ל.ע Css3

מה זה Css?

**Css**, או **C**ascading **S**tyle **S**heet, היא שפה שמתארת עיצוב ויזואלי של עמוד Html. הסטנדרט נוצר בשנת 1995, כאשר עד אז היו כותבים את העיצוב כחלק מכתיבת ה-Html.

ישנן **שלוש דרכים** לכתוב Css:

* מאפיין  
  Attribute שניתן לצרף לאלמנט html מסוים וכך לכתוב את העיצוב ב-Css ישירות בתוכו.

<article style="styles will be here">

* תגית  
  קיימת תגית **style** ב-Html, שהיא אלמנט שמכיל שפת Css, כך העיצוב יהיה חלק מדף ה-Html אבל יהיה כתוב ב-Css טהור.

<style>

 selector {

   /\* Styles will be here \*/

}

</style>

* קובץ נפרד  
  ניצור קובץ ייעודי ל-Css שנכתוב.

בקורס נכתוב Css אך ורק בתוך **קובץ נפרד**, מכמה סיבות:

* **איחוד** אלמנטים עם עיצוב דומה בקוד אחד – כך לדוגמה אם יש לי מספר תגיות h1 שארצה שיהיו מעוצבות באותו האופן, אוכל לאחד את כולן תחת אותו קטע קוד (נלמד כיצד בהמשך הל.ע). בצורה זו חסכנו קוד זהה שהיה נכתב אם היינו משתמשים ב-attribute של style.
* **טעינה מהירה** **יותר** של עמוד האינטרנט שלכם – בעקבות הקוד שנחסוך בשימוש בסלקטורים מאוחדים עבור כמה אלמנטים, העמוד שלנו ייטען מהר יותר.
* **הפרדה לוגית** בין מבנה ותוכן הדף לבין הנראות שלו – שפת ה-html מתארת לנו איזה אלמנטים אמורים להיות בדף ומה היחסים ביניהם. שפת css מתארת את המראה הוויזואלי של הדף ושל האלמנטים בו. כדי להבליט את הפרדת התפקידים נרצה ליצור שני קבצים שונים עבור הדף, אחד למען ה-html ואחד ל-css.

יצירת קובץ Css

הסביבה איתה נעבוד במקצוע web היא Visual Studio Code. על מנת ליצור קובץ Css ניצור פשוט קובץ עם סיומת .css כמו בדוגמה:



הקובץ ייראה כאוסף של קטעי קוד מאוגדים תחת שורה שנקראת **סלקטור** (עליו נדבר עוד בהמשך), או בערך ככה:

selector {

    /\* Styles will be here... \*/

}

selector {

    /\* Styles will be here... \*/

}

selector {

    /\* Styles will be here... \*/

}

לאחר שיצרנו את קובץ ה-css, נותר רק לקשר אותו לקובץ ה-html שלנו, על מנת שה-html "ידע" שקיים לו קובץ עיצוב שהוא יכול להשתמש בו. נעשה זאת באמצעות **תגית link** שתימצא בתוך ה-**head** של קובץ ה-html, שתיראה ככה:

<link rel="stylesheet" href="/..styles.css">

סינטקס

כמה מונחים בסיסיים שחשוב להכיר:

Property

תכונה, שורת עיצוב שתחול על אובייקט שנבחר. כאשר נגדיר את העיצוב שאנחנו רוצים להחיל על אובייקטים מסוימים בקוד שלנו נעשה זאת באמצעות properties. Properties הם המרכיב המרכזי של קוד ה-css שלנו אך הלמידה והכתיבה שלהם מאוד טכנית ומסתכמת בחיפוש בגוגל וצבירת ניסיון, לכן לא נתמקד בהם בל.ע הזה.

ה-property ייכתב תמיד באותיות קטנות.

selector {

    flex-grow: inherit;

    padding: 0vw 30vw 20vw 10vw;

}

סלקטור

הדרך בה נייצג/נבחר אלמנטים של html בתוך ה-css שלנו, על פי כללים ותכונות של האלמנט שנרצה לבחור. אין הגבלה על כמות ה-properties שאפשר לכתוב עבור כל סלקטור.

הסלקטור ייכתב תמיד באותיות קטנות.

selector {

    flex-grow: inherit;

    padding: 0vw 30vw 20vw 10vw;

}

אילו סוגי סלקטורים קיימים לנו?

תגית

כאשר נשתמש בסלקטור מסוג תגית, הוא יחול על כל האלמנטים בקוד ה-html שלנו שהם מסוג התגית הספציפית שהגדרנו.

בדוגמה למטה, העיצוב יחול על כל האלמנטים שיהיו מסוג **li**:

li {

    display: inline;

}

class

class היא attribute שאפשר להגדיר לאלמנט html. אפשר להגדיר את אותה class לכמות לא מוגבלת של אלמנטים. כאשר נשתמש בסלקטור css מסוג class, הוא יחול על כל האלמנטים בקוד ה-html שלנו שלהם הגדרנו את ה-class הספציפי הזה. ה-css יודע שמדובר ב-class כאשר יש לפני הסלקטור **נקודה**.

בדוגמה למטה, העיצוב יחול על כל האלמנטים שלהם מוגדר ה-class **profile-picture**:

.profile-picture {

    width: 50%;

    border-color: #363732;

}

id

id הוא attribute שאפשר להגדיר לאלמנט html. אפשר להגדיר id מסוים רק לאלמנט אחד בקוד – הוא צריך להיות חד-חד ערכי. כאשר נשתמש בסלקטור css מסוג id, הוא יחול על האלמנט בקוד ה-html שלנו שלו הגדרנו את ה-id הספציפי הזה. ה-css יודע שמדובר ב-id כאשר יש לפני הסלקטור **האשטאג**.

בדוגמה למטה, העיצוב יחול על כל האלמנטים שלהם מוגדר ה-id **user-name**

#user-name {

    font-weight: bold;

}

Combination

ניתן להשתמש ב**כמה סלקטורים** יחד תחת אותו בלוק עיצוב. ניתן לעשות זאת עם כל סוג סלקטורים שקיים, וקיימים סלקטורים מיוחדים שמטרתם לחבר ביניהם על מנת להעניק להם משמעות חדשה – כמו יחסי קינון, אלמנטים שקודמים/אחרי אלמנטים מסוימים וכו'.

בדוגמה למטה, העיצוב יחול על אלמנטים מסוג p שמקוננים באלמנטים מסוג div

div p {

    font-family: Arial, Helvetica, sans-serif;

    color: #363732;

}

Pseudo-class

זהו סלקטור מיוחד, שנועד לתאר מצב ספציפי של אלמנט – מיקום בקינון, אירוע שהמשתמש מפעיל עליו וכדומה. נשתמש בו תמיד **בצירוף אלמנט** מסוים – ניתן להשתמש בו בפני עצמו אבל לרוב לא נרצה לעשות את זה.

בדוגמה למטה, העיצוב יחול על אלמנטים מסוג button, **רק כאשר** העכבר נמצא עליהם.

button:hover {

    color: #A5FFD6;

    font-weight: bolder;

}

קיימים הרבה סלקטורים בשפה והעובדה שאתם לא מכירים את כולם יכולה להיות מלחיצה; ההיכרות והעבודה איתם תבוא עם הניסיון שלכם בתכנות ב-Css, אז אל תפחדו ממה שאתם לא יודעים.

הפצת גרסאות של Css

הוצאת הגרסאות של Css מתנהלת ככה שהגרסה החדשה תמיד עוטפת את הגרסה הקודמת ומרחיבה אותה מבחינת סינטקס. הגרסאות של Css התחילו כמודול שלם שיוצא בבת אחת, אך גרסה 3 הייתה הראשונה שיצאה ב**מודולים נפרדים** – כל מהות מסוימת של Css יצאה כמודול נפרד.

התחילו לפתח את גרסת Css3 בשנת 1999, מאז היא התחילה לצאת, כפי שנאמר, לא בבת אחת אלא כל מודול כשהסתיימה העבודה עליו. העבודה עם Css3 מומלצת כסטנדרט על ידי ה-W3C (ארגון שמטרתו ליצור קהילת מפתחי Web שעובדים תחת סטנדרט אחיד) מאז 2011 – ומאז בכל פעם שיצא מודול חדש גם הוא נכנס תחת אותו סטנדרט בזמן אחר.

מה הגרסה מוסיפה על הגרסאות הקודמות?

* סלקטורים חדשים
* Properties חדשים
* אפשרות להגדיר צבעים בפורמטים נוספים ביחס לגרסאות הקודמות

ומה עם גרסת Css4?

מי מכם שמכיר את שפת Css מלפני כן אולי שמע כל מיני דברים על סלקטורים בגרסה 4, לכן צריך להעמיד דברים על דיוקם – אין **גרסת Css שלמה** שנקראת Css4. למה? כיוון ש-Css3 חולקה למודולים נפרדים, קיימים מודולים ספציפיים שמרחיבים מודולים ברמה 3 ולכן נחשבים ב**רמה 4**, אבל זו לא גרסה שלמה כיוון שלא לכל המודולים קיימת הרחבה כזו.