CSS

By Alon Abargel

חזרה על חומר משיעור קודם

בשיעורים שעברו נגענו בנושאים הבאים:

FORMS

IFRAME

האם כל הנושאים ברורים?

Css styling

Colors

Padding

Margin

Inline

Inline-block

Float

background

Background-color

Background-repeat

Background-position

Background-attachment

Background-image

border

מהם המאפיינים של התכונה?

איזה סוגי מסגרות יש?

font

איך ממשים פונט ?

אם אנחנו משתמשים בגופן ששמו מכיל יותר ממילה אחת, נצטרך לשים את השם שלו במרכאות כי אחרת הדפדפן עשוי להתבלבל ולחשוב שזה שני גופנים או יותר, כל מילה כגופן עצמאי, לכן נרצה ליידע את הדפדפן שכל המילים שייכות לשם של גופן אחד, אז אנחנו הולכים להקיף אותם במרכאות, ולאחר מכן הנקודה-פסיק שאנחנו לא שוכחים.
יש מצבים שבהם מגדירים גופן אך בפועל לא רואים את הגופן בדף, או שבמחשב מסוים הגופן פועל ובמחשב אחר לא, וזה מכיוון שאם הגופן לא מותקן במערכת הפעלה הוא לא יוצג, ובמקומו יוצג גופן ברירת המחדל שהדפדפן מגדיר. זה משהו שחייבים להיות מודעים אליו, שיש גופנים שלא מותקנים בכל המחשבים ולכן יש משתמשים מסוימים שלא יוצג להם הגופן שאליו התכוונתם. יש דרך לעקוף את זה ולהציג גופן שלא מותקן במחשב של המשתמש ונלמד זאת בהמשך.
דוגמה לגופן עם שם שחייב להיות במרכאות:

```
body {
    font-family: 'Times New Roman';
}
```

הערמת גופנים

כאשר אנחנו מגדירים גופן בCSS שלנו, יש לנו אופציה להערים גופנים, הערמת גופנים היא בעצם רשימה מופרדת בפסיקים של גופנים, אנחנו מספקים מספר רב של אפשרויות שאם הדפדפן לא מצליח לאתר את הגופן הראשון כי הוא לא מותקן במחשב הוא משתמש בשני וכן הלאה. בעצם אנחנו אומרים במידה והראשון לא עובד האפשרות השנייה היא הטובה ביותר עבורי, ואם השני לא מותקן אז האפשרות השלישית היא הטובה ביותר עבורי וכן הלאה.

יש מושג שנקרא Web safe fonts (גופני אינטרנט בטוחים), זוהי רשימה של גופנים שמותקנים בכל המחשבים ונחשבים לבטוחים לשימוש.

דוגמה להערמת גופנים:

```
body {
    font-family: Arial, Helvetica, sans-serif;
}
```

שימוש בגופנים חיצוניים

יש אפשרות לייבא גופנים מבחוץ, לא על ידי התקנה שלהם במחשב אלא על ידי קישור אליהם למיקום שלהם ברשת. מה שצריך לעשות זה לשים את הקישור לפונט בhead של מסמך הhtml ואז אפשר להשתמש בפונט בדפים שלנו. ישנם מקורות שונים שאפשר להשתמש בגופנים שלהם כמו google fonts, adobe font ועוד.

```css
div {
    width:600px;
}

div {
    min-width: 400px;
    max-width: 800px;
}
```

המאפיין height

ניתן לקבוע גובה של אלמנט על ידי שימוש בheight. כאשר אנחנו קובעים הגדרה זו האלמנט תמיד יהיה בגובה שהוגדר. ניתן להגדיר בפיקסלים (px) וגם באחוזים (%). כאשר נשתמש בפיקסלים הגודל יהיה קבוע ולא ישתנה אף פעם, כאשר נשתמש באחוזים הגודל יהווה אחוז מגובה האלמנט שעוטף את האלמנט, אם האלמנט אב יהיה גדול יותר האלמנט יגדל בהתאמה, אם האלמנט אב יהיה קטן יותר האלמנט יקטן בהתאמה.

המאפיין min-height

מקבל את אותם אפשרות (פיקסלים ואחוזים) כמו height.
מאפיין זה מגדיר מה המינימום גובה שהאלמנט יהיה, הגובה בפועל (במידה והוא לא קטן מתנאי המינימום) והמקסימום נקבע על ידי הגדרות ברירת המחדל.

המאפיין max-height

מקבל את אותם אפשרות (פיקסלים ואחוזים) כמו height.
מאפיין זה מגדיר מה המקסימום גובה שהאלמנט יהיה, הגובה בפועל (במידה והוא לא גדול מתנאי המינימום) והמינימום נקבע על ידי הגדרות ברירת המחדל.

CSS COLORS

צבעים

איזה דרכים יש לנו להציג צבע?

מה החיסרון בלרשום את שם הצבע לעומת מספר מדוייק?

```
element {
    color: #FF0000;
}
```

צבע על ידי שימוש בשם שלו (alias)

אפשר גם לכתוב את שם הצבע, אך שיטה זו אינה מומלצת מכיוון שלכל דפדפן מוגדר קוד אחר עבור השם של הצבע.

```
element {
    color: red;
}
```

> *<u>שיטת עבודה מומלצת</u>*: הגדרה של צבעים על ידי rgb מכיוון שאפשר בעתיד להחליף את המאפיין rgb ב-rgba ואז אפשר לתת לצבע שקיפות בין 0 ל-1 (opacity).

המאפיין font-size

מאפיין הקובע את גודל גופן, יש כמה סוגים שונים של גדלים שאפשר לעבוד איתם מומלץ לעבוד עם פיקסלים (px).

```
element {
    font-size: 14px;
}
```

המאפיין text-align

מאפיין זה קבוע את כיוון הטקסט בתוך האלמנט. יש לנו 4 סוגים של כיוון/
ימין (right)

```
element {
    text-align: right;
}
```

שמאל (left)

```
element {
    text-align: left;
}
```

מרכז (center)

```
element {
    text-align: center;
}
```

יישור שורות לאותו הגודל (justify) כמו בעיתונים

```
element {
    text-align: justify;
}
```

הוספת הערות בתוך קובץ הCSS

אפשר להשאיר בקובץ שלנו הערות, לכל מטרה על ידי שימוש בתחביר קבוע. הדפדפן מתעלם מן ההערות.
הערה של שורה אחת:

```
/* This is a single-line comment */
```

הערה של יותר משורה אחת:

```
/* restnet todo list:
    - color the headers
    - make text bolder
    - remove any blue color from the text
*/
```

קיצורי כתיבה

```
div {
    padding-top: 10px;
    padding-right: 10px;
    padding-bottom: 10px;
    padding-left: 10px;
}
```

יש דרך מקוצרת לכתוב את המאפיינים בשורה אחת, כאשר לא מציינים את הכיוון וכותבים רק padding או margin ואז את כל הגדלים ברצף מופרדים ברווח, הראשון מייצג את הגודל מלמעלה, השני מייצג מצד ימין, השלישי מייצג מלמטה והרביעי מצד שמאל (מתחילים מלמעלה וממשיכים עם כיוון השעון).
דוגמה לpadding ברישום מקוצר:

```
div {
    padding: 10px 10px 10px 10px;
}
```

Inline & block

```
div {
    display: inline;
}
```

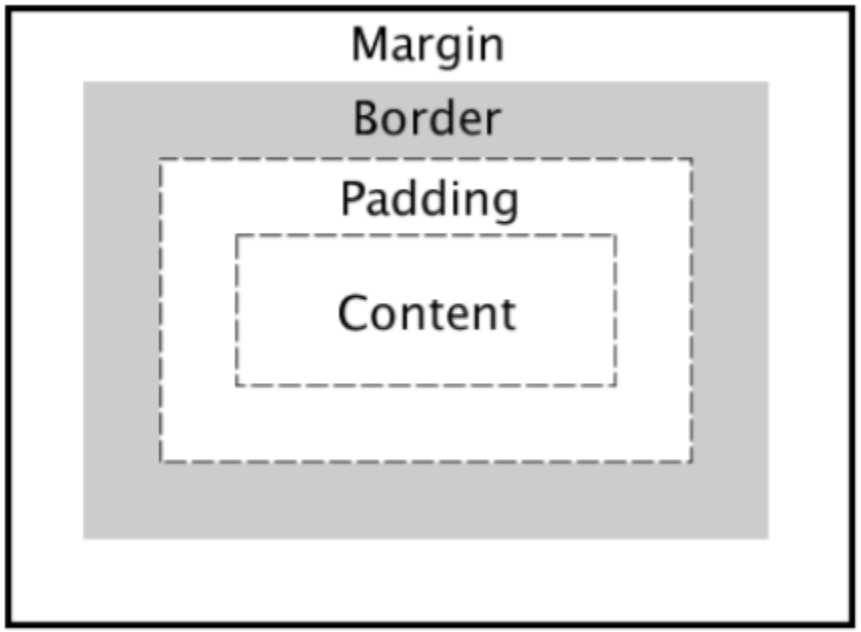
אפשר גם להחביא אלמנט שיהיה בלתי נראה כאילו מעולם הוא לא היה.

```
div {
    display: none;
}
```

יש גם את האפשרות להגדיר אלמנט כblock וגם inline יחדיו, הגדרה זו אומרת שהאלמנט הוא מסוג inline אבל יכול לקבל ערכים של גובה ורוחב כמו אלמנט מסוג block.

```
span {
    display: inline-block;
}
```

מודל הקופסא



Css position

Fixed

Abolute

Static

Relative

נסביר את כולם.

מיקום סטטי – static

מיקום סטטי זוהי ברירת מחדל של האלמנטים בHTML, אם לא ניתן מאפיין position לאלמנט הוא יהיה סטטי. המאפיינים top, bottom, right, left לא משפיעים כאשר אלמנט מוגדר להיות סטטי.

מיקום קבוע – fixed

החלת מיקום קבוע לאלמנט, על ידי קביעת המאפיין position לערך fixed, מוציא את האלמנט מהזרימה של המסמך ושאר האלמנטים בדף לא מתחשבים בו, הם מתנהגים כאילו הוא לא קיים מבחינתם.

אחרי שאנחנו קובעים לאלמנט שלנו position עם ערך fixed נוכל להזיז את האלמנט למיקום קבוע שאנחנו חפצים בו. על ידי שימוש במאפיינים top, bottom, right, left.

לא ניתן לקבוע שני מאפיינים של הזזה על אותו הציר, זאת אומרת שנבחר בין top ל-bottom (ציר אנכי) ונחבר בין right לבין left (ציר אופקי).

בסוג המיקום fixed האלמנט קובע את מיקומו ביחס לדפדפן, זאת אומרת שאם נבצע גלילה בדף נמשיך לראות את האלמנט מוצג לנו באותו המיקום, בדרך כלל נראה את זה בתפריט ניווט של אתר שמלווה אותנו לאורך כל הגלילה.

לסיכום על fixed:

אלמנטים אחרים מתעלמים מממנו ולכן לא מתחשבים בגובה שלו וברוחב שלו, בשוליים או בכל דבר אחר של האלמנט.

מיקום fixed מזיז את עצמו ביחס לדפדפן ולכן תמיד נראה אותו באותו המקום גם כאשר נבצע גלילה.

דוגמה:

```
div {
    position: fixed;
    top: 100px;
    left: 50px;
}
```

מיקום מוחלט – absolute

החלת מיקום מוחלט לאלמנט, על ידי קביעת המאפיין position לערך absolute, מוציא את האלמנט מהזרימה של המסמך ושאר האלמנטים בדף לא מתחשבים בו, הם מתנהגים כאילו הוא לא קיים מבחינתם.
אחרי שאנחנו קובעים לאלמנט שלנו position עם ערך absolute נוכל להזיז את האלמנט למיקום קבוע בדף שאנחנו חפצים בו על ידי שימוש במאפיינים top, bottom, right, left.
לא ניתן לקבוע שני מאפיינים של הזזה על אותו הציר, זאת אומרת שנבחר בין top ל-bottom (ציר אנכי) ונחבר בין right לבין left (ציר אופקי).

עד כאן נשים לב שהמיקום הזה ממש דומה למיקום fixed, אבל יש ביניהם הבדל משמעותי.
בסוג המיקום absolute האלמנט קובע את מיקומו ביחס למסמך שלנו, זאת אומרת שאם נבצע גלילה הוא יזוז עם המסמך.
יש התנהגות מיוחדת לסוג מיקום זה כאשר הוא נמצא בתוך אלמנט אחר, אם לאלמנט שמכיל אותו (לאבא שלו) לא מוגדר מיקום (על ידי position) שהוא שונה מברירת המחדל, האלמנט שלנו ימקם את עצמו ביחסית למסמך (יחסית לתחילת הbody), אם לאבא שלו מוגדר מיקום, האלמנט שלנו ימקם את עצמו יחסית לאבא שלו.
דוגמה:

```
div {
    position: absolute;
    top: 100px;
    left: 50px;
}
```

מיקום יחסי – relative

החלת מיקום יחסי לאלמנט, על ידי קביעת המאפיין position לערך relative, לא מוציא את האלמנט מהזרימה של המסמך ושאר האלמנטים בדף כן מתחשבים בו, מתחשבים במיקום המקורי של האלמנט לפני שביצענו הזזה.

אחרי שאנחנו קובעים לאלמנט שלנו position עם ערך relative נוכל להזיז את האלמנט ביחס למיקום המקורי שלו על ידי שימוש במאפיינים top, bottom, right, left.

לא ניתן לקבוע שני מאפיינים של הזזה על אותו הציר, זאת אומרת שנבחר בין top ל-bottom (ציר אנכי) ונחבר בין right לבין left (ציר אופקי).

```
div {
    position: relative;
    top: 50px;
    right: 50px;
}
```

טבלת מיקומים

position	האם יוצא מזרימת המסמך	מתייחס ל
fixed	כן	חלון הדפדפן
absolute	כן	לאבא הראשון שיש לו position אם אין לאחד האבות בהיררכיה position אז מתייחס למסמך (body)
relative	לא	למיקום המקורי של האלמנט

ירושה של מיקום

ירושה של מיקום, על ידי קביעת המאפיין position לערך inherit. מה שאומר שהערך של המאפיין position יקבע על ידי אלמנט האבא, מה שמוגדר לאבא במאפיין position יהיה גם באלמנט הבן אם יש לו inherit במאפיין position.

המאפיין z-index

המאפיין z-index קובע מי יופיע על מי כאשר יש חפיפה בשטח של אלמנטים. בברירת המחדל הערך שווה ל-1 אלא אם כן נציין אחרת.
מאפיין זה מקבל תוקף רק כאשר לאלמנט מוגדר position שאינו ברירת מחדל.

```
div {
    position: relative;
    top: 50px;
    right: 50px;
    z-index: 2;
}
```

אלמנטים צפים

אלמנט צף זה אחד מאבני היסוד בבניית פריסה של דף אינטרנט, אלמנט צף משמעותו היא להזיז אלמנט לימין או לשמאל בדף שלנו ולבטל את שבירת השורה שלאחריו. נגדיר זאת על ידי המאפיין float.

על ידי קביעת המאפיין float לאלמנט עם אחת משני האופציות שלנו left או right אנחנו מסירים את האלמנט מזרימת המסמך ואז דוחפים אותו לקצה השמאלי או הימני (לפי מה שהגדרנו). ההבדל בין קביעת position הוא שבמקרה שלנו כאן שאר האלמנטים מתייחסים לגובה והרוחב של האלמנט.

הצפה שמאלה של אלמנט:

```
div {
    float: left;
    width: 50px;
    height: 50px;
}
```

Css selectors

מהם סלקטורים?

איזה סלקטורים אנחנו עוד מכירים?

איזה סלקטורים קיימים לדעתכם?

יש סוג מיוחד של selectors, שנקרא pseudo selectors, סוג זה מהווה בעצם מצבים שונים של אלמנט בדף, כמו למשל אלמנט שכרגע העכבר נמצא מעליו, אלמנט שכרגע לוחצים עליו ועוד, השם pseudo-selectors אומר שאנחנו כותבים selector עבור מצב מסוים של אלמנט.
תגית a, כאשר משתמשים בה יש לדפדפן צבע שהוא קובע עבור תגית זו, ואם לחצנו על הקישור ונחזור לדף שממנו לחצנו על הקישור מאוחר יותר, נראה שהדפדפן צבע לנו את הקישור בצבע אחר כי כבר ביקרנו בקישור הזה בעבר, אלו הם מצבים שונים של תגית a.

מצבים שונים של תגית a:

מצב hover

מצב זה הוא עבור המקרה שבו אנחנו מרחפים עם סמן העכבר מעל התגית.

מצב link

מצב זה עבור קישור שהגולש עדיין לא ביקר בו.

מצב active

מצב זה עבור קישור שכרגע לוחצים עליו, מה יקרה בזמן הלחיצה עצמה.

מצב visited

מצב זה עבור קישור שהגולש ביקר בו בעבר.

דוגמה לכתיבת CSS עבור המצבים השונים:

```
/* unvisited link */
a:link {
    color: red;
}

/* visited link */
a:visited {
    color: green;
}

/* mouse over link */
a:hover {
    color: hotpink;
}

/* selected link */
a:active {
    color: blue;
}
```

סלקטור לפי מזהה ייחודי

ניתן לבחור אלמנט לפי תעודת הזהות שלו, הid שלו. ניתן לתת רק לאלמנט אחד בדף id.
דוגמה לאלמנט עם id בHTML:

```
<p id="leader">
    p with id
</p>
```

על מנת לתת לאלמנט עיצוב לפי הid שלו נשתמש בסולמית בקובץ הCSS ולאחר מכן את הערך של הid.

```
#leader {
    background-color: black;
    color: white;
}
```

סלקטור ספציפי

ניתן גם לשלב בין כל מה שראינו ולכתוב סלקטור שהוא ספציפי יותר.
לדוגמה, נניח הקוד HTML הבא:

```
<div id="leader">
    <p id="ad" class="big content advertisment">
        example p with 3 classes
    </p>
</div>
```

אם אנחנו רוצים להיות יותר ספציפיים לגבי האלמנט נוכל לכתוב את סוג האלמנט, את classה שלו ואת idה שלו ביחד.

```
p.big.content.advertisment {
    background-color: black;
    color: white;
}
```

כל האלמנטים - *

סמל הכוכב יחול על כל האלמנטים בדף. רבים משתמשים בטריק הזה כדי לאפס את ה־margin וה־padding של כל האלמנטים.

עדיף לא להשתמש בזה מכיוון שזה מעמיס על הדפדפן כאשר הדף שלנו יכיל הרבה אלמנטים.

```
* {
    margin: 0;
    padding: 0;
}
```

הסלקטור * יכול גם להיות על כל הבנים תחת אלמנט

```
#container * {
    border: 1px solid black;
}
```

שימוש בספריות חיצוניות

נכיר ספריות מוכנות שישדרגו לנו את עיצוב האתר

Font-awesome

Animate.css