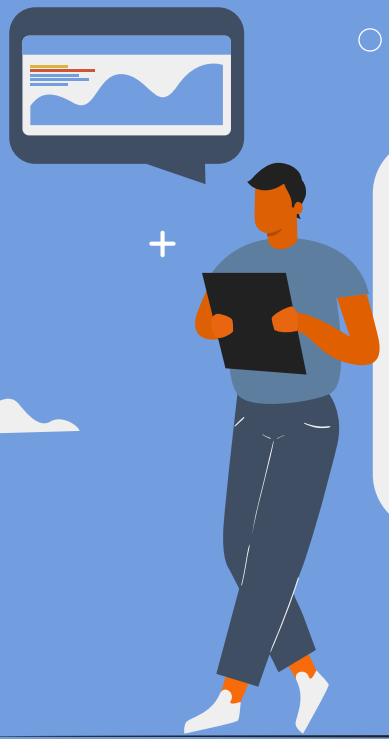


CSS Grid

פריסה דו-ממדית שבאה לעשות סדר

מה זה בעצם Grid?

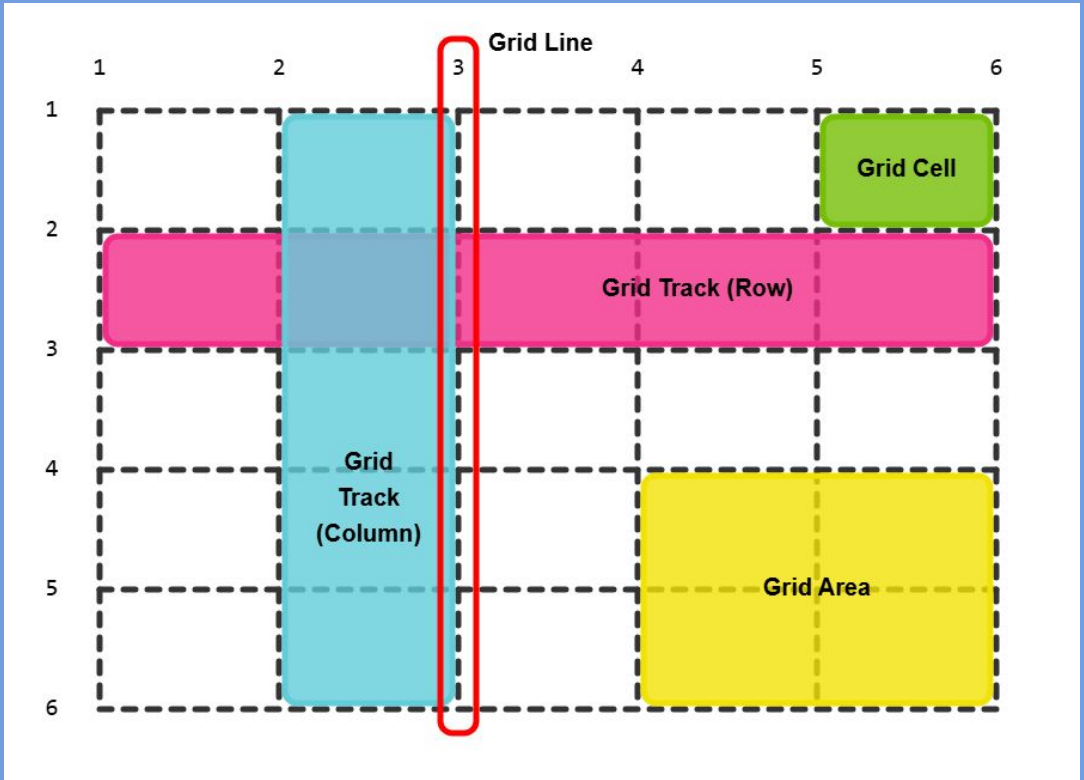
- CSS Grid הוא מודל פריסה (Layout Model) ב-CSS המיועד לפריסה דו-ממדית - כלומר, סידור אלמנטים בשורות וגם בטורים בו-זמנית.

- **הבדל מרכזי מFlexBox:**
  - Flexbox: פריסה חד-ממדית (שורה או טור). מצוין לרכיבים קטנים יותר בתוך עמוד עם אלמנטים דומים.
  - Grid: פריסה דו-ממדית (שורה וגם טור). מצוין למבנה העמוד הכללי ולפריסות מורכבות.

מה זה בעצם Grid?

- **מוסגי יסוד:**
  - **Grid Container**: האלמנט ההורה שעליו מוגדר `display: grid`.
  - **Grid Items**: הילדים הישירים של ה-Container.
  - **Grid Lines**: הקווים האופקיים והאנכיים שיוצרים את מבנה הגריד.
  - **Grid Tracks**:שורות או הטורים שנמצאים בGrid.
  - **Grid Cells**: יחידות השטח הבודדות בגריד.
  - **Grid Areas**: אזורים מוגדרים בגריד שיכולים לכלול מספר תאים.

מה זה בעצם Grid?



למה דווקא Grid?

- פתרון אלגנטי למבני עמוד מורכבים.
- יכולת יישור ומיקום מדויק בשני הצירים בו-זמנית (בלי טריקים של Flexbox).
- בניית תבניות רספונסיביות בקלות יחסית.
- הפרדה ברורה בין מבנה ה-HTML (סדר התוכן) לבין עיצוב ה-CSS (מיקום התוכן בגריד).

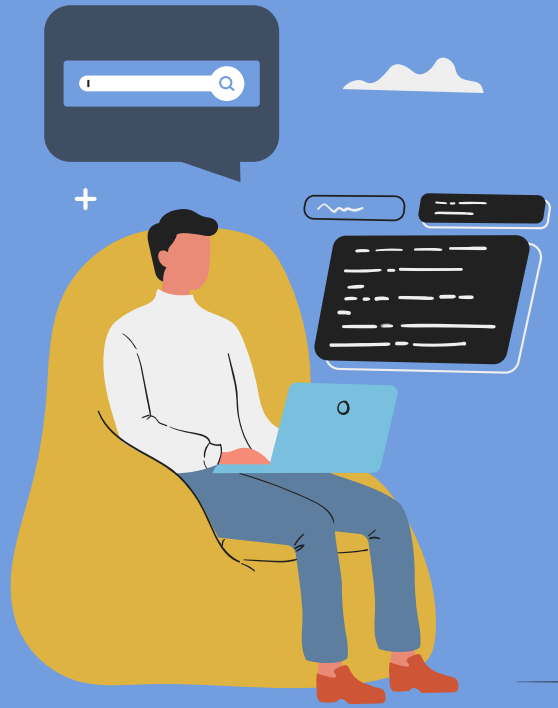
Grid Container – יוצרים את הרשת

- הכל מתחיל עם `display: grid` על האלמנט העוטף. הילדים הופכים אוטומטית ל-Grid Items ומסתדרים לפי הגריד.
- `grid-template-columns`: מגדיר את גובה השורות בגריד. אפשר להשתמש בפיקסלים, אחוזים, (`fr` fraction – חלק יחסי מהמקום הפנוי), auto ועוד.
- `grid-template-rows`: מגדיר את רוחב העמודות בגריד. תחביר דומה ל-`grid-template-columns`.
- יחידת `fr`: חשובה מאוד לגמישות. `1fr` מקצה חלק שווה מהמקום הפנוי.
  - **פונקציות שימושיות:**
  - `repeat(count, size)`: מאפשר לחזור על הגדרה של שורה או עמודה מספר פעמים. `למשל:` `repeat(3, 1fr)` יוצר 3 עמודות ברוחב שווה.
  - `minmax(min, max)`: מגדיר טווח גדלים לשורה או עמודה.

ריווח בין השורות/העמודות

- בדומה ל-`gap` ב-FlexBox, גם ב-Grid יש לנו דרך קלה להוסיף רווחים בין השורות והעמודות.
- `row-gap`: מגדיר את הרווח בין השורות.
- `column-gap`: מגדיר את הרווח בין העמודות.
- `gap`: קיצור לשניהם (`gap: row-gap column-gap`) אם מציינים ערך אחד, הוא ישמש לשניהם.

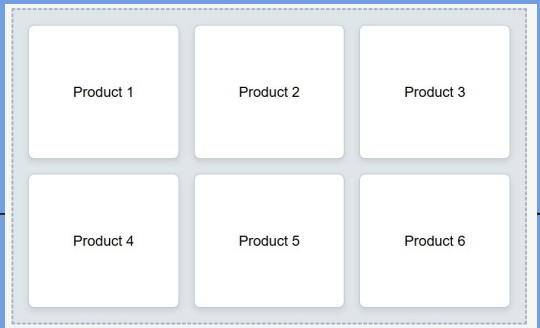
תרגול 1: מבנה בסיסי



המשימה: צרו פריסה של 3 עמודות שחולקות את המקום הפנוי באופן שווה. צרו 2 שורות, כאשר כל אחת מהן בגובה קבוע של 180px.
הוסיפו רווח אחיד של 20px בין כל הברטיסיות.

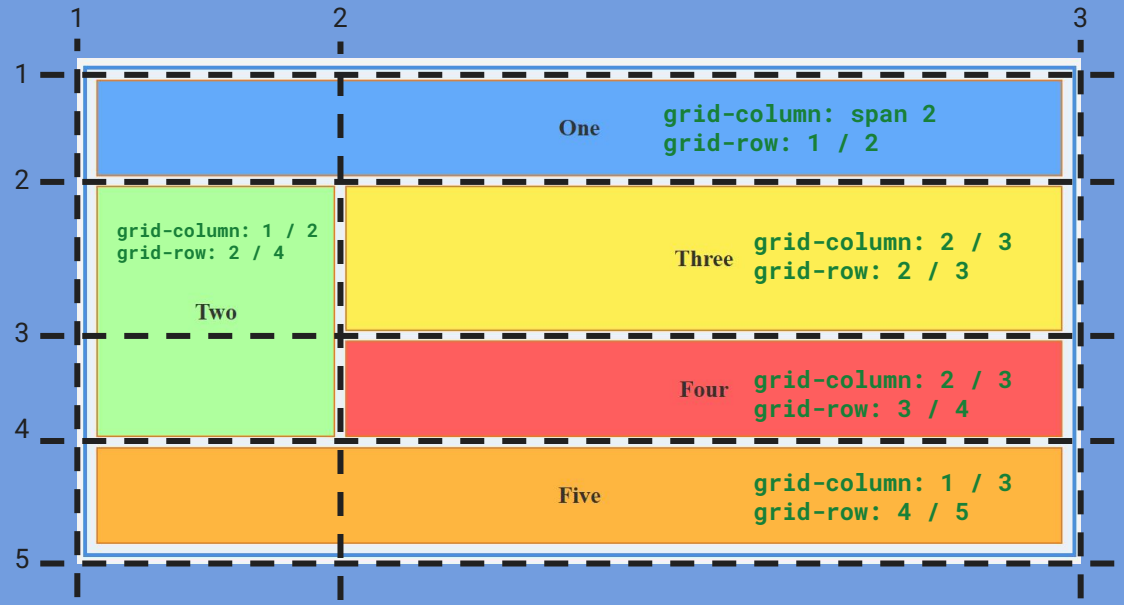
קישור : לחצו כאן.

תוצר נדרש:

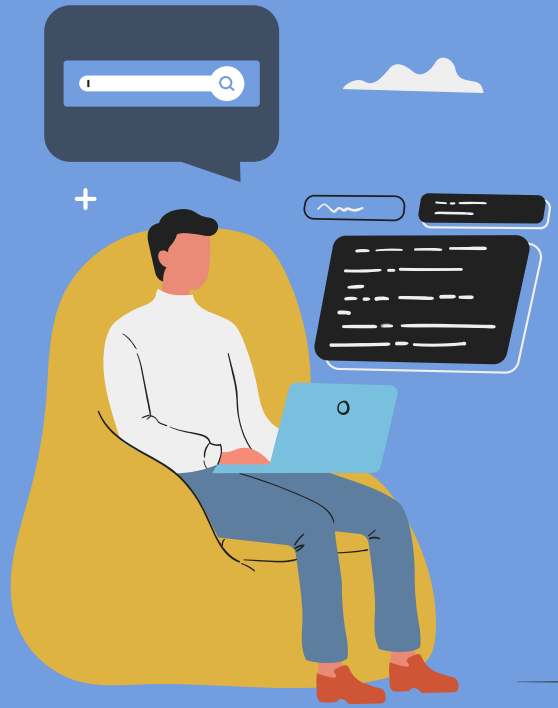


grid-column & grid-row

- מאפשר לנו "למתוח" פריט על פני מספר תאים.
- לכל קו ברשת יש מספר (מתחיל מ-1).
- grid-row-start | grid-column-start (התחל בקו שורה/עמודה.
- grid-row: 2 / 4 | grid-column: 1 / 3 (התחל בקו שורה/עמודה 2, סיים לפני קו 4).
- הקיצור: grid-column: span 2 (התפרס על פני 2 עמודות).



תרגול 2: פריסת עמוד

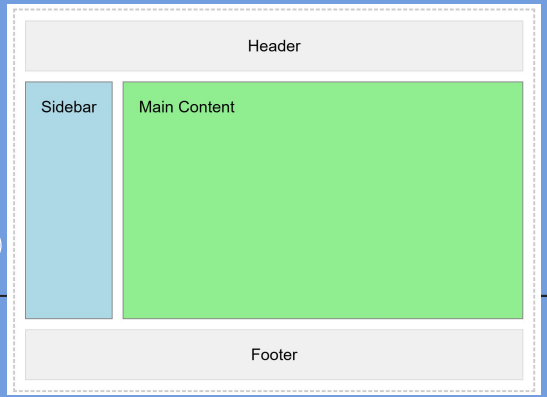


המשימה: צרו את מבנה האתר הקלאסי (Header, Sidebar, Main, Footer) בעזרת מיקום לפי מספרי קווים.

כלים: `grid-template-rows/columns`, `grid-row`, `grid-column`.

קישור : לחצו כאן.

תוצר נדרש:



grid-template-areas: לצייר את ה-Layout ב-CSS

- מאפשר לנו "למתוח" פריט על פני מספר תאים.
- השיטה האינטואיטיבית והוויזואלית ביותר להגדרת פריסה.
- **שלב 1 (ב-Container)**: מגדירים "מפה" של האזורים:

```
grid-template-areas:
  "header  header"
  "sidebar main"
  "sidebar extra"
  "footer  footer";
```

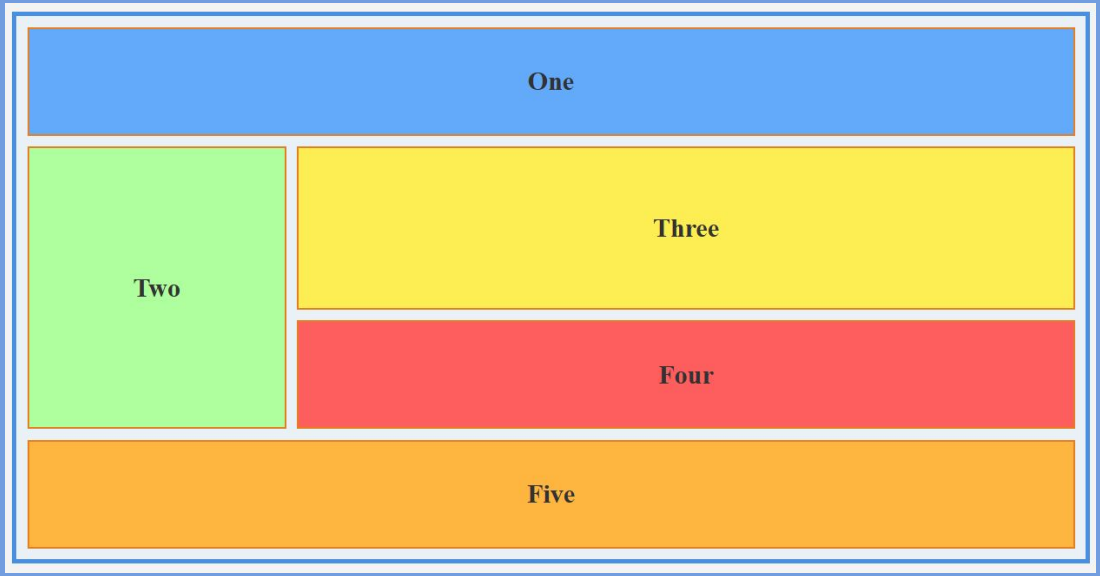
- **שלב 2 (ב-Item)**: משייכים כל פריט לאזור שלו:

```
.header{grid-area: header;}
.sidebar{grid-area: sidebar;}
.main{grid-area: main;}
.extra{grid-area: extra;}
.footer{grid-area: footer;}
```

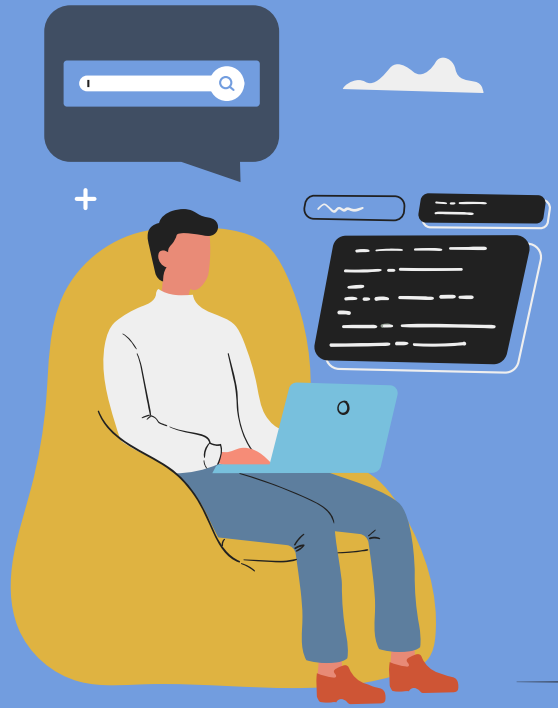
לצייר את ה-Layout ב-CSS: grid-template-areas

```
grid-template-areas:
  "header  header"
  "sidebar main"
  "sidebar extra"
  "footer  footer";
```

```
.header{grid-area: header;}
.sidebar{grid-area: sidebar;}
.main{grid-area: main;}
.extra{grid-area: extra;}
.footer{grid-area: footer;}
```



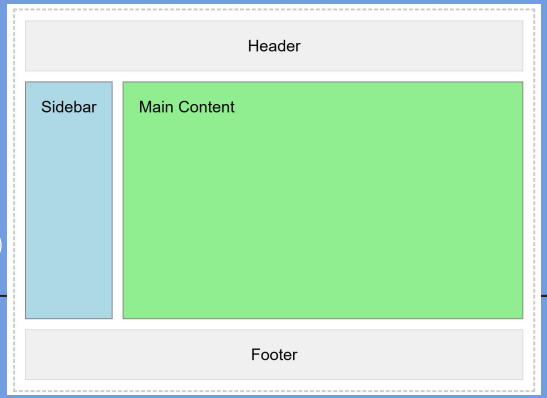
תרגול 3: פריסת עם אזורים



המשימה: בנו שוב את הפריסה מהתרגיל הקודם, אבל הפעם עם `grid-template-areas`.

קישור : לחצו כאן.

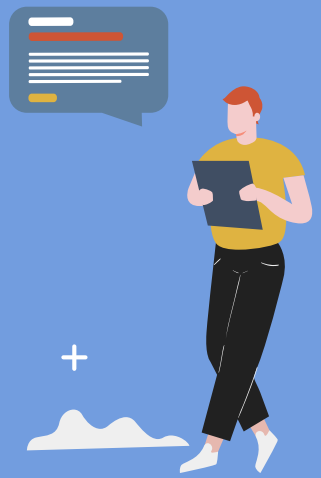
תוצר נדרש:



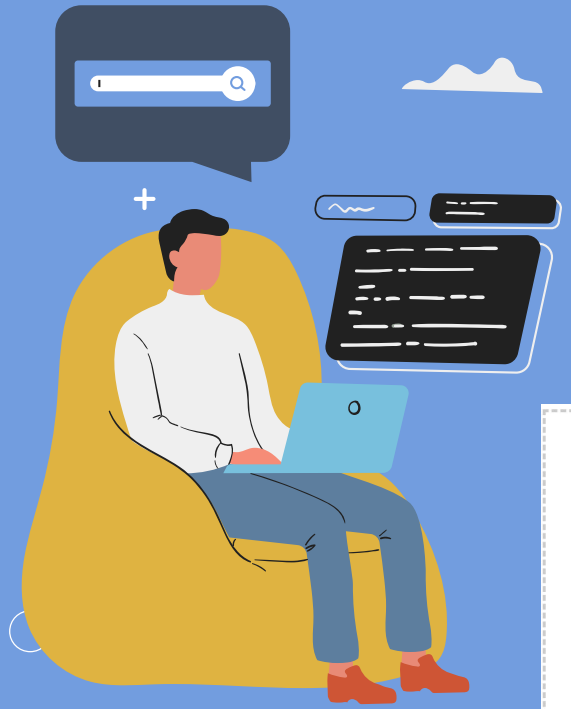
grid ב align-item/self | justify-content/self

- ב-Container (משפיע על כל הitems):
  - justify-items: יישור אופקי (ציר ה-inline).
  - align-items: יישור אנכי (ציר ה-block).

- ב-Item (דורס את ההגדרה בContainer ומשפיע על אלמנט ספציפי):
  - justify-self: יישור אופקי לפריט ספציפי.
  - align-self: יישור אנכי לפריט ספציפי.
  - ערכים נפוצים: stretch (ברירת המחדל), center, end, start.



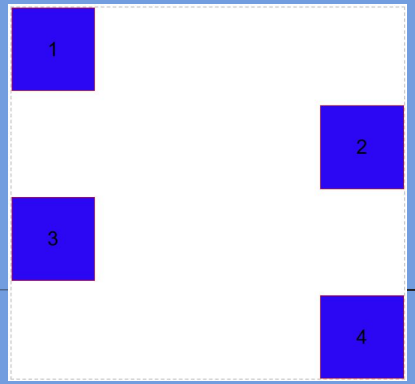
תרגול 4: משחקי יישור



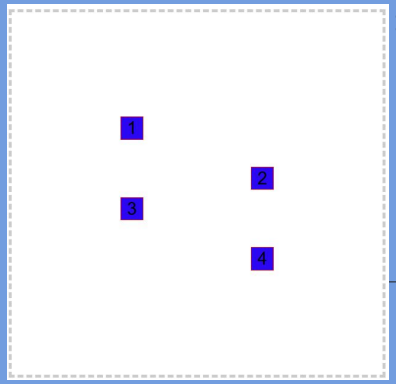
המשימה: צרו גריד פשוט ונסו למקם את הפריטים בפינות שונות של התאים שלהם. גריד עם 2 שורות ו2 עמודות שמתחלקות שווה בשווה עם ריווח של 10px, ומקמו כל אחד מהתאים כפי שמופיע ב"תוצר הנדרש".
בונוס: נסו להקטין את הgrid container ולמקם אותו במרכז.

קישור : לחצו כאן.

תוצר נדרש:

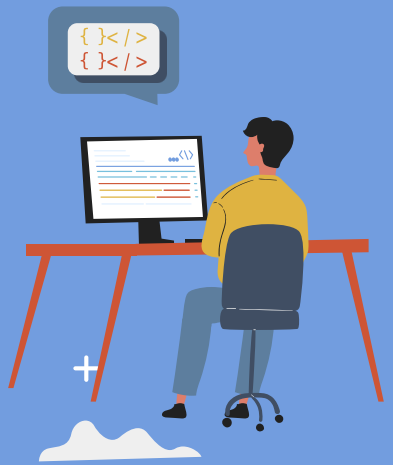


בונוס:



סיכום

- אז מה היה לנו?
  - Grid הוא הכלי האולטימטיבי לפריסה דו-ממדית.
  - `grid-template-columns/rows` ו-`gap` בונים את השלד.
  - `grid-template-areas` היא הדרך הקריאה והוויזואלית ביותר למקם פריטים.
  - `justify/align-items/self` נותנים לנו שליטה מלאה על היישור בתוך התאים.
  - כלל אצבע: Grid למבנה העמוד ולרכיבים מורכבים בתוכו, Flexbox לרכיבים פשוטים יותר שמכילים אלמנטים דומים.



תודה על ההקשבה

לינקים שימושיים

- **פתרון התרגילים:**
  - תרגיל 1 – באן
  - תרגיל 2 – באן
  - תרגיל 3 – באן
  - תרגיל 4 – באן
- **מקורות להמשך למידה:**
  - משחק מהנה ומלמד Grid Garden – באן
  - MDN Web Docs אתר מלא בדוגמאות, לינק ישירות לחלק שמדבר על Grid – באן