ΠΕΡΙΕΧΟΜΕΝΑ:

1. **Εξειδίκευση**
2. **Σειρά Εμφάνισης και Στυλ Συγγραφέα**
3. **Υπέρβαση με την !important**
4. **Σύνοψη για την επίλυση Συγκρούσεων**

Δημήτρης Πολυμέρας

Σμαραγδένιος Χορηγός Μαθήματος

Κων/νος Λαμπιδάκης

Ασημένιος Χορηγός Μαθήματος

Επίλυση Συγκρούσεων με την εξειδίκευση (specificity):

- Εφόσον ένας κανόνας ορίζεται σε δύο ή περισσότερους επιλογείς μας (προσοχή: επιλογείς που έχουμε γράψει εμεις, που δεν είναι μαρκαρισμένοι με το !important)
  - **θα επικρατήσει ο κανόνας που έχει τεθεί από τον επιλογέα με τη μεγαλύτερη εξειδίκευση.**
- Η εξειδίκευση ενός επιλογέα ορίζεται ως:
  - Inline στυλ μετράει για 1000
  - Επιλογέας ID μετράει για 100
  - Επιλογέας κλάσης, ψευδο-κλάσης και χαρακτηριστικού μετράει για 10
  - Επιλογέας τύπου και ψευδο-στοιχεία μετράνε για 1
  - Ο καθολικός επιλογέας μετράει για 0

Παράδειγμα 1: Υπολογισμός Εξειδίκευσης

Επιλογέας	Εξειδίκευση
style="…"	1000
#id1 #id2	200
#id.class	110
.class1 > .class2 ~ p	21
*.class input[type]	20
.class p::first-letter	12
p, a, h1, h2	4

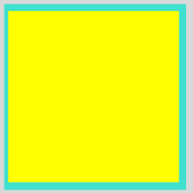
Διεστραμμένη Παρατήρηση:

- Πιο σωστό θα ήταν να μετράμε την εξειδίκευση σαν μια τετράδα αριθμών σε ένα σύστημα αρίθμησης με αυθαίρετα μεγάλη βάση.
  - Γιατί π.χ. αν ένας επιλογέας έχει 10 κλάσεις, χάνει από έναν επιλογέα που έχει 1 id: 0,1,0,0 > 0,0,10,0

Παράδειγμα 2: specificity

```
<body>
 <div id="id1" class="class1">
   <div id="id2" class="class2"></div>
 </div>
</body>
```

```
body > div > #id2 {
  background-color: yellow;
}
#id1 > div {
  background-color: black;
}
body .class2 {
  background-color: red;
}
```



- Εφόσον έχουμε ισοπαλία στους νικητές (ίδια εξειδίκευση), νικάει ο κανόνας ο οποίος έχει γραφεί τελευταίος (πιο πρόσφατος κατά την σειρά ανάγνωσης).

Παράδειγμα 3: order

```
<body>
 <div id="id1" class="class1">
   <div id="id2" class="class2"></div>
 </div>
</body>
```

```
div > #id2 {
  background-color: yellow;
}
#id1 > div {
  background-color: black;
}
```

«Μάζεμα» κανόνων από τον browser:

- Ο browser θα μαζέψει τελικά πολλούς κανόνες από πολλές πηγές και θα τους βάλει στην εξής σειρά (σαν να δημιουργείται ένα κοινό αρχείο που περιέχει τους κανόνες):

user agent stylesheet	Default κανόνες που χρησιμοποιεί ο browser
author stylesheet(s)	Τα στυλ που γράφουμε εμείς (είτε inline, είτε <style> στο <head>, είτε με <link>)
user stylesheet	Κάποιοι browser δίνουν τη δυνατότητα στο χρήστη να έχει κανόνες που θα χρησιμοποιούνται σε κάθε σελίδα (παρωχημένο)

Σημειώσεις:

- Για τα παραπάνω stylesheet(s):
- Μπορούμε να έχουμε πολλά αρχεία με στυλ και να τα έχουμε ενσωματώσει στο html έγγραφό μας, με διαδοχικές χρήσεις του <link>. Η σειρά με την οποία τα έχουμε ενσωματώσει έχει σημασία, με αυτήν θα ενσωματωθούν στο «κοινό αρχείο».

- **Εφόσον ένας κανόνας καθορίζεται από διαφορετικούς επιλογείς με ίδια εξειδίκευση, νικά ο τελευταίος κανόνας** (με βάση τη σειρά της ενσωμάτωσης των αρχείων)

Παράδειγμα 4: multiple_files

- Οι κανόνες από τις πηγές αριστερά, αναφέρονται σαν **στυλ του συγγραφέα** (author styles)
  - (Επειδή τους γράψαμε "εμείς", οι προγραμματιστές - συγγραφείς του κώδικα) - προσοχή ότι και το user-agent stylesheet μπαίνουν σε αυτήν την κατηγορία.
- Ωστόσο κανόνες στυλ μπορούν να προκύψουν και από το χρήστη (**στυλ του χρήστη**)

Παρατήρηση:

- Ο Firefox (που δίνει έμφαση στην εμπειρία του προγραμματιστή), δίνει τη δυνατότητα να ενσωματωθεί ένα αρχείο στυλ, το οποίο θα εφαρμόζεται σε όλες τις σελίδες
- Παλιότερα και άλλοι browser πρόσφεραν αυτή τη δυνατότητα.
- Ο Chrome το έχει καταργήσει εδώ και αρκετά χρόνια

Κανόνας:

- **Τα στυλ χρήστη «νικάνε» τα στυλ συγγραφέα.**
  - Έτσι υπάρχει περίπτωση κάποιος power user να αλλοιώνει την εμφάνιση της σελίδας, γράφοντας και ενσωματώνοντας τα στον browser (π.χ. Firefox)

Σημείωση:

- Τα στυλ χρήστη είναι εξαιρετικά προχωρημένο (για τις περισσότερες σελίδες) χαρακτηριστικό.
- Δεν θα ασχοληθούμε περαιτέρω σε αυτή τη σειρά μαθημάτων (αλλά θα ξέρουμε ότι υπερβαίνουν τα στυλ συγγραφέα)

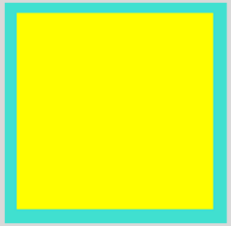
3. Υπέρβαση με την !important

- Μπορούμε να **<u>υπερβούμε την «νίκη» κάποιου κανόνα</u>** με βάση την εξειδίκευση:
  - **<u>Μαρκάροντας ως !important</u>** έναν κανόνα.
  - Το μαρκάρισμα γίνεται αμέσως μετά τον κανόνα και πριν από το ερωτηματικό:

```
selector {
  property: value !important;
}
```

<u>Παράδειγμα 5: important</u>

```
div {
  background-color: yellow !important;
}
#id {
  width: 100px;
  height: 100px;
  background-color: black;
}
.class {
  background-color: red;
}
```



- Αν η ίδια ιδιότητα έχει μαρκαριστεί ως important σε πολλούς επιλογείς, τότε πάλι ισχύουν οι κανόνες εξειδίκευσης.
  - π.χ. αν οριστεί ως important κάποιο inline style, τότε αυτό θα επικρατήσει των important που έχουν συμβατικοί κανόνες

<u>Παράδειγμα 6: important2</u>

```
div {
  background-color: yellow !important;
}
#id {
  width: 100px;
  height: 100px;
  background-color: black !important;
}
```

```
<body>
  <div id="id" class="class"
       style="background-color: blue
       !important;"></div>
</body>
```

<u>Παρατηρήσεις:</u>

- Το !important αλλοιώνει τη λογική του φύλλου στυλ και αν και φαίνεται εύκολη λύση, ίσως να είναι καταστροφική για ένα μεγάλο project
  - Γιατί μία υπέρβαση, ίσως να χρειαστεί και επόμενη, με αποτέλεσμα να αλλοιωθεί όλος ο κορμός του σκεπτικού των φύλλων στυλ
- **<u>Γι' αυτό ποτέ δεν θα χρησιμοποιούμε το !important στα φύλλα μας.</u>**
  - Ωστόσο, υπάρχουν αρκετά προχωρημένες χρήσεις του, που είναι απαραίτητες (π.χ. αν χρησιμοποιήσουμε εξωτερικό φύλλο στυλ από αλλον προγραμματιστή (π.χ. bootstrap) και θέλουμε να υπερβούμε κάποιο στυλ).

Συνοψίζοντας η επίλυση συγκρούσεων (πως παίρνει την τελική του τιμή ένα property) γίνεται ως εξής:

!important	Style που έχει μαρκαριστεί ως important (author < inline < user)
user	User Style (στυλ χρήστη στον browser)
inline	Inline Style στον κώδικα HTML
specificity	Έχουν οριστεί τιμές στο property σε διαφορετικούς επιλογείς. Ο νικητής επιλέγεται με βάση τους κανόνες εξειδίκευσης (της πρώτης διαφάνειας)
last rule	Έχει οριστεί τιμή στο property σε διαφορετικούς επιλογείς και υπάρχει ισοπαλία στους νικητές. Νικάει αυτός που είναι τελευταίος στη σειρά εμφάνισης.
inherited	Δεν έχει οριστεί τιμή στο property σε κανέναν επιλογέα. Το property κληρονομείται. Παίρνει την τιμή του γονέα του.
default	Δεν έχει οριστεί τιμή στο property σε κανέναν επιλογέα. Το property δεν κληρονομείται. Παίρνει την default τιμή του

- Κάθε κόμβος νικάει τον από κάτω του (π.χ. ο τελευταίος κανόνας νικάει την κληρονομημένη τιμή κ.ο.κ.

Παρατηρήσεις:

- Τα important! και inline πρέπει να αποφεύγονται δια ροπάλου
  - οδηγούν σε προσδιορισμό τιμών που είναι δύσκολο να τους υπερβούμε ξανά και το φύλλο στυλ γίνεται εύκολα μη διαχειρίσιμο.
- Τα user stylesheets δεν υποστηρίζονται από όλους τους browsers, οπότε δεν χρειάζεται να τα λαμβάνουμε υπόψη στο σχεδιασμό
- (οπότε πρακτικά μας ενδιαφέρει να καταλαβαίνουμε την ιεραρχία των 4 τελευταίων: **specificity > last rule > inherited > default**

Συνέχεια μαθημάτων:

- Γνωρίζοντας τον βασικό μηχανισμό με τον οποίο δουλεύει η CSS μπορούμε να εστιάσουμε στα βασικά στοιχεία που επηρρεάζουμε σε μία σελίδα στα επόμενα μαθήματα:
  - Κείμενο
  - Μελέτη του «Κουτιού»
  - Διευθέτηση στοιχείων στη σελίδα
  - Χρώμα και Εικόνες
  - Animation
  - Responsive Design
  - Προχωρημένα Θέματα