



Flexbox Advanced Positioning

Αθανάσιος Ανδρούτσος

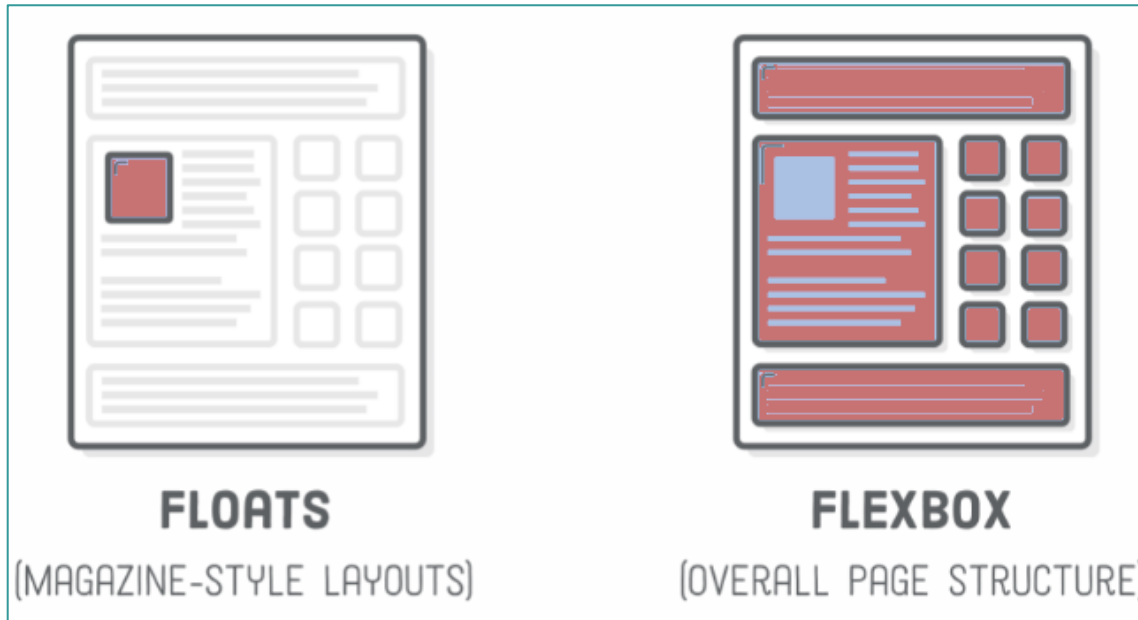


Flexbox (1)

- Το Flexible Box ή Flexbox Model είναι ένα καινούργιο (CSS3, 2009) μοντέλο για Layout management που έχει περισσότερα πλεονεκτήματα σε σχέση με το παλιότερο float (CSS1, 1996)
- Δίνοντας στο display property ενός container box την τιμή flex (**display: flex**) προστάζουμε τον browser ότι όλο το **περιεχόμενο του container box θα πρέπει να γίνει render με το Flexbox model**, αντί του default box model



Floats vs Flexbox



- Επομένως προτείνεται τα floats να χρησιμοποιούνται για magazine-style layout όπου κάνουμε wrap μία εικόνα γύρω από κείμενο ενώ το Flexbox για το γενικότερο Layout των σελίδων μας

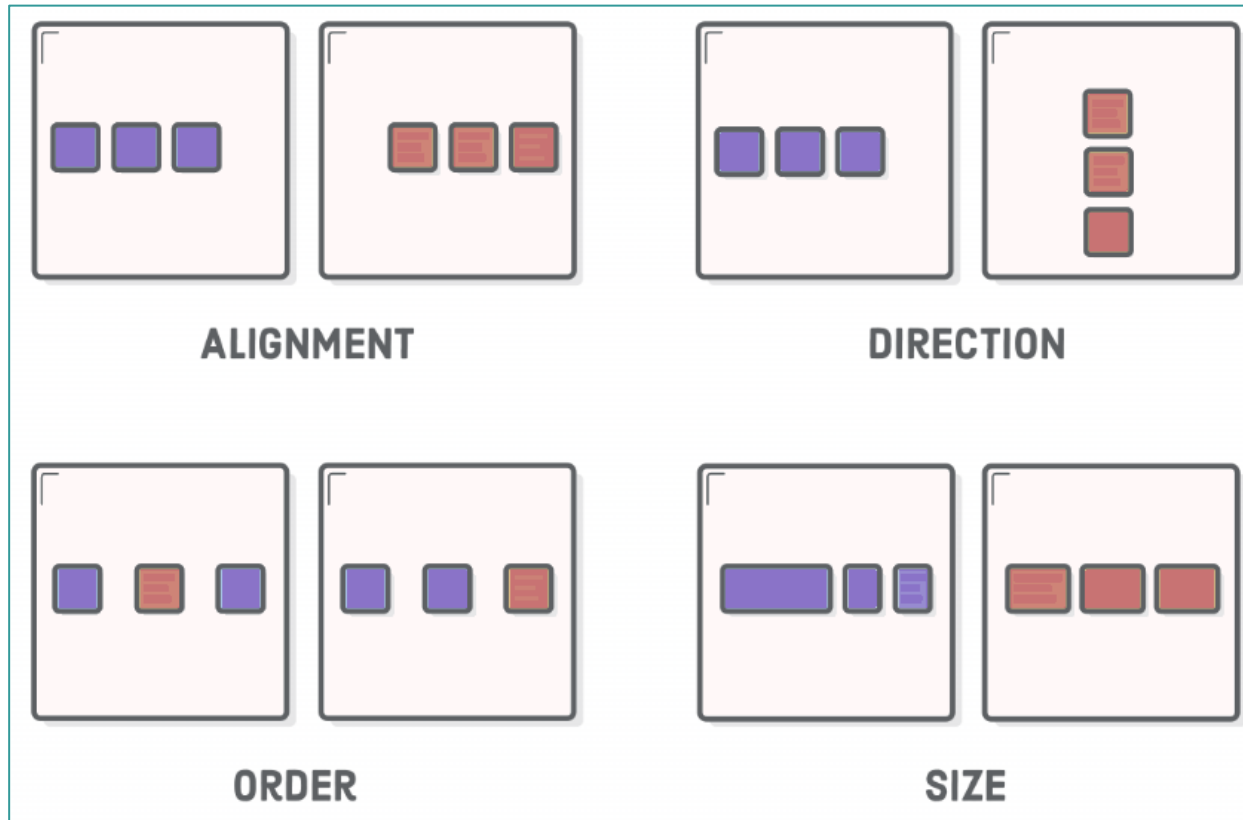


Flexbox Advs

- Ενώ τα floats επιτρέπουν μόνο την το οριζόντιο positioning των boxes, το flexbox παρέχει επίσης δυνατότητες διαχείρισης των:
 - Στοίχισης (**alignment** -- οριζόντια και κάθετα)
 - Κατεύθυνσης του flow (**position** -- οριζόντια, κάθετη)
 - **Sizing** (μέγεθος) των float inner αντικειμένων
 - **Ordering** (κατάταξης των flex elements του flex container)



Flex Items



Όπως έχουμε αναφέρει το Flexbox αφορά το layout των inner αντικειμένων (flex items)



Flexbox more advs

Προγραμματισμός στο Web

- Επομένως το positioning και το Layout management γίνεται πιο εύκολο με το Flexbox
- Επίσης, το Flexbox είναι **responsive** , **mobile friendly**, τα **margins του container δεν κάνουν collapse** με τα margins του περιεχομένου του και μπορούμε εύκολα να **αλλάξουμε το order** των child elements



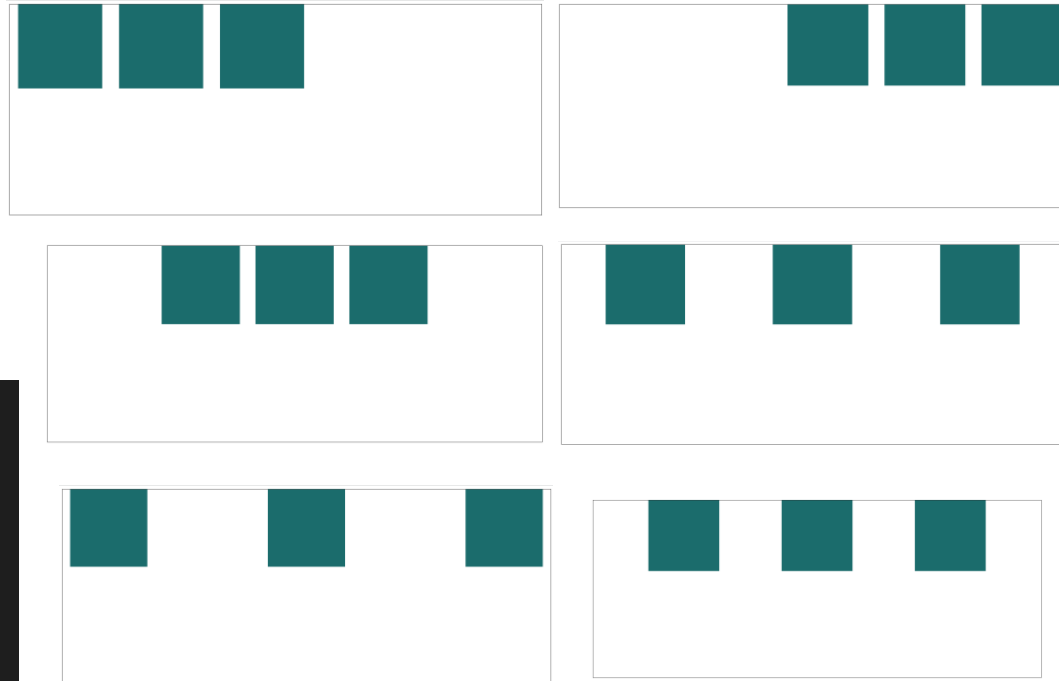
Aligning Flex Items

```
<body>
  <div class="container">
    <div class="box1"></div>
    <div class="box2"></div>
    <div class="box3"></div>
  </div>
</body>
```

```
.container {
  display: flex;
  justify-content: center;

  width: 100%;
  height: 500px;
  border: 1px solid black;
}

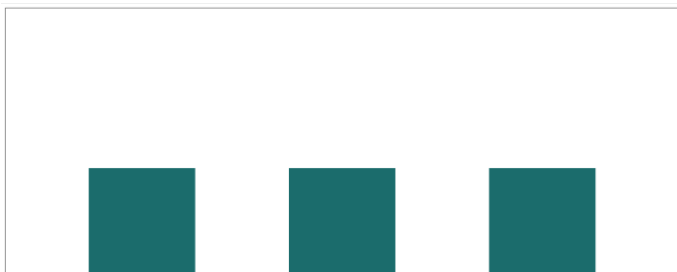
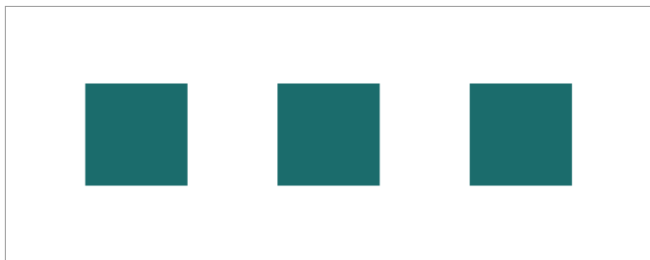
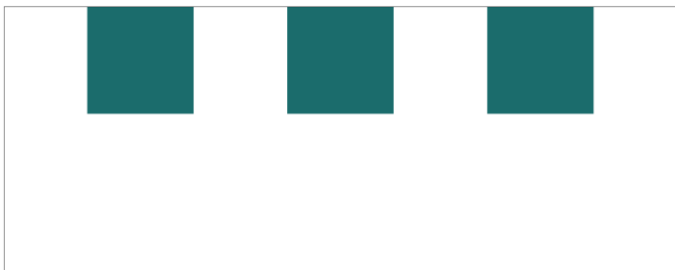
div[class^=box] {
  width: 200px;
  height: 200px;
  margin: 0 20px;
  background-color: #1f7874;
}
```



- Εμφανίζονται με τη σειρά από αριστερά προς τα δεξιά και από πάνω προς τα κάτω
flex-start | flex-end | center | space-around | space-between | space-evenly

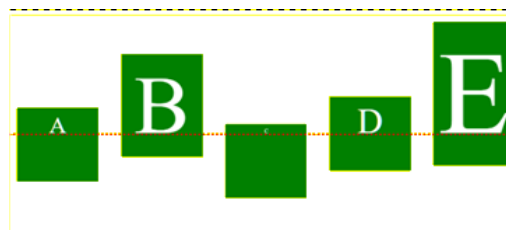


Κάθετη Στοίχιση



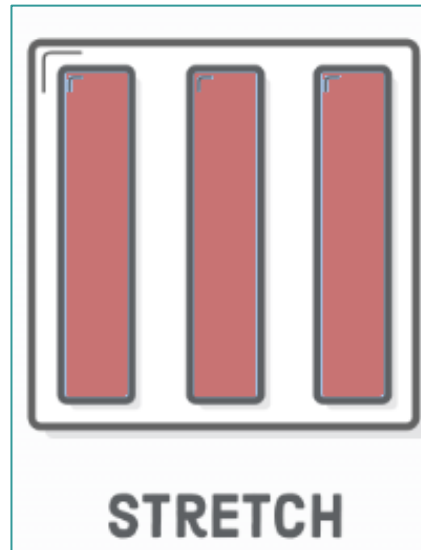
```
.container {  
  display: flex;  
  justify-content: space-evenly;  
  align-items: center;  
  width: 100%;  
  height: 500px;  
  border: 1px solid black;  
}
```

- Από πάνω προς τα κάτω στις εικόνες Flex-start | center | flex-end
- Με *stretch* καταλαμβάνουν όλο τον κάθετο χώρο. Αν στοιχίσουμε με *baseline*, η στοίχιση γίνεται με βάση τα περιεχόμενα





Align-items: stretch



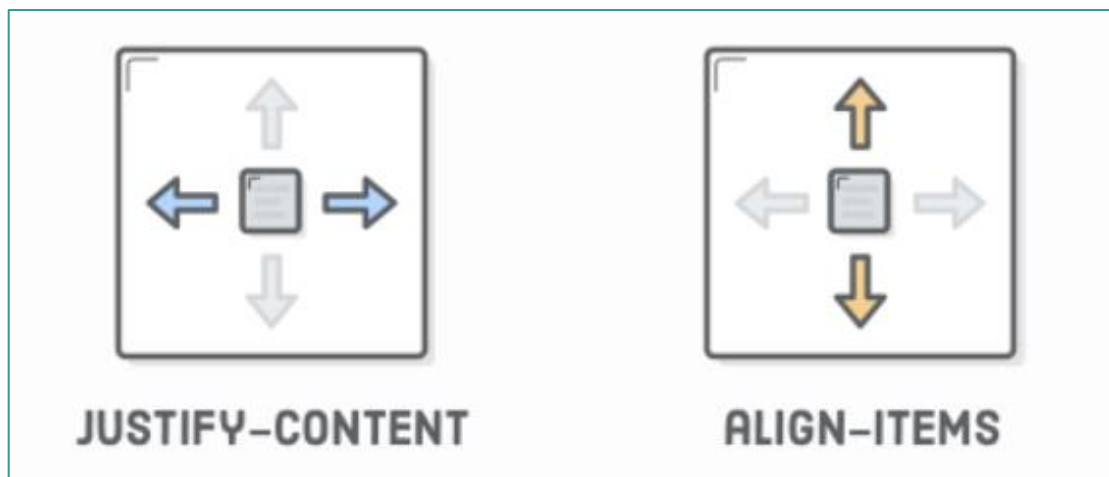
- Με **align-items: stretch** μπορούμε να δώσουμε ίδιο ύψος σε columns ανεξάρτητα από το περιεχόμενο κάτι πολύ δύσκολο να γίνει με floats



Στοιχίση Οριζόντια και Κάθετα

Προγραμματισμός στο Web

- Το Flexbox διαχειρίζεται μόνο τα άμεσα παιδιά του (one level deep). Κάθε HTML element που βρίσκεται μέσα σε ένα flex container είναι ένα 'flex item'
- Τα flex items μπορούμε να τα στοιχίζουμε όπως είδαμε οριζόντια με **justify-content** και κάθετα με **align-items**, όταν το direction είναι οριζόντιο





Grouping Flex Items (1)

Προγραμματισμός στο Web

```
13 .header {
14     display: flex;
15     /* overflow: hidden; */
16     background-color: #762124;
17     position: sticky;
18     top: 0;
19 }
20
21 /* .logo {
22     float: left;
23 } */
24
25 .logo > img {
26     width: auto;
27     height: 120px;
28     vertical-align: middle;
29 }
30
```

```
31 .nav {
32     /* float: left; */
33     width: 700px;
34 }
35
36 .nav > ul {
37     height: 120px;
38     list-style: none;
39     display: flex;
40     justify-content: space-around;
41     align-items: center;
42 }
```

- Το CSS του μενού της homepage μπορεί να υλοποιηθεί με Flexbox



Grouping Flex Items (2)

Προγραμματισμός στο Web

```
<div class="header">
  <div class="logo">
    
  </div>
  <div class="nav">
    <ul>
      <li><a href="">Home</a></li>
      <li><a href="">We Educate</a></li>
      <li><a href="">We Innovate</a></li>
      <li><a href="">We Are</a></li>
    </ul>
  </div>
```

- Έχουμε δύο επίπεδα flexbox: 1) στον header ώστε να στοιχίσουμε τα logo και nav, και 2) στο -που είναι μέσα στο header- ώστε να στοιχίσουμε τα



Wrapping Flex Items

- Ας υποθέσουμε ότι έχουμε πολλά εσωτερικά `<div>`. Για παράδειγμα έχουμε πολλά ``

```
<body>  
  <div class="container">  
    <div class="box1"></div>  
    <div class="box2"></div>  
    <div class="box3"></div>  
    <div class="box4"></div>  
    <div class="box5"></div>  
    <div class="box6"></div>  
    <div class="box7"></div>  
    <div class="box8"></div>  
    <div class="box9"></div>  
  </div>  
</body>
```

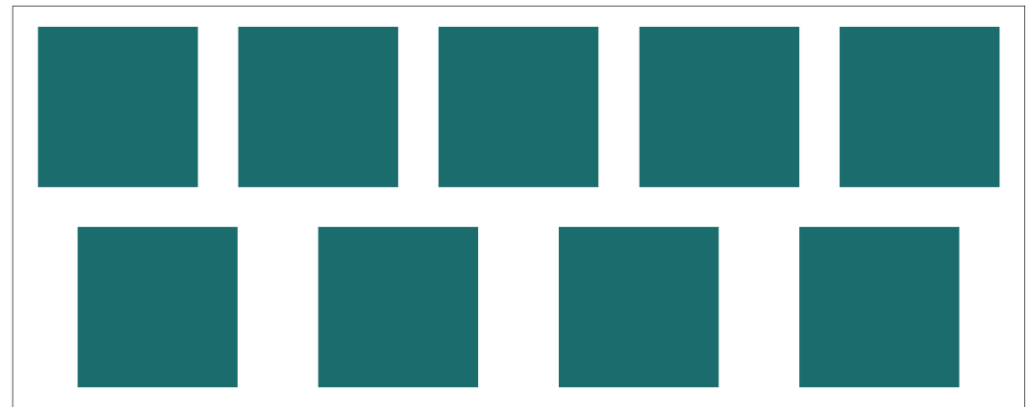


Παρατηρούμε ότι χαλάει το styling γιατί το flexbox προσπαθεί να χωρέσει όλα τα `<div>` σε ένα row



Flex-wrap

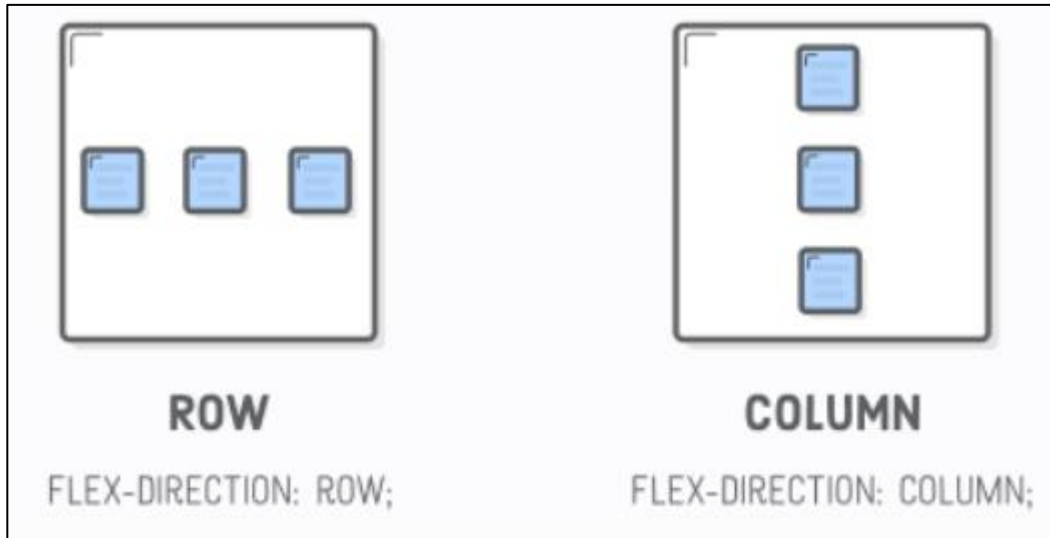
```
.container {  
  display: flex;  
  justify-content: space-evenly;  
  align-items: center;  
  width: 100%;  
  height: 500px;  
  border: 1px solid black;  
  flex-wrap: wrap;  
}
```



- Με **flex-wrap: wrap** μπορούν τα στοιχεία να στοιχιστούν σε περισσότερες από μία γραμμές

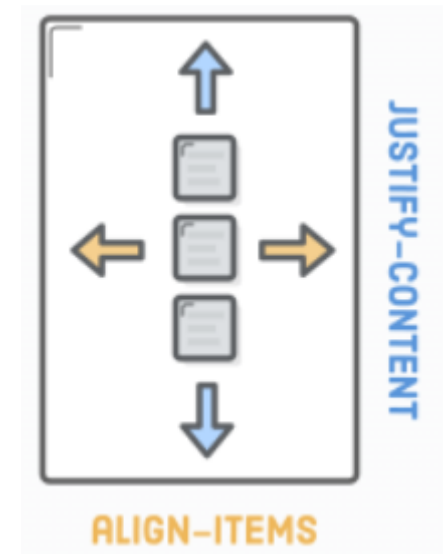


Flex-direction



To default flex-direction είναι row

- Όταν είναι flex-direction: column τα justify-content και align-items δουλεύουν αντίστροφα ως προς τον κύριο άξονα που τώρα είναι ο y





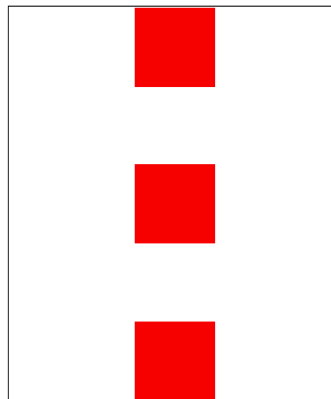
flex-direction: column

Προγραμματισμός στο Web

```
1  * {
2    box-sizing: border-box;
3  }
4  .container {
5    margin-left: auto;
6    margin-right: auto;
7    display: flex;
8    flex-direction: column;
9    justify-content: space-between;
10   align-items: center;
11   width: 250px;
12   height: 250px;
13   border: 1px solid black;
14   flex-wrap: wrap;
15 }
16
17 div[class^="box"] {
18   background-color: red;
19   width: 60px;
20   height: 50px;
21   text-align: center;
22   line-height: 50px;
23 }
```

```
<body>
  <div class="container">
    <div class="box1"></div>
    <div class="box2"></div>
    <div class="box3"></div>
  </div>
</body>
</html>
```

ry-REBOOT/webprojects/coding-factory/testbed/wrap.html



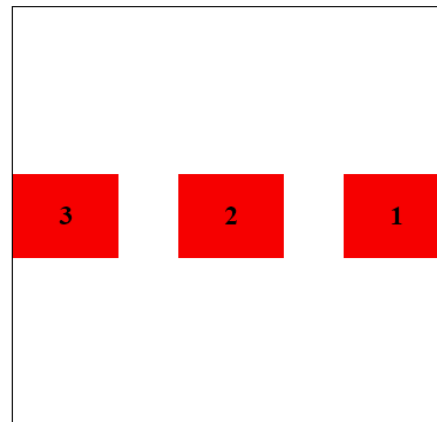


Flex-direction: row-reverse

```
1  * {
2    box-sizing: border-box;
3  }
4  .container {
5    margin-left: auto;
6    margin-right: auto;
7    display: flex;
8    flex-direction: row-reverse;
9    justify-content: space-between;
10   align-items: center;
11   width: 250px;
12   height: 250px;
13   border: 1px solid black;
14   flex-wrap: wrap;
15 }
16
17 div[class^="box"] {
18   font-weight: 900;
19   background-color: red;
20   width: 60px;
21   height: 50px;
22   text-align: center;
23   line-height: 50px;
24 }
```

```
<body>
  <div class="container">
    <div class="box1">1</div>
    <div class="box2">2</div>
    <div class="box3">3</div>
  </div>
</body>
</html>
```

~/REBOOT/webprojects/coding-factory/testbed/wrap.html



- Right-to-left ordering αντί left-to-right



flex-direction: column-reverse

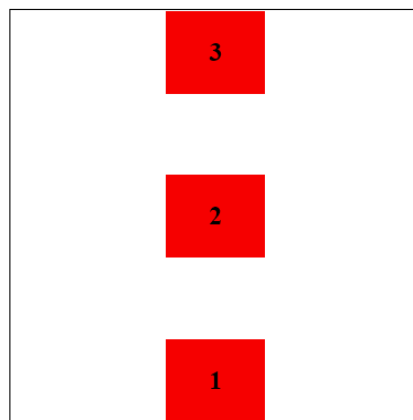
Προγραμματισμός στο Web

```
* {
  box-sizing: border-box;
}
.container {
  margin-left: auto;
  margin-right: auto;
  display: flex;
  flex-direction: column-reverse;
  justify-content: space-between;
  align-items: center;
  width: 250px;
  height: 250px;
  border: 1px solid black;
  flex-wrap: wrap;
}

div[class^="box"] {
  font-weight: 900;
  background-color: red;
  width: 60px;
  height: 50px;
  text-align: center;
  line-height: 50px;
}
```

```
<body>
  <div class="container">
    <div class="box1">1</div>
    <div class="box2">2</div>
    <div class="box3">3</div>
  </div>
</body>
</html>
```

EBOOT/webprojects/coding-factory/testbed/wrap.html

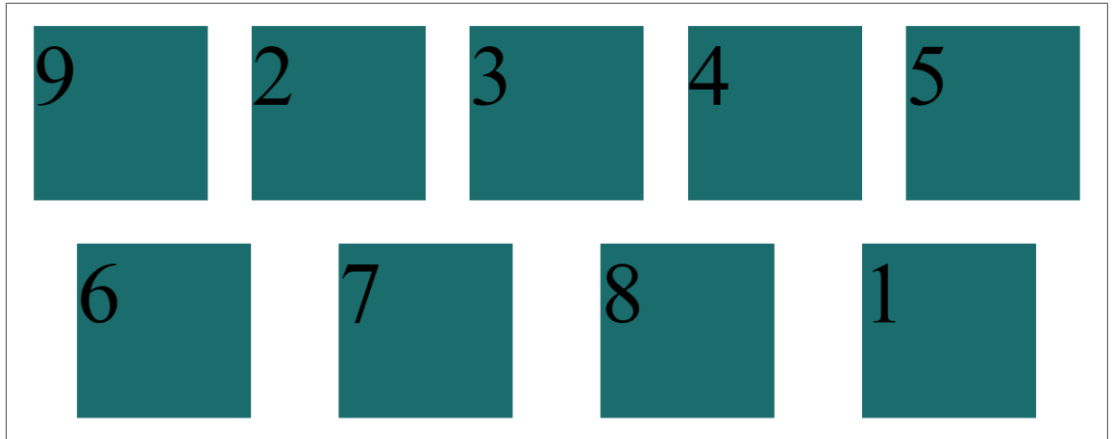


- bottom-up ordering αντί top-to-bottom



Order

```
.box1 {  
  order: 9;  
}  
  
.box9 {  
  order: -1;  
}
```



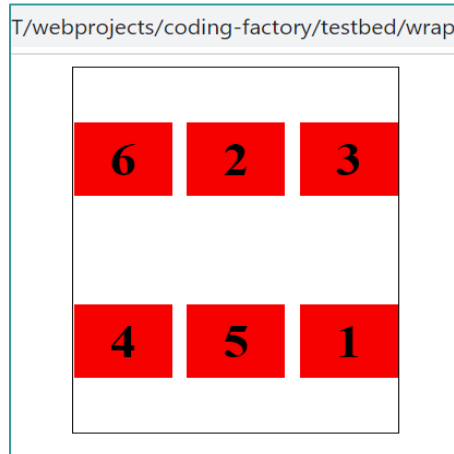
- Μέχρι τώρα είδαμε πως μπορούμε να **διαχειριστούμε τα flex items μέσω του parent container**. Μπορούμε όμως να **διαχειριστούμε και κάθε individual flex item**. Για παράδειγμα, το κάθε box μπορεί να ορίσει το order του
- Τη σειρά, δηλαδή, εμφάνισής του με την ιδιότητα order. Το default order είναι 0. Στην εικόνα έχουμε κάνει swap το 1^ο και το τελευταίο box



Swap items

```
5 .container {
6   margin-left: auto;
7   margin-right: auto;
8   display: flex;
9   flex-wrap: wrap;
10  justify-content: space-between;
11  align-items: center;
12  width: 200px;
13  height: 250px;
14  border: 1px solid black;
15 }
16
17 div[class^="box"] {
18   font-weight: 900;
19   background-color: red;
20   width: 60px;
21   height: 50px;
22   text-align: center;
23   line-height: 50px;
24   font-size: 32px;
25 }
26
27 .box1 {
28   order: 6;
29 }
30
31 .box6 {
32   order: -1;
33 }
```

```
<body>
  <div class="container">
    <div class="box1">1</div>
    <div class="box2">2</div>
    <div class="box3">3</div>
    <div class="box4">4</div>
    <div class="box5">5</div>
    <div class="box6">6</div>
  </div>
</body>
</html>
```



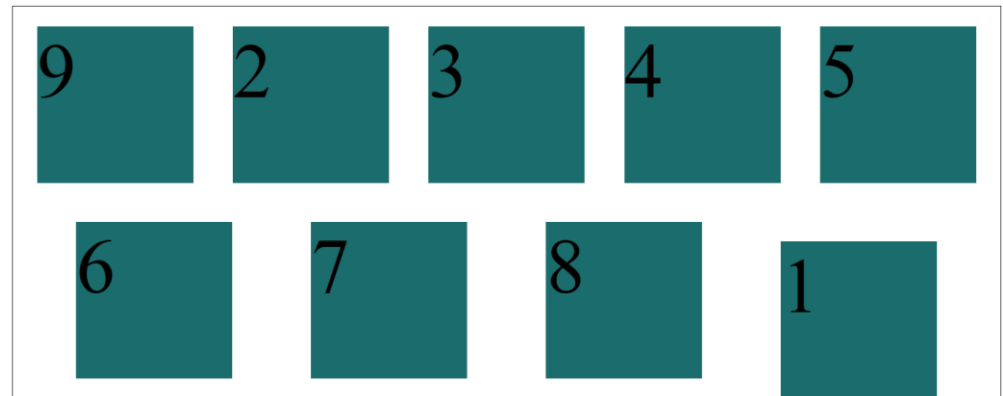
- Ανταλλάσσει αμοιβαία τα items 1 και 6



Align-self (1)

- Μπορεί το κάθε flex-item να στοιχίσει τον εαυτό του στην κάθετη διάσταση. Κάνει override το alignment του parent container

```
.box1 {  
  order: 9;  
  align-self: flex-end;  
}
```

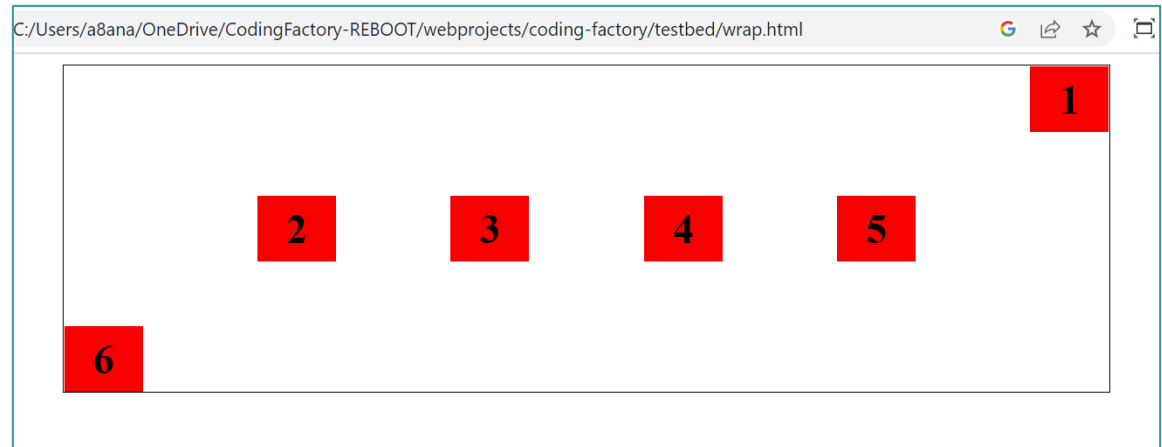




Align-self (2)

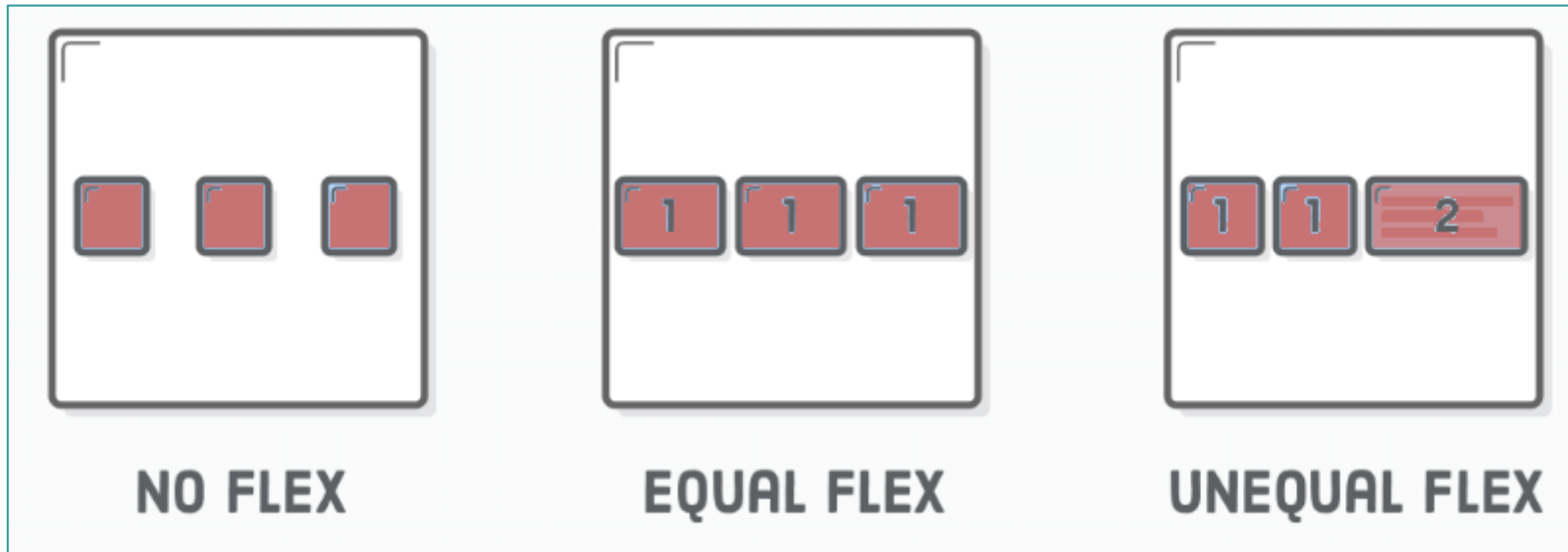
```
5 .container {
6   margin-left: auto;
7   margin-right: auto;
8   display: flex;
9   flex-wrap: wrap;
10  justify-content: space-between;
11  align-items: center;
12  width: 800px;
13  height: 250px;
14  border: 1px solid black;
15 }
16
17 div[class^="box"] {
18   font-weight: 900;
19   background-color: red;
20   width: 60px;
21   height: 50px;
22   text-align: center;
23   line-height: 50px;
24   font-size: 32px;
25 }
26
27 .box1 {
28   order: 6;
29   align-self: flex-start;
30 }
31
32 .box6 {
33   order: -1;
34   align-self: flex-end;
```

```
<body>
  <div class="container">
    <div class="box1">1</div>
    <div class="box2">2</div>
    <div class="box3">3</div>
    <div class="box4">4</div>
    <div class="box5">5</div>
    <div class="box6">6</div>
  </div>
</body>
</html>
```





Size των flex-items (2)



- Τα flex items είναι flexible! Μπορούν να αυξήσουν το μέγεθός τους πέρα από αυτό που έχουμε ορίσει εμείς με width **ώστε να καλύψουν τον ελεύθερο χώρο** Η ιδιότητα **flex** δίνει αυτή τη δυνατότητα
- Το flex: 1 έχει ένα default size, το flex: 2 το διπλάσιο, κοκ.

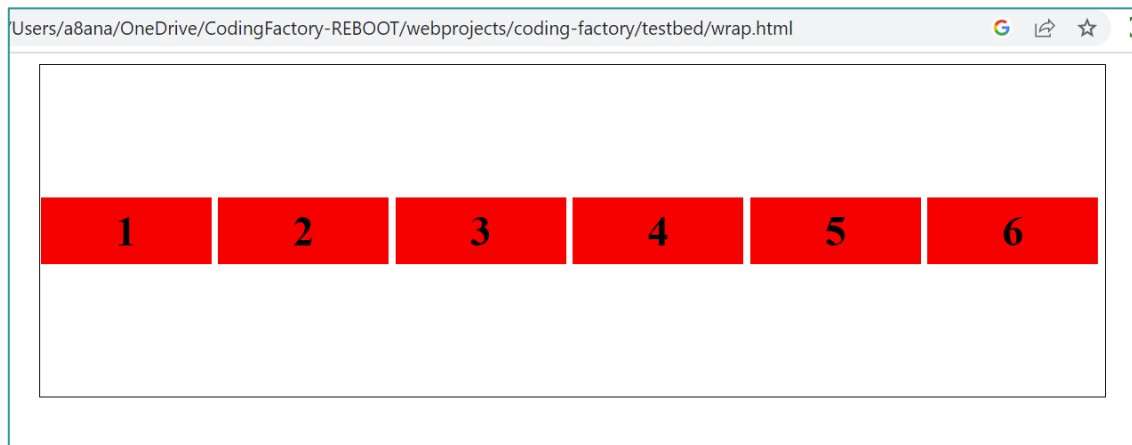


Size των flex-items (1)

Προγραμματισμός στο Web

```
1  * {  
2    box-sizing: border-box;  
3  }  
4  
5  .container {  
6    margin-left: auto;  
7    margin-right: auto;  
8    display: flex;  
9    flex-wrap: wrap;  
10   justify-content: space-between;  
11   align-items: center;  
12   width: 800px;  
13   height: 250px;  
14   border: 1px solid black;  
15 }  
16  
17 div[class^="box"] {  
18   font-weight: 900;  
19   background-color: red;  
20   width: 60px;  
21   height: 50px;  
22   text-align: center;  
23   line-height: 50px;  
24   font-size: 32px;  
25   flex: 1;  
26   margin-right: 5px;  
27 }
```

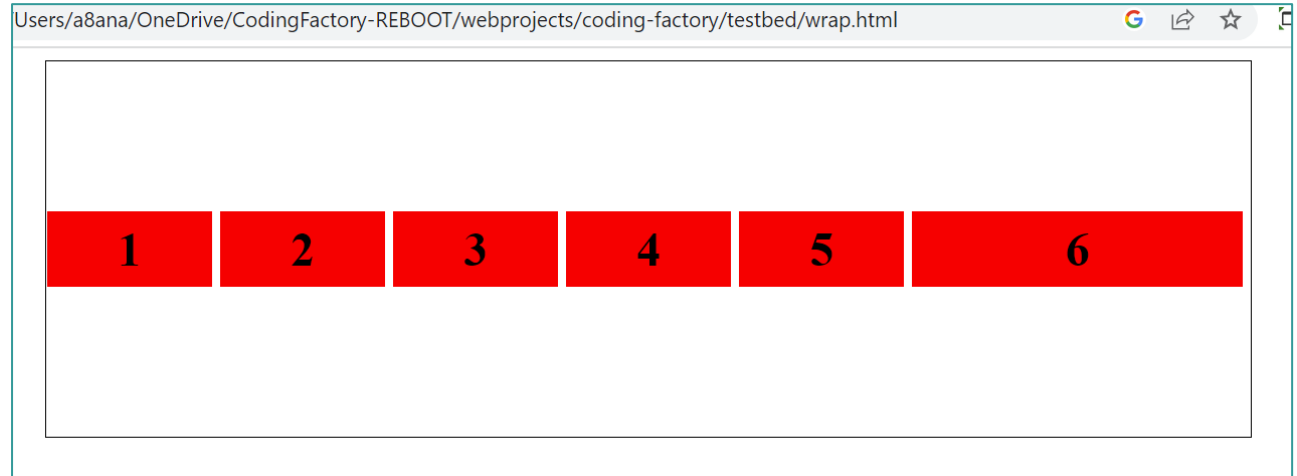
```
<body>  
  <div class="container">  
    <div class="box1">1</div>  
    <div class="box2">2</div>  
    <div class="box3">3</div>  
    <div class="box4">4</div>  
    <div class="box5">5</div>  
    <div class="box6">6</div>  
  </div>  
</body>  
</html>
```





Size των flex-items (2)

```
29  .box1, .box2,  
30  .box3, .box4,  
31  .box5 {  
32      flex: 1;  
33  }  
34  
35  .box6 {  
36      flex: 2;  
37  }
```



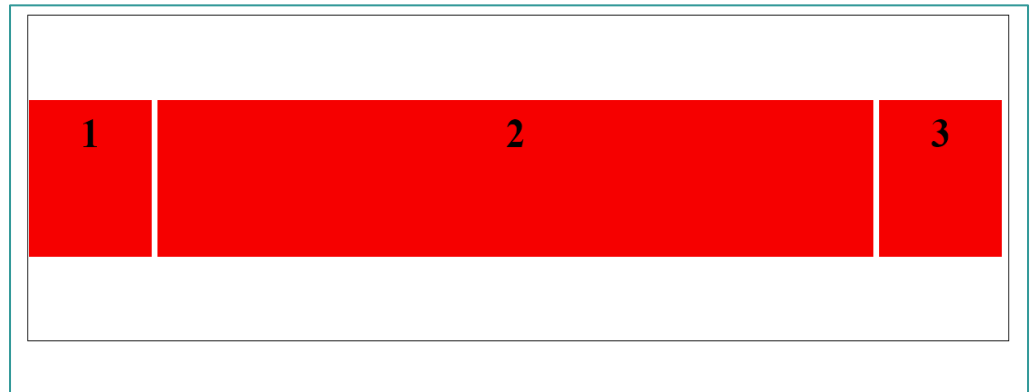


Συνδυασμός fixed width & flex

Προγραμματισμός στο Web

```
17  div[class^="box"] {
18      font-weight: 900;
19      background-color: red;
20      width: 100px;
21      height: 120px;
22      text-align: center;
23      line-height: 50px;
24      font-size: 32px;
25      /* flex: 1; */
26      margin-right: 5px;
27  }
28
29  .box1, .box3{
30      flex: initial;
31  }
32
33  .box2 {
34      flex: 1;
35  }
36
```

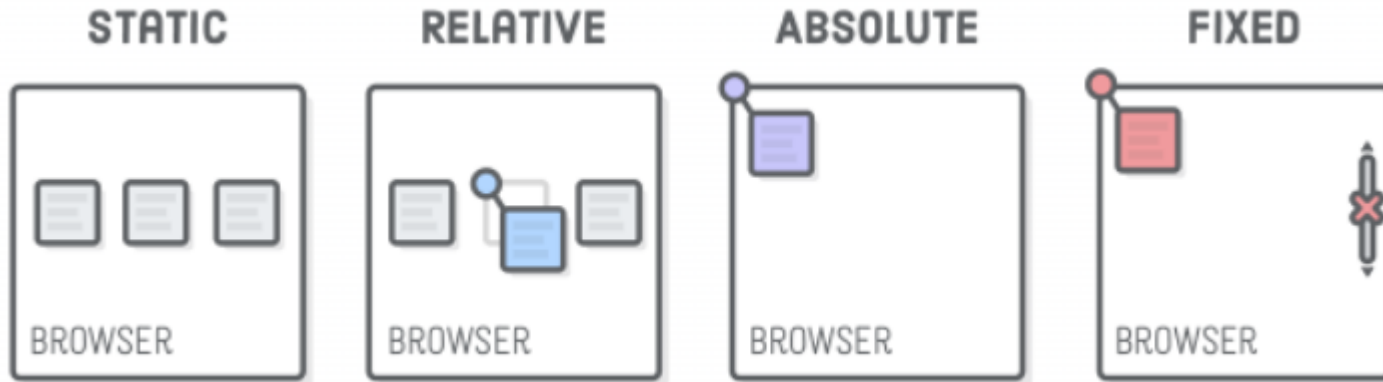
```
<body>
  <div class="container">
    <div class="box1">1</div>
    <div class="box2">2</div>
    <div class="box3">3</div>
  </div>
</body>
</html>
```





Advanced Positioning

Προγραμματισμός στο Web

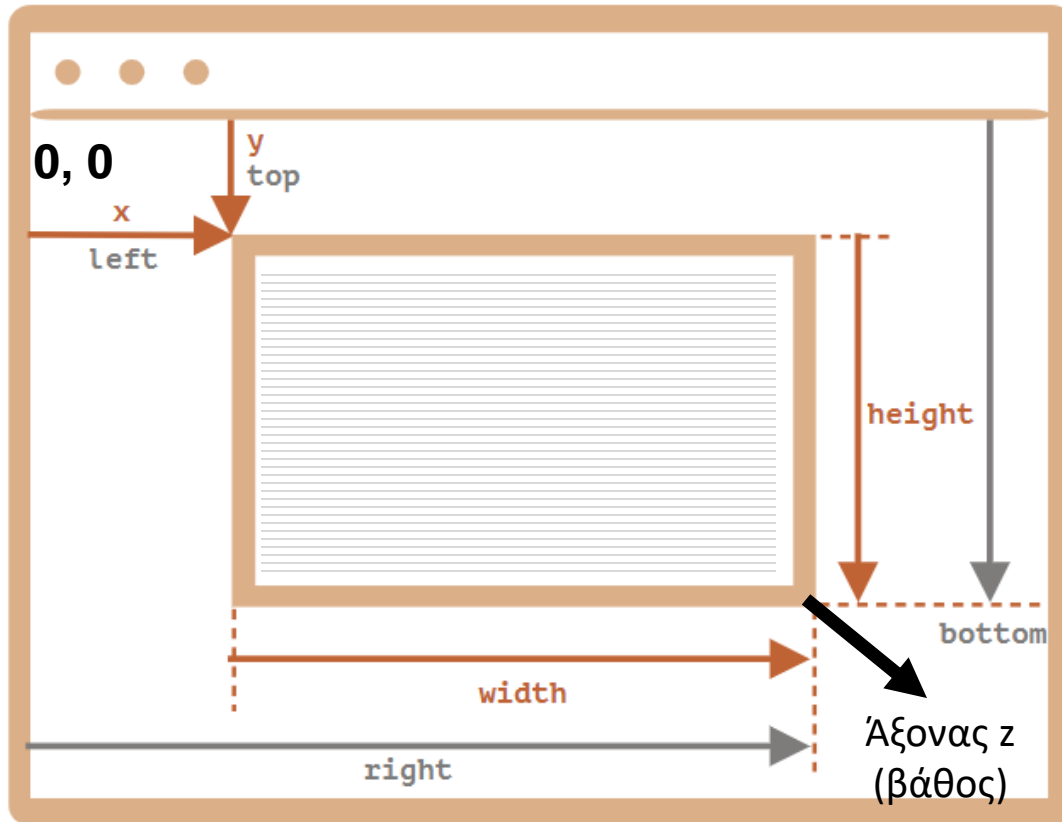


- **Static Positioning** = Normal Flow
 - CSS Box Model, Floats, Flexbox
- Ακόμα τέσσερις τύποι positioning
 - **Relative, Absolute, Fixed, Sticky.** Μας επιτρέπουν να τοποθετήσουμε τα HTML στοιχεία manually με συγκεκριμένες συντεταγμένες (π.χ. με pixels)



Window Coordinates System

Προγραμματισμός στο Web



Αν τοποθετήσουμε ένα στοιχείο με positioning absolute, relative, fixed, sticky φεύγει από το normal flow

- Η πάνω αριστερή γωνία του παραθύρου έχει συντεταγμένες $x=0, y=0$
- Το x (οριζόντια διάσταση) το τοποθετούμε με left / right
- Το y (κάθετη διάσταση) το τοποθετούμε με top / bottom
- Το z (βάθος) το τοποθετούμε με την ιδιότητα z-index. Το **z-index** δουλεύει μόνο για **positioned** και **flex** στοιχεία. Έχει default τιμή το 0



Positioning Systems

Προγραμματισμός στο Web

- Συνήθως χρησιμοποιούμε το default static positioning
- Τα **άλλα positioning σχήματα** χρησιμοποιούνται όταν θέλουμε να κάνουμε πιο **advanced λειτουργίες**, όπως να τροποποιήσουμε τη θέση ενός αντικειμένου, να το διατηρήσουμε σταθερό σε σχέση με άλλα αντικείμενα (π.χ. dropdown menu) χωρίς να διαταράξουμε τα υπόλοιπα UI στοιχεία

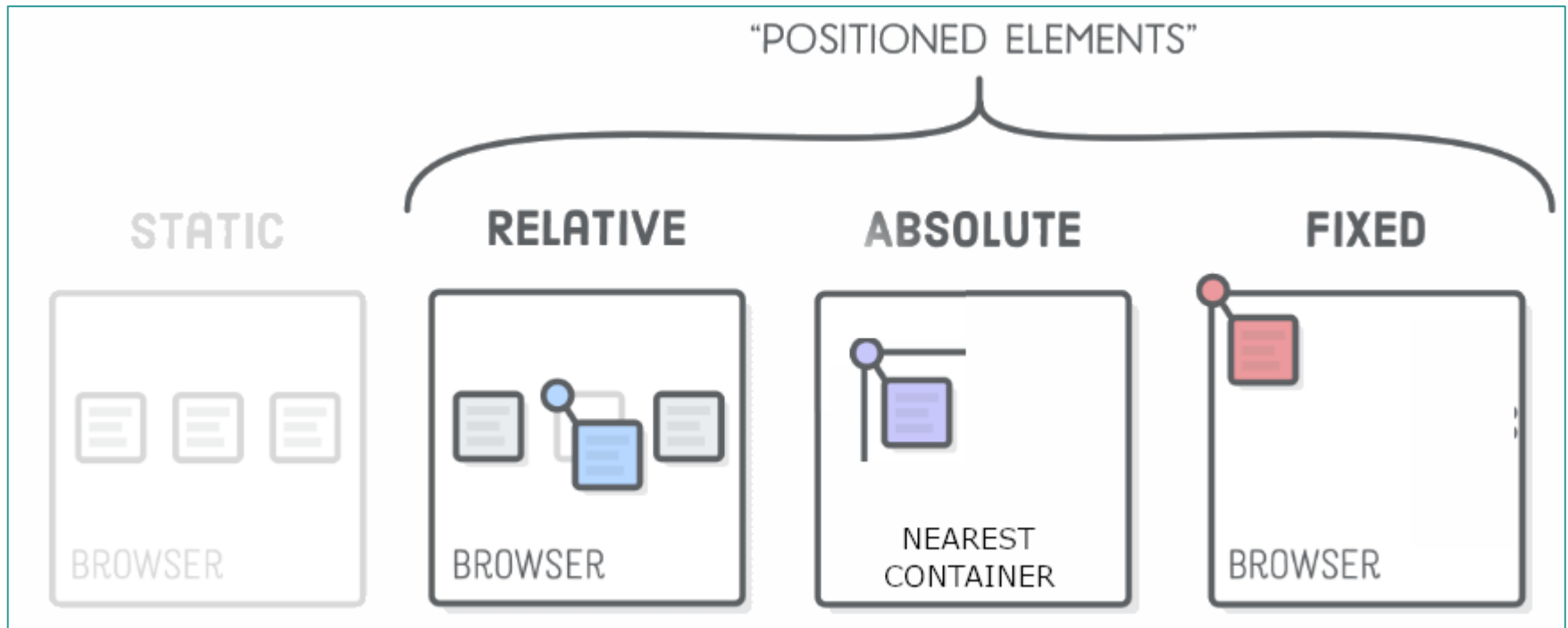


Positioned elements (1)

- Η CSS position property (π.χ. position: relative ή position: absolute ή position: fixed ή position: sticky) μας επιτρέπει να αλλάξουμε το positioning ενός HTML αντικειμένου και να 'βγάλουμε' το στοιχείο αυτό από το normal flow (σσ. position: static) και να το τοποθετήσουμε σε κάποιο συγκεκριμένο σημείο
- Όταν το position ενός στοιχείου UI δεν είναι static τότε το στοιχείο ονομάζεται 'positioned element'



Positioned elements (2)



- Για να είναι το absolute positioning, relative to nearest container θα πρέπει ο nearest container να είναι relative, αλλιώς γίνεται fall back σε σχέση με τον browser όπως και το fixed



Βασικό Template

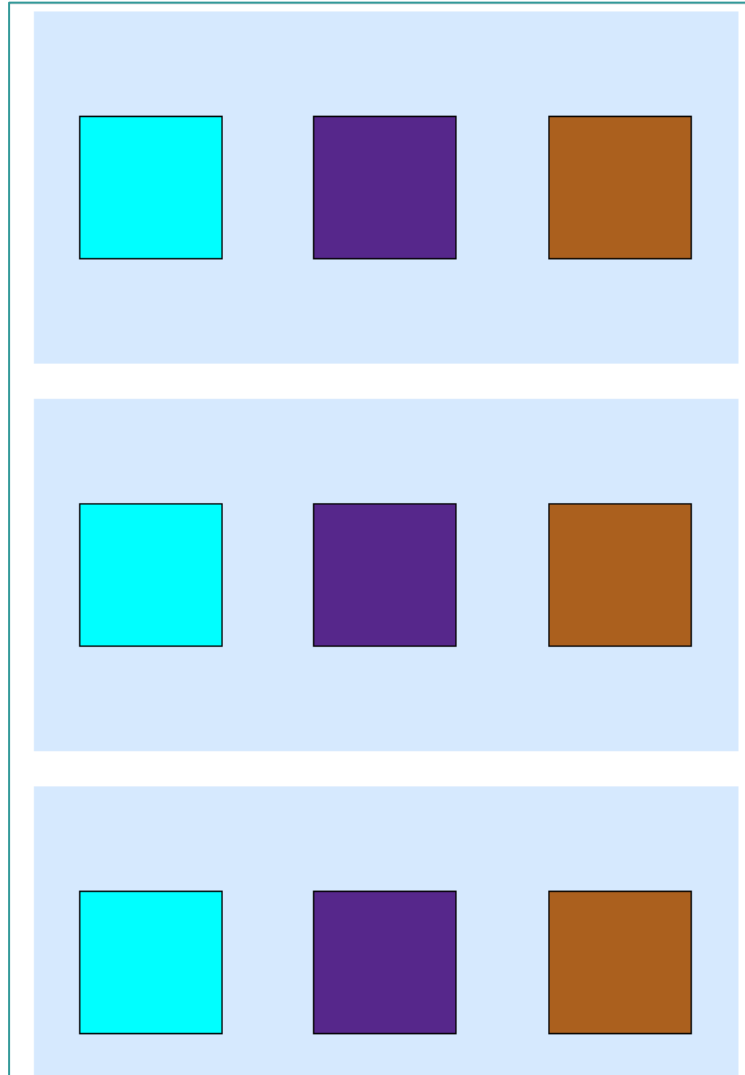
```
53 <body>
54   <div class='container'>
55     <div class='box1'></div>
56     <div class='box2'></div>
57     <div class='box3'></div>
58   </div>
59
60   <div class='container'>
61     <div class='box1'></div>
62     <div class='box2'></div>
63     <div class='box3'></div>
64   </div>
65
66   <div class='container'>
67     <div class='box1'></div>
68     <div class='box2'></div>
69     <div class='box3'></div>
70   </div>
71
72   <div class='container'>
73     <div class='box1'></div>
74     <div class='box2'></div>
75     <div class='box3'></div>
76   </div>
77 </body>
```

- Έστω το template αριστερά που θα χρησιμοποιήσουμε για να εφαρμόσουμε τους διάφορους τύπους positioning



Template & CSS

```
.container {  
  margin: 0 auto;  
  display: flex;  
  justify-content: space-around;  
  align-items: center;  
  background-color: #D6E9FE;  
  margin-bottom: 40px;  
  width: 800px;  
  height: 400px;  
}  
  
.box1 {  
  border: 1px solid black;  
  padding: 80px;  
  background-color: aqua;  
}  
  
.box2 {  
  border: 1px solid black;  
  padding: 80px;  
  background-color: rgb(86, 39, 139);  
}  
  
.box3 {  
  border: 1px solid black;  
  padding: 80px;  
  background-color: rgb(171, 96, 30);  
}
```

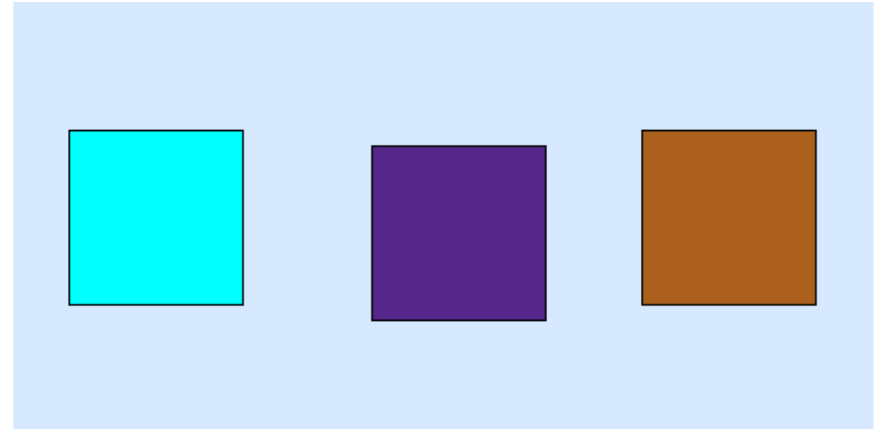


- Και το αντίστοιχο CSS που δίνει δομή και χρώμα στα 'κουτιά'



Relative

```
<body>
  <div class='container'>
    <div class='box1'></div>
    <div class='box2 item-relative'></div>
    <div class='box3'></div>
  </div>
```



- Το relative positioning μεταθέτει το αντικείμενο σε νέα θέση (offset) **σχετικά με την θέση που θα είχε στο normal flow** (μπορεί να λάβει και αρνητικές τιμές)
- Είναι σαν το margin αλλά δεν επηρεάζονται τα υπόλοιπα UI elements

```
.item-relative {
  position: relative;
  top: 15px;
  left: 15px;
}
```



Absolute (1)

```
<body>
<div class='container'>

  <div class='box1'></div>
  <div class='box2 item-relative'></div>
  <div class='box3'></div>

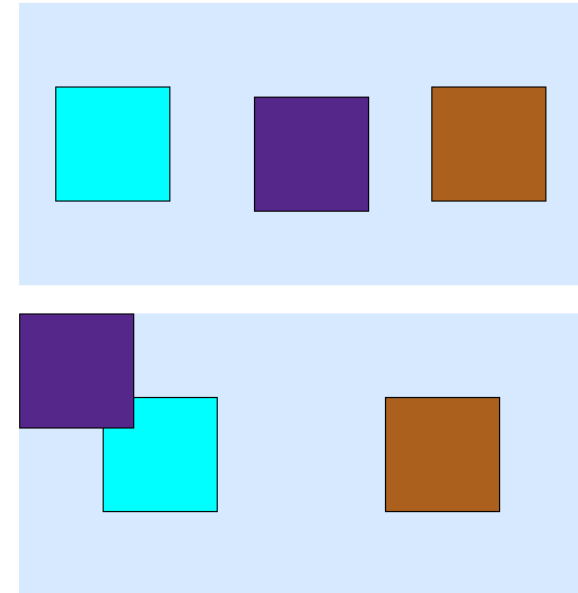
</div>

<div class='container'>

  <div class='box1'></div>
  <div class='box2 item-absolute'></div>
  <div class='box3'></div>

</div>
```

```
.container {
  position: relative;
  margin: 0 auto;
  display: flex;
  justify-content: space-around;
  align-items: center;
  background-color: #D6E9FE;
  margin-bottom: 40px;
  width: 800px;
  height: 400px;
}
```



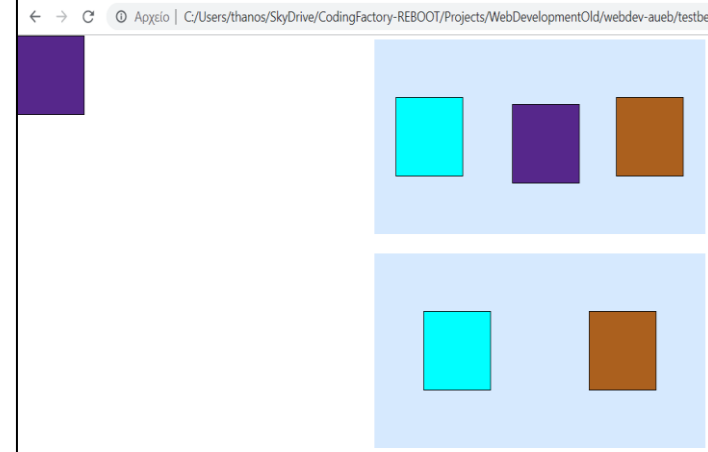
- Το Absolute Positioning πάντα **πρέπει να είναι relative σε σχέση με κάποιο parent container**, όπως εδώ. Αν δεν δώσουμε ρητά relative σε κάποιο parent element, τότε στοιχίζεται σε σχέση με τον browser/viewport
- Αν κάνουμε **scroll** τότε το positioned element κινείται κανονικά σε σχέση με το relative element



Absolute (1)

```
.container {  
  margin: 0 auto;  
  display: flex;  
  justify-content: space-around;  
  align-items: center;  
  background-color: #D6E9FE;  
  margin-bottom: 40px;  
  width: 800px;  
  height: 400px;  
}
```

```
.item-absolute {  
  position: absolute;  
  top: 0;  
  left: 0;  
}
```



- Ο parent container δεν είναι relative, οπότε στοιχίζεται σε σχέση με τον browser
- Και εδώ κάνει scroll και κινείται το στοιχείο



Fixed Positioning

- Είναι ίδιο με το προηγούμενο absolute positioning σε σχέση με τον browser, **μόνο που το κόκκινο box στοιχείο δεν κάνει scroll**, αλλά παραμένει στην θέση του
- Το Μωβ box στην αριστερή πάνω γωνία του browser που είναι absolute κινείται με το scroll

```
<div class='container'>  
  
  <div class='box1'></div>  
  <div class='box2 bg-red item-fixed'></div>  
  <div class='box3'></div>  
  
</div>
```





Z-index

```
.z-front {  
  z-index: 1;  
}  
  
</style>  
</head>  
<body>  
  <div class='container'>  
  
    <div class='box1'></div>  
    <div class='box2 item-relative'></div>  
    <div class='box3'></div>  
  
  </div>  
  
  <div class='container'>  
  
    <div class='box1'></div>  
    <div class='box2 z-front item-absolute'></div>  
    <div class='box3'></div>  
  
  </div>
```

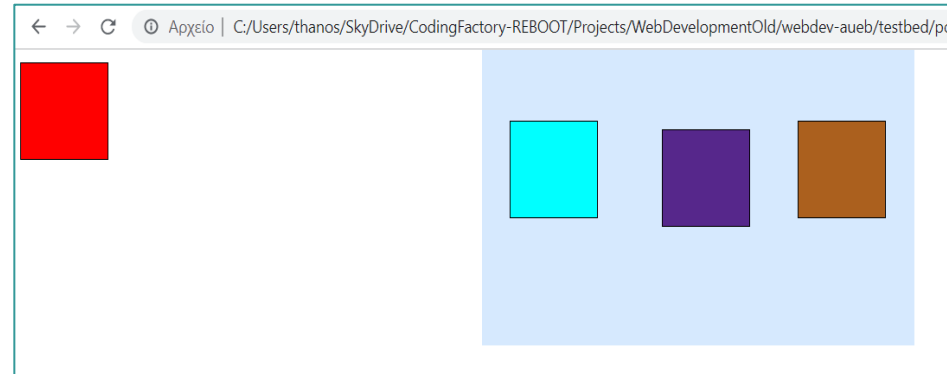


- Η ιδιότητα z-index προσαρμόζει το 'βάθος' του αντικειμένου (default τιμή 0)
- Οποιαδήποτε θετική τιμή ανεβάζει το αντικείμενο πιο μπροστά στη z-διάσταση, όπως το μωβ κουτί στο παράδειγμα



Sticky (relative + fixed)

```
.item-sticky {  
  position: -webkit-sticky;  
  position: sticky;  
  top: 0;  
}  
  
body {  
  height: 2000px;  
}  
</style>  
</head>  
<body>  
  <div class='container item-sticky'>  
    <div class='box1'></div>  
    <div class='box2 item-relative'></div>  
    <div class='box3'></div>  
  </div>
```



- Είναι **relative** με το viewport μέχρι το σημείο που ορίζουμε (π.χ. top: 0) και κάνει scroll ενώ μετά γίνεται **fixed** (δεν κάνει scroll)



Vendor prefixes

```
.sticky {  
  position: -webkit-sticky;  
  position: -moz-sticky;  
  position: -ms-sticky;  
  position: -o-sticky;  
  top: 15px;  
}
```

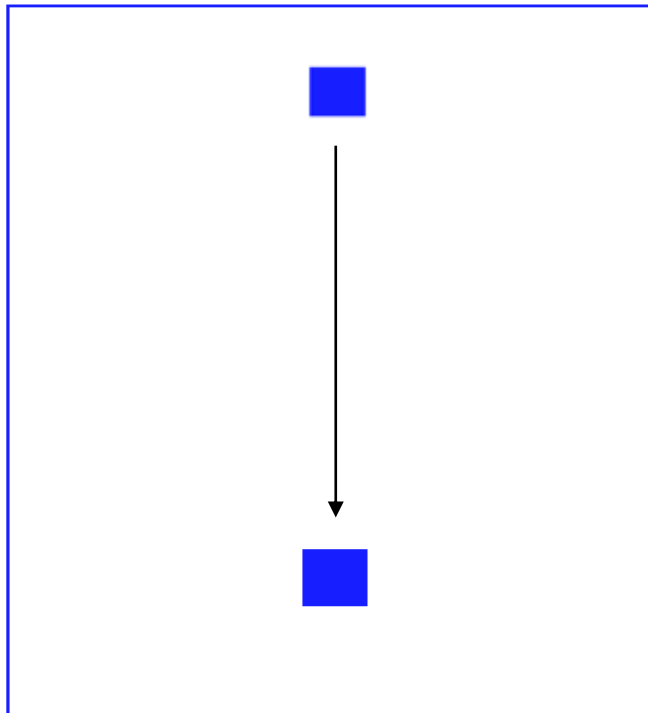
Safari (WebKit) / Chrome-MS Edge (Blink)
Mozilla Firefox (Gecko)
Internet Explorer (Trident)
Opera (Presto)

- Τα **Vendor prefixes** χρησιμοποιούνται όταν κάποιος (old) browser δεν υποστηρίζει κάποιες νέες CSS ιδιότητες
- Θα ήταν καλύτερα οι χρήστες να ανανεώνουν τους browsers και οι vendors να υποστηρίζουν όλες τα CSS ιδιότητες
- Το Blink (browser engine) ξεκίνησε ως fork του WebKit.



Animation (1)

- Ένα από τα use cases του relative και absolute positioning είναι το animation



- Έστω ότι θέλουμε να κινήσουμε ένα box από μία αρχική θέση στο πάνω μέρος του container προς τα κάτω



Animation – HTML/JavaScript

Προγραμματισμός στο Web

```
7 <title>Animation</title>
8 <link rel="stylesheet" href="./css/absolute-animation.css">
9 </head>
10 <body>
11 <div class="container">
12 |   <div class="item-abs"></div>
13 </div>
14
15 <script>
16 |   const debounceTimeout = setInterval(moveBox, 10)
17 |   let topPos = 10;
18
19 |   function moveBox() {
20 |     const box = document.querySelector('.item-abs')
21
22 |     box.style.backgroundColor = 'blue'
23
24 |     topPos += 5
25 |     box.style.top = topPos + 'px'
26 |     if (topPos >= 380) {
27 |       clearInterval(debounceTimeout)
28 |     }
29
30 |     console.log("Offset from top: " + topPos)
31 |   }
32 </script>
33 </body>
34 </html>
```

- Πρόκειται για ένα απλό πρόγραμμα JavaScript όπου με την `setInterval()` ορίζουμε την εκτέλεση μίας συνάρτησης, της `moveBox()` κάθε 10 milliseconds
- Η `setInterval` επιστρέφει ένα timer id (`debounceTimeout`)
- Για να κάνουμε cancel το repeated task, κάνουμε `clearInterval()` με παράμετρο το timer id
- Η JavaScript δεν έχει strictly typed system, δεν έχει static policy, αλλά dynamic policy (δεν έχουμε δηλώσεις με τύπους όπως `int`, κλπ.) αλλά με `let`, `const` που εκτελούνται @runtime



JavaScript

```
7 <title>Animation</title>
8 <link rel="stylesheet" href="./css/absolute-animation.css">
9 </head>
10 <body>
11 <div class="container">
12 |   <div class="item-abs"></div>
13 | </div>
14
15 <script>
16 |   const debounceTimeout = setInterval(moveBox, 10)
17 |   let topPos = 10;
18
19 |   function moveBox() {
20 |     const box = document.querySelector('.item-abs')
21 |
22 |     box.style.backgroundColor = 'blue'
23 |
24 |     topPos += 5
25 |     box.style.top = topPos + 'px'
26 |     if (topPos >= 380) {
27 |       clearInterval(debounceTimeout)
28 |     }
29 |
30 |     console.log("Offset from top: " + topPos)
31 |   }
32 </script>
33 </body>
34 </html>
```

Προγραμματιστική
αντιστοίχιση UI
elements κάνουμε με
document.querySelector
(css-selector)

Με .style.<css-property>
ορίζουμε/αλλάζουμε
CSS ιδιότητες @runtime

Με console.log()
γράφουμε στην κονσόλα
(όχι στον Browser)



Animation - CSS

Προγραμματισμός στο Web

- Στο CSS file δίνουμε το αρχικό styling το οποίο μπορεί να τροποποιηθεί @runtime με JavaScript

```
1  * {  
2    box-sizing: border-box;  
3  }  
4  
5  .container {  
6    margin-left: auto;  
7    margin-right: auto;  
8    width: 400px;  
9    height: 500px;  
10   border: 2px solid blue;  
11   position: relative;  
12 }  
13  
14 .item-abs {  
15   position: absolute;  
16   top: 10px;  
17   left: 180px;  
18   width: 40px;  
19   height: 40px;  
20   background-color: red;  
21 }
```



Μενού/Dropdown Υπομενού

Προγραμματισμός στο Web

```
<body>
  <div class="container">
    <div class="header">
      <div class="logo">
        
      </div>
      <div class="navbar">
        <ul class="nav">
          <li class="dropdown"><a href="javascript:void(0)">Home &#9662</a>
            <ul class="dropdown-content">
              <li><a href="#">Εκπαίδευση</a></li>
              <li><a href="#">Έρευνα</a></li>
              <li><a href="#">Υπηρεσίες</a></li>
              <li><a href="#">Διασύνδεση</a></li>
            </ul>
          </li>
          <li><a href="#">We Educate</a></li>
          <li><a href="#">We Innovate</a></li>
          <li><a href="#">We Are</a></li>
        </ul>
      </div>
    </div>
  </div>
```

- Έχουμε κάνει wrap το submenu μέσα σε ένα `` με κλάση `dropdown`
- Το `javascript: void(0)` χρησιμοποιείται ως URL placeholder για να δείξει ότι κάποιο onclick (ή onhover εδώ) event είναι συνδεδεμένο με το link
- Το `` είναι το submenu με κλάση `dropdown-content`



Βασικό Styling

```
1  * {
2    box-sizing: border-box;
3    margin: 0;
4    padding: 0;
5  }
6
7  .container {
8    width: 98%;
9    margin-left: auto;
10   margin-right: auto;
11 }
12
13 .header {
14   display: flex;
15   justify-content: space-between;
16   align-items: center;
17   background-color: #762124;
18   position: sticky;
19   top: 0;
20 }
21
22 .logo > img {
23   width: auto;
24   height: 120px;
25   vertical-align: middle;
26 }
27
```

```
28 .navbar {
29   width: 700px;
30 }
31
32 .navbar > ul {
33   height: 120px;
34   list-style: none;
35   display: flex;
36   justify-content: space-around;
37   align-items: center;
38 }
39
40 .navbar a {
41   display: block;
42   color: white;
43   text-decoration: none;
44   font-size: 1.2rem;
45   padding: 5px 10px;
46 }
47
48 .navbar a:active {
49   background-color: rgb(172, 172, 231);
50 }
51
52 .nav > li:not(:first-child) > a:hover {
53   background-color: rgb(116, 149, 206);
54 }
55
```



Dropdown Styling (1)

```
56 .dropdown {
57     position: relative;
58     color: white;
59     font-size: 1.2rem;
60     padding: 5px 10px;
61
62 }
63
64 .dropdown-content {
65     display: none;
66     width: 160px;
67     position: absolute;
68     top: 0;
69     left: -10px;
70     padding-top: 50px;
71     background-color: #618cbd;
72     list-style: none;
73     flex-direction: column;
74     justify-content: space-around;
75     align-items: flex-start;
76 }
77
78 .dropdown-content > li {
79     width: 100%;
80     border-bottom: 1px solid white;
81 }
```

- Το dropdown-content πρέπει να είναι **absolute** και να βγει από το normal flow γιατί διαφορετικά χαλάει όλα τα διπλανά UI elements
- Επίσης δίνουμε top, left και padding-top καθώς και display: flex
- Παρατηρήστε το **display: none** (γρ. 65) μιας και αρχικά το submenu δεν θέλουμε να φαίνεται
- Επίσης το dropdown-content είναι **flex-direction: column**
- Το dropdown επίσης πρέπει να είναι **relative** για να πάει το dropdown-content σε σχέση με το dropdown
- Τα μέσα στο dropdown-content έχουν width: 100% για να δώσουμε στη συνέχεια και στα <a> width: 100% (του) οπότε το hover να έχει effect σε όλα το πλάτος του . Επίσης έχουν border-bottom



Dropdown Styling (2)

```
83 .dropdown-content > li:last-of-type {
84     border-bottom: none;
85 }
86
87 .dropdown-content a {
88     display: block;
89     width: 100%;
90     color: white;
91     text-decoration: none;
92     font-size: 1.2rem;
93     padding: 10px 10px;
94 }
95
96 .dropdown-content a:hover {
97     background-color: rgb(22, 152, 109);
98 }
99
100 .dropdown > a {
101     position: relative;
102     z-index: 1;
103 }
104
105 .dropdown:hover > ul {
106     display: flex;
107 }
```

- Στο τελευταίο δεν δίνουμε border-bottom
- Δίνουμε styling στα <a> και a:hover
- Στο dropdown > a (δηλαδή στο <a> με το javascript:void(0) δίνουμε z-index: 1 γιατί καλύπτεται από το padding-top του dropdown-content
- Για να δουλέψει το z-index πρέπει το στοιχείο να είναι positioned, οπότε δίνουμε position: relative
- Επίσης, στο mouse-over (hover) πάνω από το με κλάση dropdown θα πρέπει να κάνουμε display: flex οπότε να εμφανίζεται το που ήταν display: none



CSS Transitions & Transforms

Προγραμματισμός στο Web

- Αφορούν σε movements και απλά animations ώστε να προσθέσουμε αλληλεπίδραση με τον χρήστη και να αυξήσουμε τη λειτουργικότητα ή την εμπειρία του χρήστη
- Οι οποιεσδήποτε κινήσεις θα πρέπει να αποσπούν την προσοχή του χρήστη από το βασικό στόχο της σελίδας



Transforms / Transitions


Προγραμματισμός στο Web

- Transforms. Μετακινήσεις ή αλλαγή της εμφάνισης ενός HTML στοιχείου
- Transitions. Περισσότερες επιλογές κίνησης από μία κατάσταση σε μία άλλη



Transitions

```
transition: [property] [duration] [timing-function] [delay];
```

- Shorthand μορφή 
- Αρχικά προσδιορίζουμε την CSS ιδιότητα που θέλουμε να εφαρμόσουμε το transition (π.χ. width, height, κλπ. ή all για όλες τις ιδιότητες)
- Τη διάρκεια σε seconds (ή milliseconds)
- Το εφέ (timing function) όπως ease, ease-in, ease-out, ease-in-out, linear, cubic-bezier, steps



Απλά Transitions

Προγραμματισμός στο Web

```
.box1 {  
  width: 200px;  
  height: 200px;  
  background-color: red;  
  transition: width 2.15s ease;  
}
```

```
.box1:hover {  
  width: 500px;  
}
```

```
.box1 {  
  width: 200px;  
  height: 200px;  
  background-color: red;  
  transition: width 2.15s ease-in;  
}
```

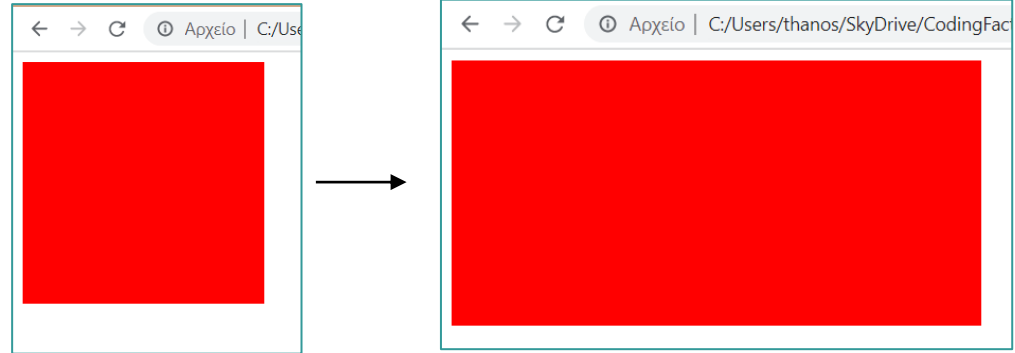
```
.box1:hover {  
  width: 500px;  
}
```

- Το default είναι το ease (Αργά-Γρήγορα-Αργά)
- Το easy-in ξεκινάει αργά και μετά πάει γρήγορα μέχρι το τέλος
- Το trigger στο παράδειγμα είναι το hover. Αν φύγει το hover ξαναγυρνάει στην αρχική κατάσταση με τον ίδιο τρόπο



Transitions

```
.box1 {  
  width: 200px;  
  height: 200px;  
  background-color: red;  
  transition: width 2.15s ease-in;  
}  
  
.box1:hover {  
  width: 500px;  
  transition: width 4s steps(6, end);  
}
```



- Ξεκινάει στο hover με 6 steps σε 4s μέχρι το τέλος και επιστρέφει μετά το hover με ease-in σε 2.15s



Transform

- Rotate, translate (move), skew, scale
- 2D και 3D
- Triggered (πυροδοτούνται) όταν το CSS στοιχείο αλλάξει κατάσταση (state)



Scale

- Αυξάνει / μειώνει το size
- Στο παράδειγμα αυξάνει το width στο 3πλάσιο

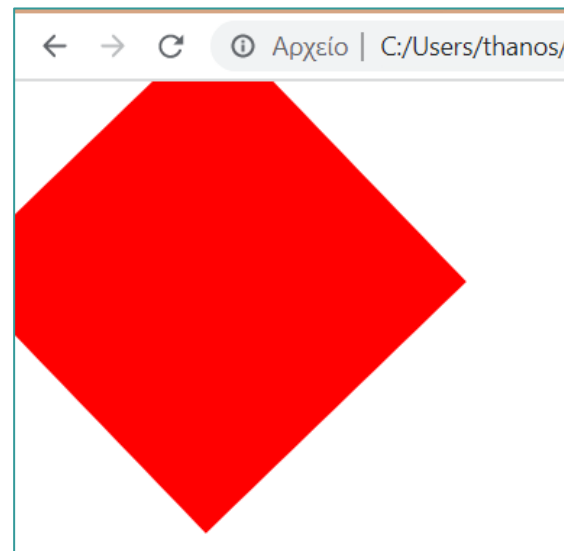
```
.box1 {  
  width: 200px;  
  height: 200px;  
  background-color: red;  
  transition: transform 2.15s ease-in;  
}  
  
.box1:hover {  
  transform: scale(3);  
}
```



Rotate

- Δεξιόστροφη περιστροφή με βάση μοίρες. Για παράδειγμα 90deg περιστρέφει 90 μοίρες δεξιόστροφα, ενώ -90deg περιστρέφει αριστερόστροφα 90 μοίρες

```
.box1 {  
  width: 200px;  
  height: 200px;  
  background-color: red;  
  transition: transform 2.15s ease-in;  
}  
  
.box1:hover {  
  transform: rotate(180deg);  
}
```





Translate (Move)

- Left / right / up / down (Θετικές τιμές δεξιά και κάτω, όπως στο παράδειγμα, αρνητικές αριστερά και πάνω)

```
.box1 {  
  width: 200px;  
  height: 200px;  
  background-color: red;  
  transition: transform 2.15s;  
}  
  
.box1:hover {  
  transform: translate(20px, 20px);  
}
```



Skew (Στρέβλωση)

Προγραμματισμός στο Web

- Θετικό X δίνει αριστερή κλίση / Αρνητικό X δεξιά κλίση
- Αρνητικό Y δίνει κλίση προς τα πάνω / Θετικό Y δίνει κλίση προς τα κάτω

```
.box1 {  
  width: 200px;  
  height: 200px;  
  background-color: red;  
  transition: transform 2.15s;  
}  
  
.box1:hover {  
  transform: skewX(-20deg);  
}
```



Transform-origin

- Ορίζει το σημείο της περιστροφής που by default είναι το κέντρο

```
.box1 {  
    width: 200px;  
    height: 200px;  
    background-color: red;  
    transform-origin: left top;  
    transition: transform 2.15s;  
}  
  
.box1:hover {  
    transform: rotate(20deg);  
}
```



Συνδυασμός Transform

Προγραμματισμός στο Web

```
.box1 {  
  width: 200px;  
  height: 200px;  
  background-color: red;  
  transition: transform 2.15s;  
}  
  
.box1:hover {  
  transform: rotate(90deg) scale(2) translateY(-100%) translateX(40%);  
}
```

- Περιστροφή 90 μοίρες, διπλάσιο size, move δεξιά 100% του πλάτους και 40% του ύψους





Άσκηση

- Επεκτείνετε το μενού του παραδείγματος ώστε όλα τα menu items του να είναι dropdown εισάγοντας δικά σας submenu items