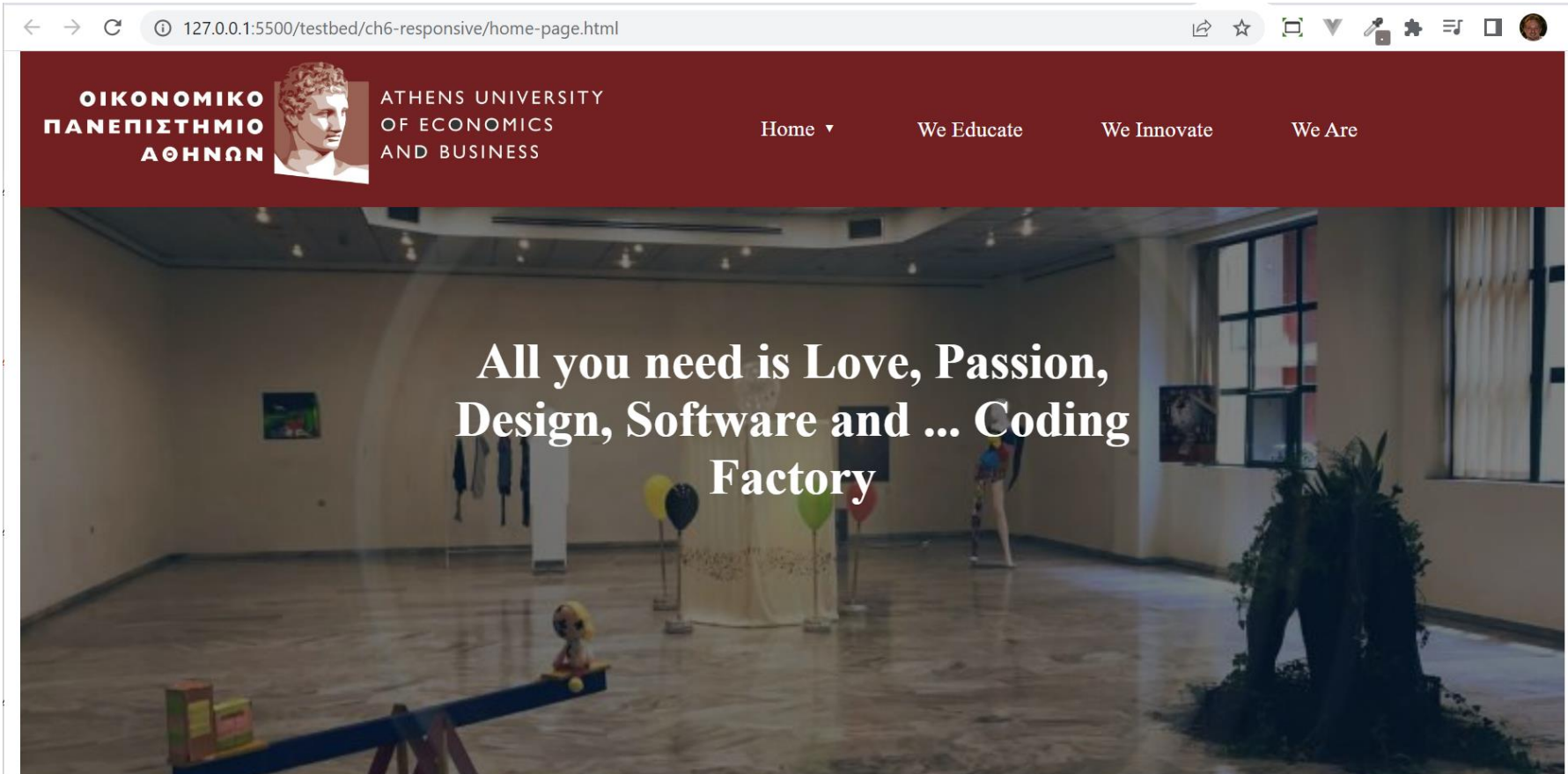
```
27 /* Mobile First Design */
28 .col-sm-1 { width: 8.33%; }
29 .col-sm-2 { width: 16.66%; }
30 .col-sm-3 { width: 25%; }
31 .col-sm-4 { width: 33.33%; }
32 .col-sm-5 { width: 41.66%; }
33 .col-sm-6 { width: 50%; }
34 .col-sm-7 { width: 58.33%; }
35 .col-sm-8 { width: 66.66%; }
36 .col-sm-9 { width: 75%; }
37 .col-sm-10 { width: 83.33%; }
38 .col-sm-11 { width: 91.66%; }
39 .col-sm-12 { width: 100%; }
40
41 @media (min-width: 992px) {
42     .col-1 { width: 8.33%; }
43     .col-2 { width: 16.66%; }
44     .col-3 { width: 25%; }
45     .col-4 { width: 33.33%; }
46     .col-5 { width: 41.66%; }
47     .col-6 { width: 50%; }
48     .col-7 { width: 58.33%; }
49     .col-8 { width: 66.66%; }
50     .col-9 { width: 75%; }
51     .col-10 { width: 83.33%; }
52     .col-11 { width: 91.66%; }
53     .col-12 { width: 100%; }
54 }
```

- Αλλάξαμε τη σειρά των κλάσεων καθώς και το max-width σε min-width μιας και το media query τώρα αναφέρεται στις μεγάλες συσκευές
- Το αποτέλεσμα δεν αλλάζει αλλά με αυτό τον τρόπο η ταχύτητα εμφάνισης είναι καλύτερη σε κινητά



# Βασική σελίδα

Προγραμματισμός στο Web



- Θα ξεκινήσουμε από τη βασική μας σελίδα και θα αναπτύξουμε ένα responsive menu



# Toggle menu - Responsive

Προγραμματισμός στο Web

- Μέσα στο header και μετά το .menu δίνουμε ένα νέο <div> με class name *nav-toggle* με ένα button (type button) που περιέχει το σύμβολο ☰;



```
11 <body>
12   <div class="container">
13
14     <!-- Header -->
15     <div class="header">
16       <div class="logo">
17         
18       </div>
19       <div class="navbar">
20         <ul class="nav">
21           <li class="dropdown"><a href="javascript:void(0)">Home ☰</a>
22             <ul class="dropdown-content">
23               <li><a href="#">Εκπαίδευση</a></li>
24               <li><a href="#">Έρευνα</a></li>
25               <li><a href="#">Διασυνδεση</a></li>
26               <li><a href="#">Υπηρεσίες</a></li>
27             </ul>
28           </li>
29           <li><a href="#">We Educate</a></li>
30           <li><a href="#">We Innovate</a></li>
31           <li><a href="#">We Are</a></li>
32         </ul>
33       </div>
34
35       <div class="nav-toggle">
36         <button type="button">☰</button>
37       </div>
38     </div>
```



# nav-toggle

```
182
183   .nav-toggle {
184       display: none;
185       margin-right: 30px;
186   }
187
188   .nav-toggle > button {
189       width: 50px;
190       height: 50px;
191   }
```

- Αρχικά δίνουμε **display: none** ώστε το *nav-toggle*, άρα και το *button* που περιέχει να μην φαίνεται
- Δίνουμε επίσης **width** και **height** στο toggle button



# Media Query

```
194 @media (max-width: 1200px) {  
195     .navbar {  
196         display: none;  
197     }  
198  
199     .nav-toggle {  
200         display: block;  
201     }  
202 }
```

- Θέτουμε ένα breakpoint στα 1200px και κάνουμε hidden με `display: none;` το `.navbar`, ενώ εμφανίζουμε με `display: block;` το `nav-toggle`





# Αποτέλεσμα Μενού

Προγραμματισμός στο Web

127.0.0.1:5500/testbed/ch6-responsive/home-page.html

ΟΙΚΟΝΟΜΙΚΟ  
ΠΑΝΕΠΙΣΤΗΜΙΟ  
ΑΘΗΝΩΝ



ATHENS UNIVERSITY  
OF ECONOMICS  
AND BUSINESS



All you need is Love, Passion,  
Design, Software and...  
Coding Factory

Viewport: 1151 x 593

Width: 1166 x 690





# Άλλες ρυθμίσεις (1)

```
204 @media (max-width: 1160px) {
205     .home-title span {
206         font-size: 2.1rem;
207     }
208 }
209
210 @media (max-width: 992px) {
211     .home-title span {
212         font-size: 1.8rem;
213     }
214 }
215
216 @media (max-width: 850px) {
217     .home-title span {
218         font-size: 1.5rem;
219     }
220
221     .logo > img {
222         width: auto;
223         height: 100px;
224     }
225 }
```

```
228 @media (max-width: 710px) {
229     .home-title {
230         line-height: 0.8;
231     }
232
233     .home-title span {
234         font-size: 1.3rem;
235     }
236 }
```

- Οι ρυθμίσεις αφορούν τα κείμενα καθώς και τις εικόνες
- Συνήθως στα κείμενα δίνουμε relative sizes (με rem) ενώ στις εικόνες δίνουμε px



# Άλλες ρυθμίσεις (2)

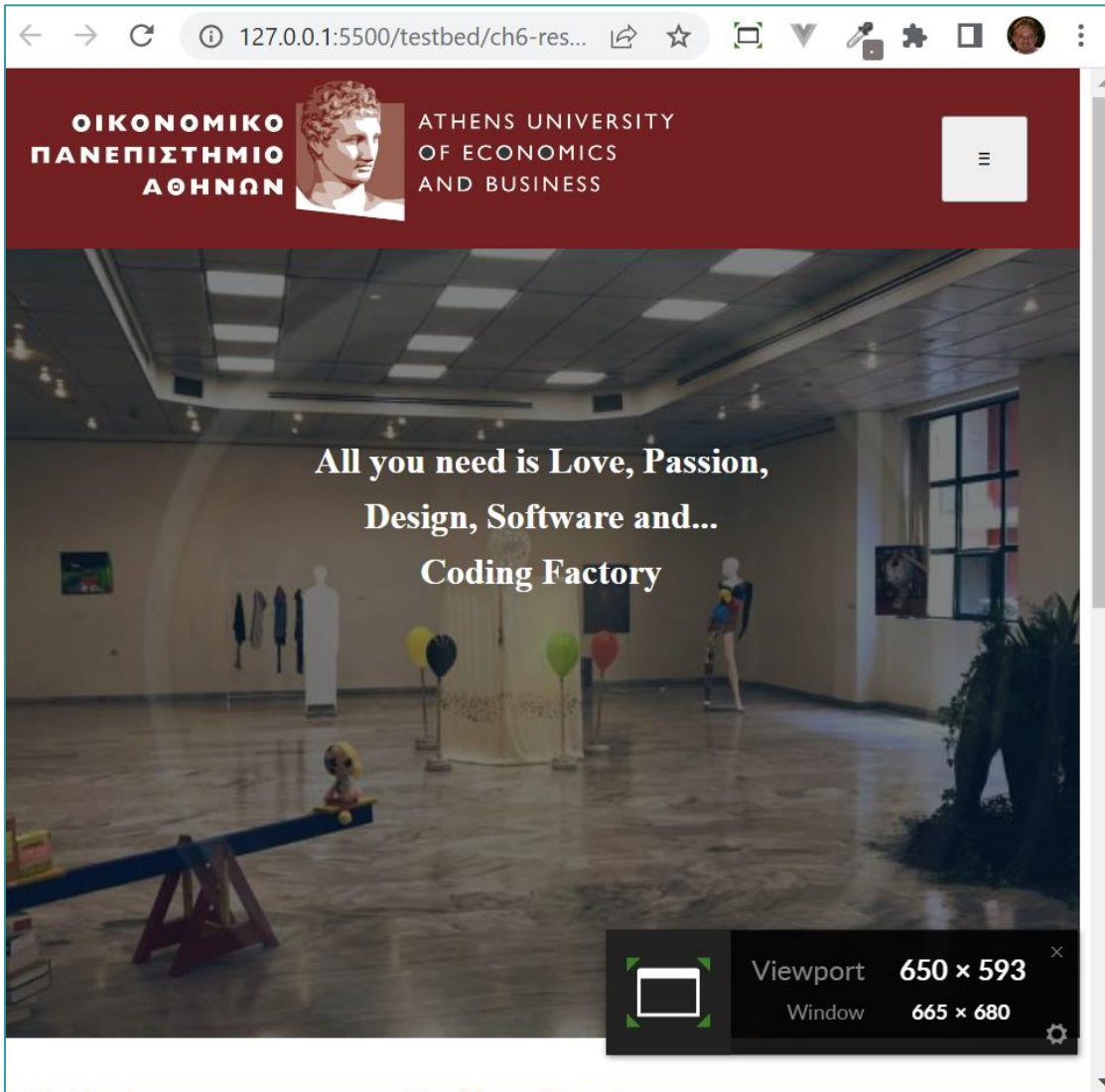
```
238 @media (max-width: 610px) {  
239     .home-title {  
240         line-height: 0.7;  
241     }  
242  
243     .home-title span {  
244         font-size: 1.1rem;  
245     }  
246  
247     .nav-toggle > button {  
248         width: 40px;  
249         height: 40px;  
250     }  
251  
252     .logo > img {  
253         width: auto;  
254         height: 80px;  
255     }  
256 }  
257
```

- Όπως πριν οι ρυθμίσεις αφορούν τα κείμενα καθώς και τις εικόνες
- Συνήθως στα κείμενα δίνουμε relative sizes (με rem) ενώ στις εικόνες δίνουμε px



# Breakpoint 710px

## Προγραμματισμός στο Web



- Το viewport είναι στα 650px. Το breakpoint είναι στα 710px και κάτω μέχρι 610px
- Έχουμε μειώσει το line-height και έχουμε αυξήσει το font-size

```
228 @media (max-width: 710px) {  
229     .home-title {  
230         line-height: 0.8;  
231     }  
232  
233     .home-title span {  
234         font-size: 1.3rem;  
235     }  
236 }
```



# Events

- Τα events είναι γεγονότα που δημιουργούνται είτε από το σύστημα ή από τον χρήστη, όπως το onclick event που δημιουργείται όταν ο χρήστης πατήσει κλικ πάνω σε ένα button
- Θα δούμε αναλυτικά τα events και τον event-driven programming με JavaScript στα επόμενα κεφάλαια αλλά ας δούμε τώρα ένα εισαγωγικό παράδειγμα



# Events – toggle button Μενού

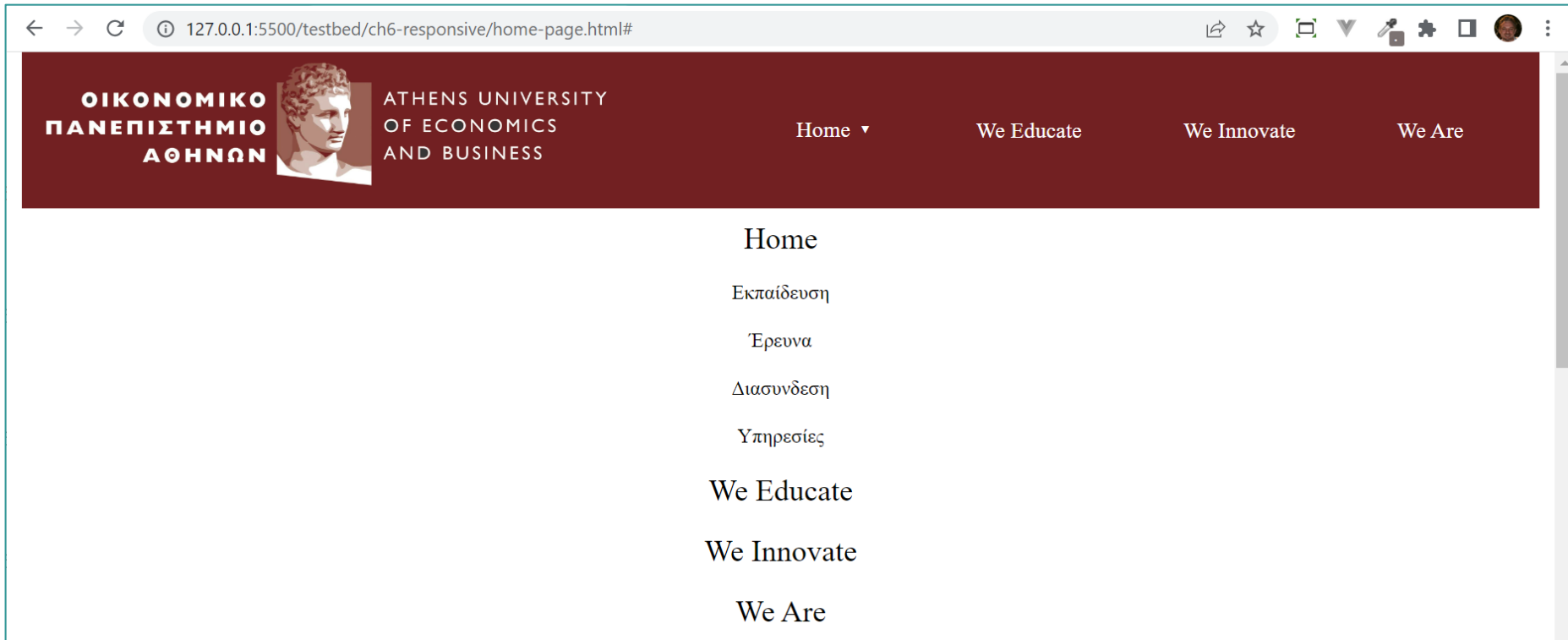
Προγραμματισμός στο Web

- Θα θέλαμε όταν ο χρήστης πατήσει **κλικ πάνω στο nav-toggle button** να ανοίξει ένα μενού παρόμοιο με το μενού που έχουμε σε πλήρη οθόνη
- Για να δίνει αυτό θα πρέπει: 1) να ορίσουμε στο HTML αρχείο το μενού, 2) να ορίσουμε στο CSS ότι αρχικά δεν θα φαίνεται καθώς και όταν εμφανιστεί πως θα φαίνεται, και 3) στο onclick event του nav-toggle button να εισάγουμε λίγο κώδικα JavaScript που να κάνει το μενού να εμφανίζεται και αν το ξαναπατήσουμε να ξανακλείνει



# Mock-up Toggle Menu

Προγραμματισμός στο Web



- Πρόκειται για ένα απλό μενού με μικρότερο font-size στα υπομενού. Τα υπομενού δεν είναι hidden για καλύτερη εμπειρία/ευχρηστία χρήστη



# HTML

```
40 <div class="navbar-toggle">
41   <ul class="nav">
42     <li class="dropdown"><a href="javascript:void(0)">Home</a>
43       <ul class="dropdown-content">
44         <li><a href="#">Εκπαίδευση</a></li>
45         <li><a href="#">Έρευνα</a></li>
46         <li><a href="#">Διασυνδεση</a></li>
47         <li><a href="#">Υπηρεσίες</a></li>
48       </ul>
49     </li>
50     <li><a href="#">We Educate</a></li>
51     <li><a href="#">We Innovate</a></li>
52     <li><a href="#">We Are</a></li>
53   </ul>
54 </div>
```

- Μετά το header και πριν το home-image κάνουμε copy/paste το Μενού μας και δίνουμε class-name navbar-toggle



# CSS – Αρχική απόκρυψη Toggle Μενού

Προγραμματισμός στο Web

```
258 .navbar-toggle {  
259     display: none;  
260 }
```

```
262 /* Vertical Layout*/  
263 .navbar-toggle .nav,  
264 .navbar-toggle .dropdown-content {  
265     display: flex;  
266     flex-direction: column;  
267     /* justify-content: center; */  
268     /* align-items: center; */  
269 }
```

- Θέτουμε το βασικό μενού `.navbar-toggle` σε `display: none`
- Τα `.nav` και `.dropdown-content` γίνονται flex με `flex-direction: column`
- Η στοίχιση είναι προαιρετική δεδομένου ότι θα στοιχίσουμε στο οριζόντιο κέντρο τα `<li>`





# CSS – Μορφοποίηση Toggle Μενού

Προγραμματισμός στο Web

```
279 .navbar-toggle li {
280     width: 100%;
281     text-align: center;
282     /* border-bottom: 1px solid black; */
283     /* background-color: aquamarine; */
284     list-style: none;
285 }
286
287 .navbar-toggle .dropdown-content li {
288     /* background-color: rgb(218, 209, 198); */
289 }
290
291
292 .navbar-toggle a {
293     display: block;
294     padding: 10px 0;
295     text-decoration: none;
296     color: black;
297     font-size: 1.5rem;
298 }
299
300 .navbar-toggle .dropdown-content a {
301     font-size: 1rem;
302 }
303
304 .hidden {
305     display: none;
306 }
```

- Μπορούμε να κρατήσουμε το toggle menu απλό
- Αν ωστόσο θέλουμε μπορούμε να δώσουμε border-bottom και hover
- Για hover πρέπει όπως παρακάτω να γίνει το υπομενού αρχικά display: none και μετά στο hover

```
/* .navbar-toggle .dropdown-content {
|     display: none;
| } */

/* .navbar-toggle .dropdown:hover .dropdown-content {
|     display: flex;
| } */
```



# JavaScript - View

```
111     <script>
112         const img = document.querySelector('.home-img')
113         const toggleBtn = document.querySelector('.nav-toggle')
114         const navbarToggle = document.querySelector('.navbar-toggle')
115         const main = document.querySelector('.main')
116         const footer = document.querySelector('.footer')
117
118         toggleBtn.addEventListener('click', function() {
119
120             img.classList.toggle('hidden')
121             navbarToggle.classList.toggle('show')
122             main.classList.toggle('hidden')
123             footer.classList.toggle('hidden')
124         })
125     </script>
126 </body>
127 </html>
```

- Πρώτα αντιστοιχούμε τα UI elements σε προγραμματιστικά αντικείμενα, για όλα τα αντικείμενα που θέλουμε να κάνουμε handle. Όλο το παράθυρο είναι το **View**, αυτό που βλέπει ο χρήστης



# JavaScript – Controller (1)

Προγραμματισμός στο Web

```
111 <script>
112     const img = document.querySelector('.home-img')
113     const toggleBtn = document.querySelector('.nav-toggle')
114     const navbarToggle = document.querySelector('.navbar-toggle')
115     const main = document.querySelector('.main')
116     const footer = document.querySelector('.footer')
117
118     toggleBtn.addEventListener('click', function() {
119
120         img.classList.toggle('hidden')
121         navbarToggle.classList.toggle('show')
122         main.classList.toggle('hidden')
123         footer.classList.toggle('hidden')
124     })
125 </script>
126 </body>
127 </html>
```

- Ο Controller είναι μία συνάρτηση που χειρίζεται events. Με `addEventListener()` στο button (toggle button) κάνουμε register στο button ένα Listener / Handler. Παίρνει δύο παραμέτρους, το event που εδώ είναι το click (στο button) και μία callback συνάρτηση που κάνει το handle (Handler ή controller).



# JavaScript – Controller (2)

Προγραμματισμός στο Web

## CSS

```
111 <script>
112   const img = document.querySelector('.home-img')
113   const toggleBtn = document.querySelector('.nav-toggle')
114   const navbarToggle = document.querySelector('.navbar-toggle')
115   const main = document.querySelector('.main')
116   const footer = document.querySelector('.footer')
117
118   toggleBtn.addEventListener('click', function() {
119
120     img.classList.toggle('hidden')
121     navbarToggle.classList.toggle('show')
122     main.classList.toggle('hidden')
123     footer.classList.toggle('hidden')
124   })
125 </script>
126 </body>
127 </html>
```

```
304 .hidden {
305   display: none;
306 }
307
308 .show {
309   display: block;
310 }
```

- Οι controllers κάνουν handle όλα τα events του View. Εδώ, η callback συνάρτηση καλεί την μέθοδο toggle που κάνει εναλλάξ add/remove την κλάση που παίρνει ως παράμετρο. Την κλάση hidden για να αποκρύψει και την κλάση show για να εμφανίσει



- Βελτιώστε τη σελίδα που έχετε ξεκινήσει να δημιουργείτε και κάντε τη Responsive (Αν δεν έχετε ξεκινήσει ακόμα, ξεκινήστε τη δημιουργία μίας αρχικής σελίδας της επιλογής σας)
- Προσθέστε ένα μενού αν δεν έχετε ήδη προσθέσει και κάντε το Responsive