# מבוא לרשתות האינטרנט

# תרגיל בית #1

## 1. הגדר אינטרנט.

**האינטרנט** הוא רשת עולמית של מחשבים שמחוברים אחד לשני ו**מעבירים מידע** כמו תמונות, אתרים, סרטונים, קבצי קול ועוד בעזרת **פרוטוקולים**.

התקשורת נעשית בין מחשב ששולח בקשה (הלקוח - Client) לבין מחשב אחר שמחזיר תשובה (השרת - Server).

## 2. מהו WWW?

WWW הם ראשי תיבות של World Wide Web –הרשת הכלל-עולמית או "המארג הכלל־עולמי".

WWWהוא מרחב של מסמכי היפרטקסט מקושרים.

## 3. הגדר צד לקוח וצד שרת.

**לקוח :(Client)** מחשב או תוכנה ששולחים בקשה לשרת, לדוגמה כשגולשים באתר אינטרנט – הדפדפן שולח בקשה לשרת.

**שרת :(Server**) מחשב או תוכנה שמקבלים את הבקשות מהלקוחות, מטפלים בהן ושולחים בחזרה את המידע שהתבקש (כמו טקסט, תמונה, וידאו וכו'.(

## 4. הגדר פרוטוקול.

פרוטוקול הוא אוסף כללים שמגדירים כיצד מחשבים מתקשרים ומעבירים מידע ביניהם בצורה מסודרת וברורה.

## 5. הגדר פרוטוקול HTTP.

HTTP הוא קיצור של HyperText Transfer Protocol – פרוטוקול תקשורת שנועד להעברת דפי HTML (במיוחד ב-World Wide Web) ואובייקטים שהם מכילים (כמו תמונות, קובצי קול, סרטונים וכו'). הלקוח שולח בקשה, והשרת מחזיר תגובה בהתאם.

## 6. ציין שני דברים שעשויים להופיע ב- http request header

📋 טבלת HTTP Request Headers נפוצים

| **Header** | **תיאור בעברית** | **דוגמה** |
| --- | --- | --- |
| User-Agent | מזהה את הדפדפן, מערכת ההפעלה והמכשיר | Mozilla/5.0 (Windows 10) |
| Accept | אילו סוגי תוכן הלקוח מוכן לקבל מהשרת | text/html, application/json |
| Host | מציין את שם הדומיין של השרת שאליו נשלחת הבקשה | example.com |
| Content-Type | סוג התוכן שנשלח בבקשה (רלוונטי ל־POST/PUT) | application/json |
| Authorization | מידע לאימות המשתמש (כמו שם משתמש וסיסמה) | Bearer eyJhbGciOi... |
| Accept-Language | שפות מועדפות שהלקוח מעדיף לקבל | he-IL, en-US |
| Referer | כתובת העמוד שממנו הגיעה הבקשה | https://google.com |
| Cookie | מידע מהעוגיות השמורות בדפדפן | sessionId=abc123 |

## 7. ציין שתי מתודות מרכזיות בפרוטוקול HTTP, מתי נרצה להשתמש בכל אחת מהן?

**מתודה GET** – משמשת לבקשת משאבים מן השרת; מתאימה לקריאה בלבד.

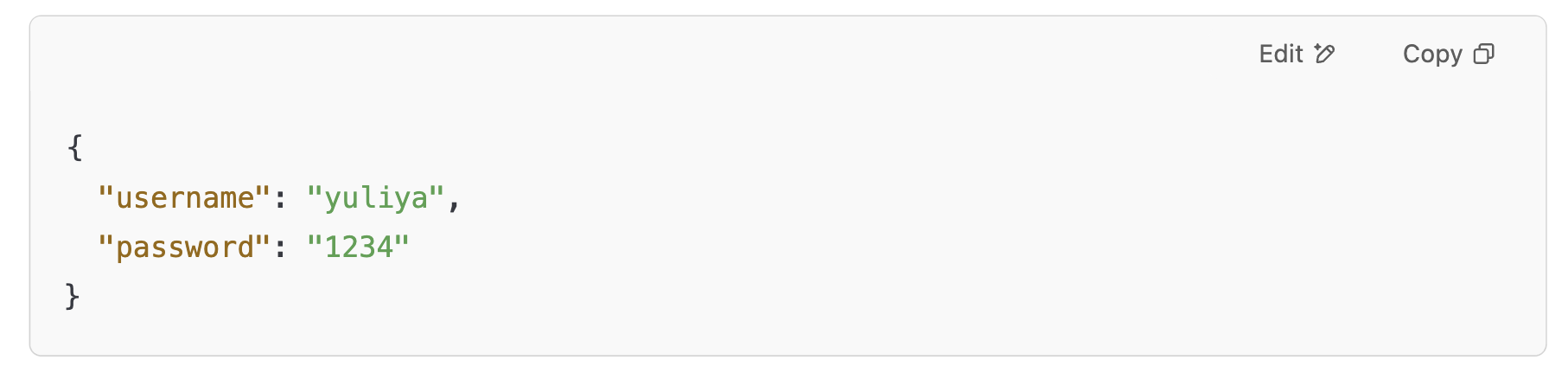
**מתודה POST –** משמשת ליצירת ושליחת מידע חדש לשרת, לדוגמה בטופסי הרשמה.

📊 השוואה בין מתודות HTTP נפוצות

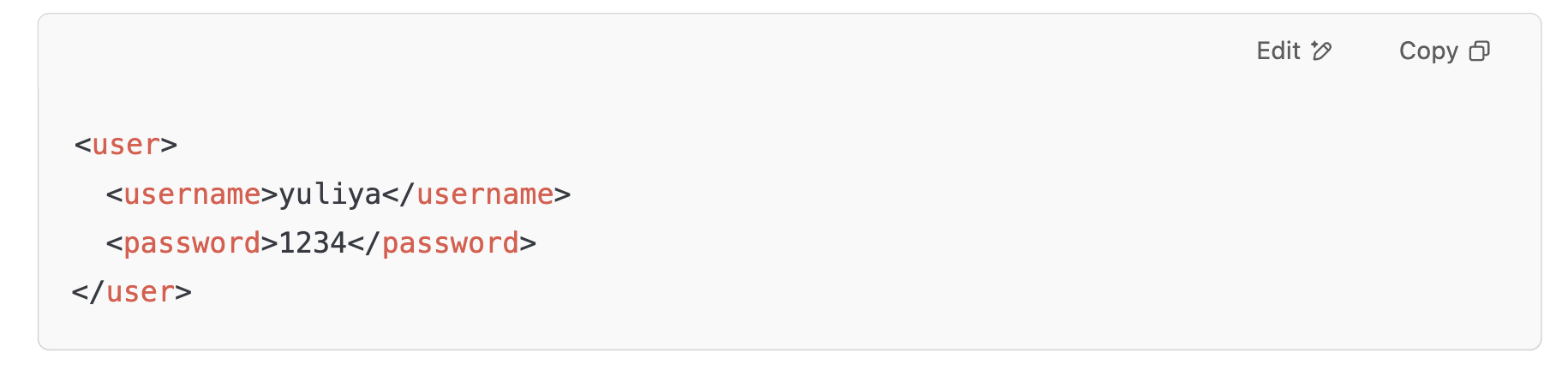
| **מתודה** | **פעולה עיקרית** | **האם יש ?body** | **האם משנה מידע בשרת?** | **שימוש נפוץ** |
| --- | --- | --- | --- | --- |
| GET | בקשת משאב / מידע | ❌ לא | ❌ לא | שליפת דף, תמונה, נתונים ממסד נתונים |
| POST | שליחת מידע חדש לעיבוד או יצירה | ✅ כן | ✅ כן | הרשמה, התחברות, יצירת תגובה |
| PUT | עדכון **מלא** של משאב קיים | ✅ כן | ✅ כן | עדכון פרופיל משתמש |
| DELETE | מחיקת משאב קיים | ✅ לעיתים | ✅ כן | מחיקת פוסט, תגובה, משתמש |

## 8. ציין שני פורמטים מרכזיים שעשויים לשמש לשליחת מידע ב-http request body.

**JSON (JavaScript Object Notation)**



**XML (eXtensible Markup Language)**

****

## 9. מהי התוכנה שיודעת לשלוח http request?

התוכנה ששולחת HTTP request נקראת דפדפן אינטרנט, כמו Chrome, Firefox, Safari. היא מאפשרת למשתמש (Client) לשלוח בקשות לשרת (Server) ולקבל את התוכן של אתרי אינטרנט.

## 10. ממה יוכל להיות מורכב http response body?

**HTTP response body** )גוף התגובה) זהו החלק שבו נמצא **המידע שהלקוח ביקש מהשרת**. הוא יכול להכיל למשל:

* **טקסט** – כמו HTML, XML או JSON
* **תמונות**
* **סרטונים**
* **קבצי קול**
* **קבצים להורדה** (PDF, DOC)

אם ה־ **HTTP response body**מכיל מידע בפורמט כמו **JSON** או ,**XML** הוא יהיה בנוי מ־**key-value** pairs – כלומר:

* **Key מפתח:** השם שמתאר את סוג המידע
* **Value ערך:** הנתון עצמו

## 11. מהו http response status? הבא שתי דוגמאות.

ה־**HTTP response status** הוא קוד מספרי שהשרת מחזיר ללקוח כדי להודיע מה קרה עם הבקשה – הצלחה, שגיאה, הפניה וכו'.

🔢 קודי סטטוס HTTP נפוצים

| קוד | שם | משמעות |
| --- | --- | --- |
| 200 | OK | הבקשה הצליחה והתקבל מידע |
| 201 | Created | המשאב נוצר בהצלחה (למשל, אחרי(POST |
| 204 | No Content | הבקשה הצליחה, אבל אין תוכן להחזיר |
| 400 | Bad Request | הבקשה לא תקינה (בעיה מצד הלקוח( |
| 403 | Forbidden | אין הרשאה לגשת למשאב |
| 404 | Not Found | הדף או המשאב לא נמצא |
| 500 | Internal Server Error | שגיאה כללית בשרת |
| 503 | Service Unavailable | השרת לא זמין כרגע (אולי בעומס או תחזוקה( |

## 12. מה הוא מבנה URL?

URLהוא כתובת של משאב באינטרנט. הכתובת בנויה לפי הסדר הבא:

scheme://domain:port/path?query#fragment

הסבר על כל חלק:

* – **Scheme** סוג הפרוטוקול שבו משתמשים. למשל.http, https
* – **Domain** שם האתר או השרת. לדוגמה. [www.example.com](http://www.example.com/)
* – **Port** מספר הפורט לתקשורת עם השרת. לדוגמה: 80 או 443. לא תמיד חייב להופיע.
* – **Path** הנתיב אל הדף או הקובץ המבוקש. לדוגמה: /products/item1
* – **Query** מידע נוסף שנשלח לשרת, כמו מזהה משתמש או סוג סינון. לדוגמהid=5&sort=asc
* – **Fragment** חלק מסוים בתוך הדף, למשל קפיצה ישירה לסעיף מסוים כמו section2

**דוגמה לכתובת URL מלאה:**  
https://www.ulidesserts.com/menu?type=parve#cakes.

## 13. ממה מורכבת כתובת ה-IP הסטנדרטית? (בונוס: ממה מורכבת כתובת IP בגרסה ipv6?)

IP – Internet Protocol

כתובת IP סטנדרטית (בגרסת (IPv4 מורכבת מארבעה מספרים בין 0 ל-255, שמופרדים בנקודות.  
למשל, 198.68.0.255  
כל מספר מייצג חלק שונה של הרשת או המחשב בתוך הרשת.

כתובת IPv6 היא גרסה חדשה ומורחבת של כתובת IP והיא מורכבת מ-8 קבוצות של מספרים הקסדצימליים (בסיס 16), שמופרדות בנקודותיים.  
למשל,2001:0db8:85a3:0000:0000:8a2e:0370:7334

היא מאפשרת הרבה יותר כתובות ונועדה להחליף את IPv4 בעתיד.

## 14. מהו שרת DNS?

שרתDNS (Domain Name System) הוא מחשב ברשת שתפקידו **להמיר שמות דומיין (כמו** [**www.google.com**](http://www.google.com) **) לכתובות IP מספריות** (כמו 142.250.185.36), כדי שהמחשב ידע לאן להתחבר.

אפשר לחשוב על DNS כמו על **"ספר טלפונים של האינטרנט"** – אנחנו זוכרים שמות, והמחשב צריך מספרים. ה-DNS עושה את התרגום הזה.

## 15. מהו המספר הדצימלי של המספר הבינארי 101? (רשות)

המספר הבינארי **101** מייצג את המספר הדצימלי .**5**

1×2² + 0×2¹ + 1×2⁰ = 4 + 0 + 1 = 5

💡 איך ממירים מספר בינארי לעשרוני?

כל ספרה במספר בינארי מייצגת [[1]](#footnote-1)**חזקה של 2**, לפי המיקום שלה – מימין לשמאל:

1 0 1

2² 2¹ 2⁰

ועכשיו נכפיל כל ספרה בחזקה המתאימה:

* הספרה הראשונה משמאל: 1 × 2² = 1 × 4 = 4
* הספרה האמצעית: 0 × 2¹ = 0 × 2 = 0
* הספרה הימנית: 1 × 2⁰ = 1 × 1 = 1

נחבר את כל התוצאות:  
**4 + 0 + 1 = 5**

## 16. חפש ברחבי הרשת: ב- http request האם ניתן לשלוח נתונים שלא ב-body?

(רמז: איך אפשר לשלוח נתונים במתודת ?GET)

בבקשות HTTP, ניתן לשלוח נתונים לא רק בגוף הבקשה ,(body) אלא גם באמצעות אמצעים אחרים, כמו פרמטרים בURL- וכותרות .(headers)

### ****16.1 שליחת נתונים בפרמטרים של URL****

במתודת , **GET** נהוג להעביר נתונים באמצעות פרמטרים ב URL הנקראים. "query parameters" הפרמטרים מצורפים ל URL-אחרי סימן השאלה ,(?)וכל פרמטר מורכב מזוג מפתח-ערך, כאשר פרמטרים מרובים מופרדים באמצעות אמפרסנד .(&)

לדוגמה:  
https://example.com/search?query=apple&sort=asc

במקרה זה, הפרמטר queryמכיל את הערך "apple" והפרמטר sort מכיל את הערך "asc".

### ****16.2 שליחת נתונים בכותרות (Headers):****

בנוסף, ניתן להעביר מידע בכותרות הבקשה .(HTTP headers) כותרות משמשות להעברת מידע מטא-דאטה על הבקשה או על הלקוח.

לדוגמה:   
Authorization: Bearer token123  
User-Agent: Mozilla/5.0

כותרת Authorization משמשת להעברת מידע אימות, וכותרת User-Agent מציינת את סוג הדפדפן או הלקוח השולח את הבקשה.

## 17. מהו element במבנה ?xml

אלמנט הוא יחידת מידע בסיסית בקובץ ,XML שמורכבת מתג פתיחה, תוכן, ותג סגירה.



האלמנטים הם: book, year, language

## 18. מהו content במבנה ?xml

ה־content הוא הטקסט שנמצא **בתוך** האלמנט בין תג הפתיחה לתג הסגירה.  
בדוגמה:

* האלמנט<year> מכיל את התוכן 1943
* האלמנט<language> מכיל את התוכן French

## 19. מהו attribute במבנה ?xml

Attributeהוא מידע נוסף או תכונה שמופיע בתוך **תג הפתיחה** של האלמנט ונותן פרטים על האלמנט.

title="The Little Prince"  
author="Antoine de Saint–Exupéry"

שני אלה הם attributes של האלמנט book.

## 20. ממה מורכב property במבנה JSON?

ב־**JSON** כל property (מאפיין) מורכב מ־**שם (key)** ו־**ערך ,(value)** כשהם מופרדים בנקודתיים.

**המבנה הכללי:**  
“key”: value

כל קובץ JSON מורכב מזוגות כאלה, והם נכתבים בתוך סוגריים מסולסלים { }.

## 21. מה הוא מערך **(Array)** במבנה JSON?

מערך ב**JSON-** הוא רשימה של ערכים, שמופיעה בתוך **סוגריים מרובעים**[ ] .  
הערכים במערך יכולים להיות מכל סוג: מספרים, מחרוזות (טקסט), ערכים בוליאניים ,(true/false) אובייקטים, או אפילו מערכים נוספים.

**דוגמה למערך של מחרוזות:**

"colors": ["red", "green", "blue"]

**דוגמה למערך של אובייקטים:**

"students": [

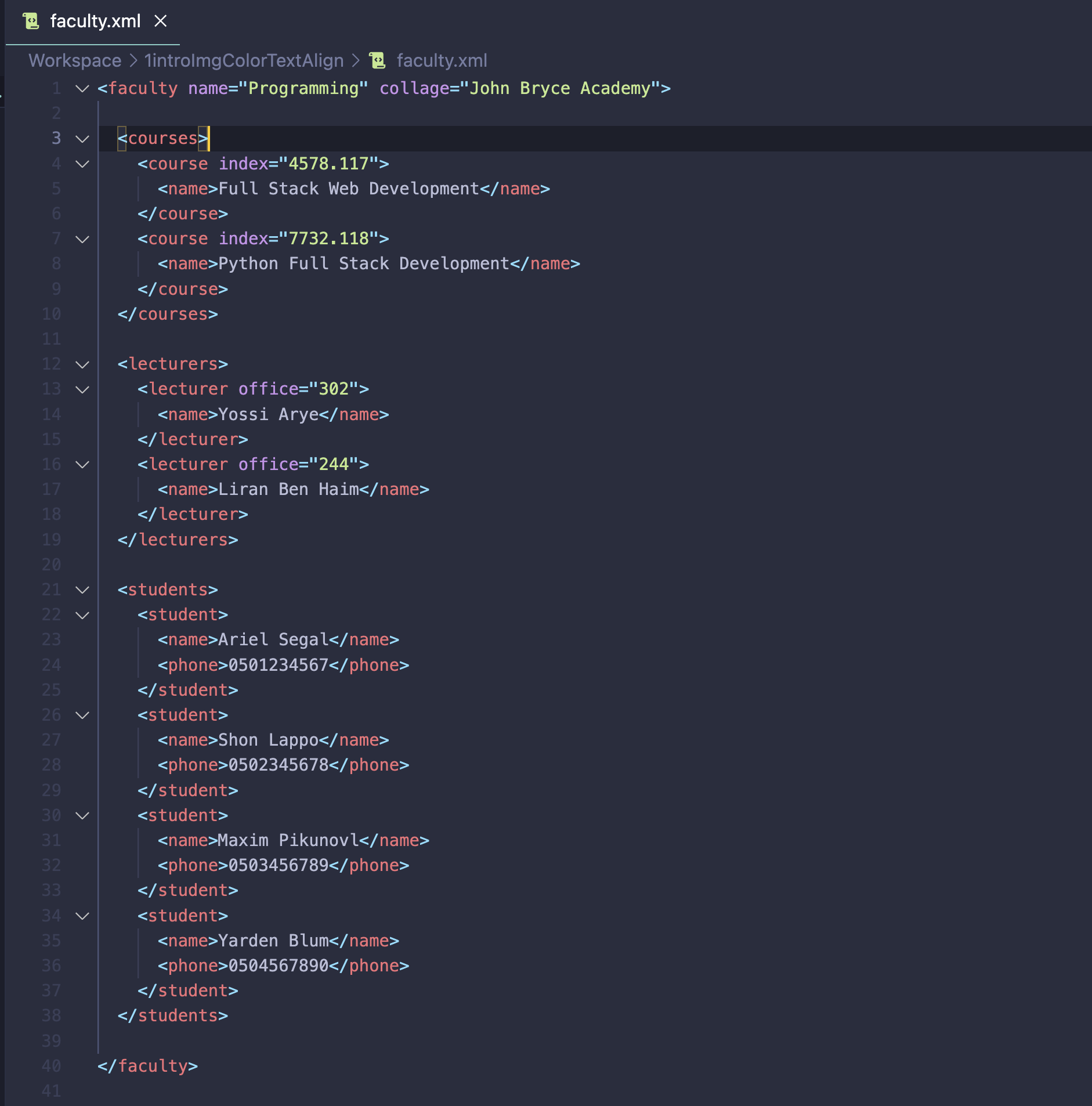
{ "name": "Yuliya", "grade": 10 },

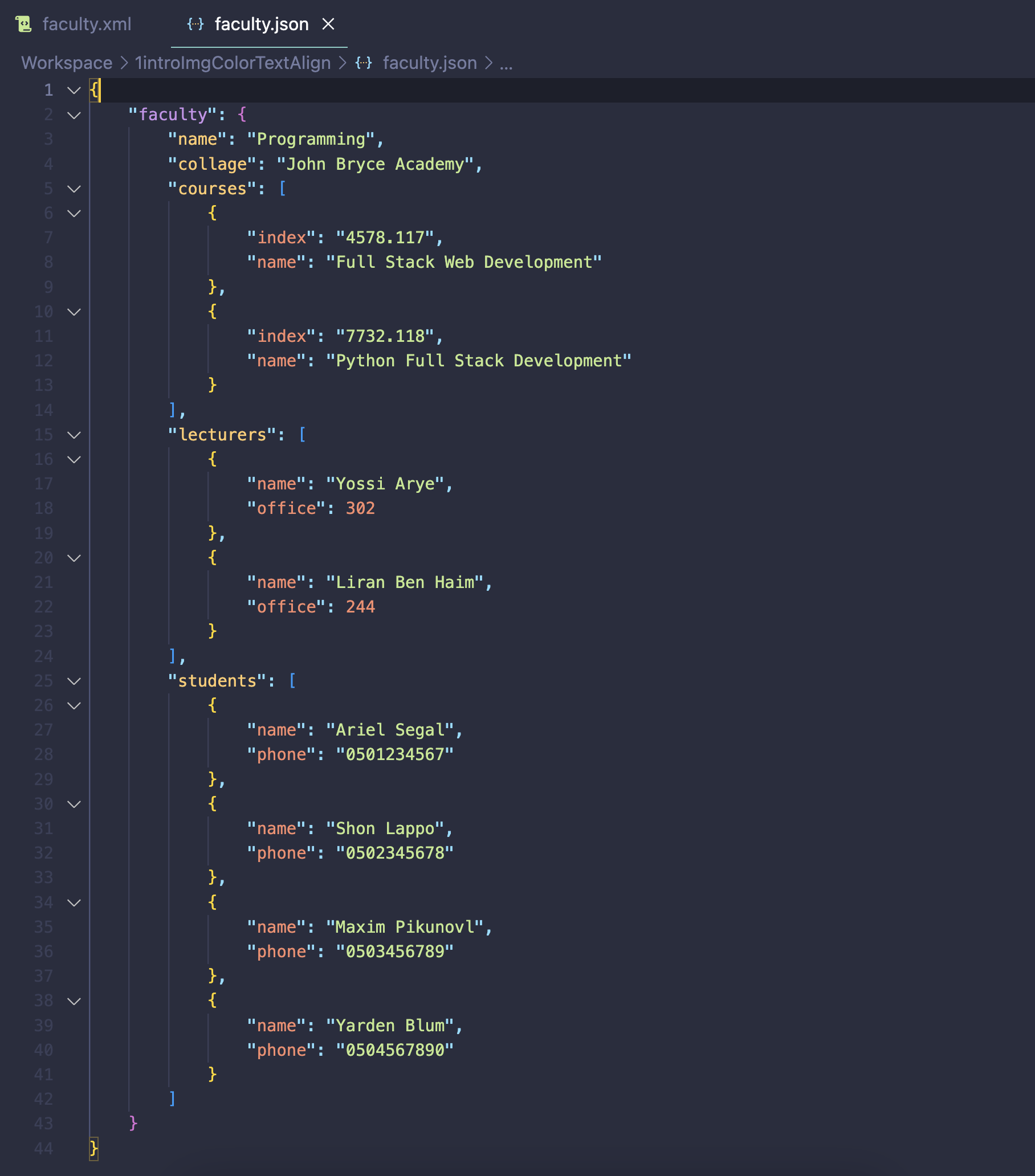
{ "name": "Lior", "grade": 11 }

]

🔸 המערך מאפשר לשמור כמה ערכים תחת אותו מפתח (key) בצורה מסודרת.

22. קודד פקולטה שמורכבת משני קורסים, שני מרצים וארבע סטודנטים, כולל תכונות (attributes) במבנה XML ו-JSON.





# תרגיל בית #2

## 1. מה תפקידו של doctype בקובץ ה־HTML?

ה־<!DOCTYPE html> מציין לדפדפן באיזו גרסה של HTML כתוב המסמך. נכון לעכשיו מדובר ב-HTML5.

## 2. חפש ברשת: האם אפשר לכתוב doctype באותיות קטנות?

כן, ניתן לכתוב <doctype html!> באותיות קטנות. זה תקני, מכיוון ש־DOCTYPE אינו case sensitive בשפת HTML5.

NB (השאלה עד כמה זה נהוג...)

## 3.1. מה זה case sensitive?

שפה שהיא **case sensitive** מבדילה בין אותיות **קטנות** ל־**גדולות**.

למשל: Variable, variable, ו־VARIABLE נחשבים שלושה מזהים (identifier) שונים לגמרי.

דוגמאות לשפות**case sensitive**:

| שפה | הסבר |
| --- | --- |
| **Java** | int num ≠ int Num |
| **JavaScript** | myVar ≠ MyVar |
| **Python** | name ≠ Name |

## 3.2 מה זה ?case insensitive

שפה **case insensitive** לא מבדילה בין אותיות קטנות וגדולות.  
למשל: TITLE, title, ו־TiTlE נחשבים אותו דבר.

דוגמאות לשפות **case insensitive**:

| שפה | הסבר |
| --- | --- |
| **HTML** | <h1> זהה ל־<H1> |
| **SQL** | SELECT = select = SeLeCt |

## 4.1. הוסף את הקישור הבא כ־src של תמונה (img)

<img src="https://upload.wikimedia.org/wikipedia/commons/6/68/Joe\_Biden\_presidential\_portrait.jpg" style="width: 100%;">



## 4.2 בדוק ב־DevTools עבור מגוון של מידות מסך,

מה קורה כאשר נשים בהערה את תגית

<meta name="viewport" content="width=device-width, initial-scale=1.0">?

כאשר תגית ה־viewport קיימת, הדפדפן **מתאים את רוחב הדף לרוחב המכשיר בפועל**, ומאפשר עיצוב רספונסיבי אמיתי באמצעות CSS. [[2]](#footnote-2)

## 4.3 מה מסקנתך לגבי תגית meta זו?

**מסקנה:** תגית <meta name="viewport"> קריטית *לעיצוב מותאם* למסכים שונים, והיא חובה בכל אתר מודרני שרוצה להיות מותאם לניידים וטאבלטים.

## 5. מה הוא root element?

ה־root element הוא התג הראשי והאלמנט העליון ביותר במבנה היררכי של מסמך או נתונים שמכיל את כל תוכן הדף. ב־HTML זהו תג <html> .

## 6. בשיעור הוספנו CSS לתמיכה בסטייל בצורה inline – הייתם ממליצים על זה? למה?

❌ לא מופרד מהתוכן – מקשה על קריאות ותחזוקה.

❌ שימוש חוזר כמעט בלתי אפשרי – אי אפשר להחיל אותו על אלמנטים אחרים בלי לשכפל קוד.

❌ שובר את עקרון ההפרדה בין תוכן לעיצוב (Separation of Concerns).

## .7 font-size ניתן להגדרה על ידי px, %, rem,

כך ש-

* **px** גודל קבוע בהתאם להגדרה
* **%** יחסי לגודל של האלמנט האב.
* **rem** משנה את font size ביחס ל־font size של ה־root element

בהינתן שכך מה יהיו הגדלים של ה־p elements עבור קטע ה־HTML הבא, אחרי ששינינו ב־dev-tools את font-size של ה־<html> element ל־20px ?



התשובה:



# תרגיל בית #3

## 1. הסבר מהי 🎨 RGB?

RGBהוא מודל צבעים שמבוסס על שילוב של **אדום ,(Red)** **ירוק (Green)** ו־**כחול .(Blue)**

התחביר: rgb) אדום, ירוק, כחול — (כל ערך בין **0 ל־255.**

לדוגמה:

Red = rgb(255, 0, 0)

Green = rgb(0, 255, 0)

Blue = rgb(0, 0, 255)

ככל שהמספר גבוה יותר – הצבע חזק יותר. שילוב של שלושת הצבעים יוצר גוונים שונים.

– 🎨 RGBAצבע עם שקיפות

כמו,rgb אבל עם ערך רביעי לשקיפות .(Alpha)

התחביר: rgba) אדום, ירוק, כחול, שקיפות(

השקיפות היא בין) 0 שקוף) ל־) 1אטום(.



## 2. תאר את הצבע RGB 237,25,143 לפורמט הקסדצימלי (hex).

rgb(237, 25, 143) → hex = #ED198F   
זה צבע בגוון ורוד־פוקסיה

כדי להפוך כל אחד מהם להקסדצימלי[[3]](#footnote-3), מחלקים כל מספר ב־16.

שלב 1: אדום – 237

1. 237 ÷ 16 = 14 עם שארית 13
2. 14 = E, 13 = D

תוצאה**ED :**

שלב 2: ירוק – 25

1. 25 ÷ 16 = 1 עם שארית 9
2. 1 = 1, 9 = 9

תוצאה: 19

שלב 3: כחול – 143

1. 143 ÷ 16 = 8 עם שארית 15
2. 8 = 8, 15 = F  
   תוצאה**8F** :

## 3. נתון border-width: 10px 12px 7px, מהו הרוחב של הגבול ב-4 פאותיו?

תשובה:

| פאה | רוחב הגבול |
| --- | --- |
| למעלה (Top) | 10px |
| ימין (Right) | 12px |
| למטה (Bottom) | 7px |
| שמאל (Left) | 12px |

הסבר:

CSS Border Shorthand (TRBL)

כשמשתמשים בקיצור ,border-width הערכים ניתנים לפי הסדר הבא:

**T R B L**  
Top → Right → Bottom → Left

ערך אחד = לכל הפאות

שני ערכים = [Top & Bottom], [Right & Left]

שלושה ערכים = [Top], [Right & Left], [Bottom]

ארבעה ערכים = [Top], [Right], [Bottom], [Left]

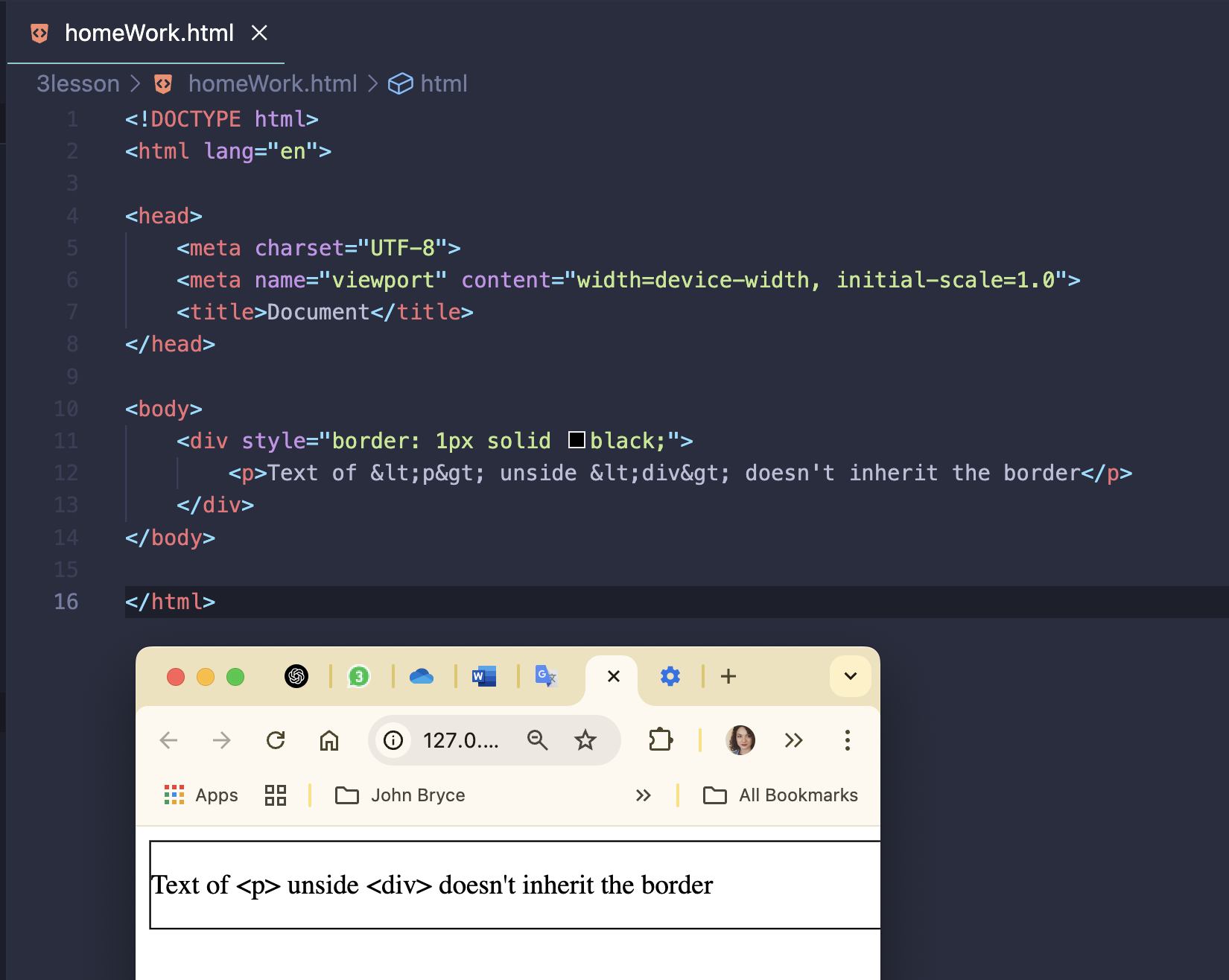
## 4. קבע נכון או לא נכון: תמיד סטייל עובר בירושה. הבא דוגמה שתתמוך בקיעה.

❌ **לא נכון** – לא כל תכונה (style) ב־CSS עוברת בירושה.

הסבר:

ב־CSS, רק **חלק** מהתכונות עוברות בירושה אוטומטית (כמו צבע טקסט, גודל גופן, כיוון כתיבה). רוב התכונות הוויזואליות **לא עוברות בירושה** אלא אם מציינים זאת במפורש עם inherit.

דוגמה שתומכת בקביעה (שלא תמיד עובר):



כאן, ה־ <div> מקבל גבול,  
אבל ה־<p> **לא מקבל גבול**– כי border **לא עובר בירושה**.

## 5. מהם שלושת האפשרויות לקביעת סטייל עבור קובץ html, מתי והאם בכלל נשתמש בכל אחת מהן?

Style Adding Methods

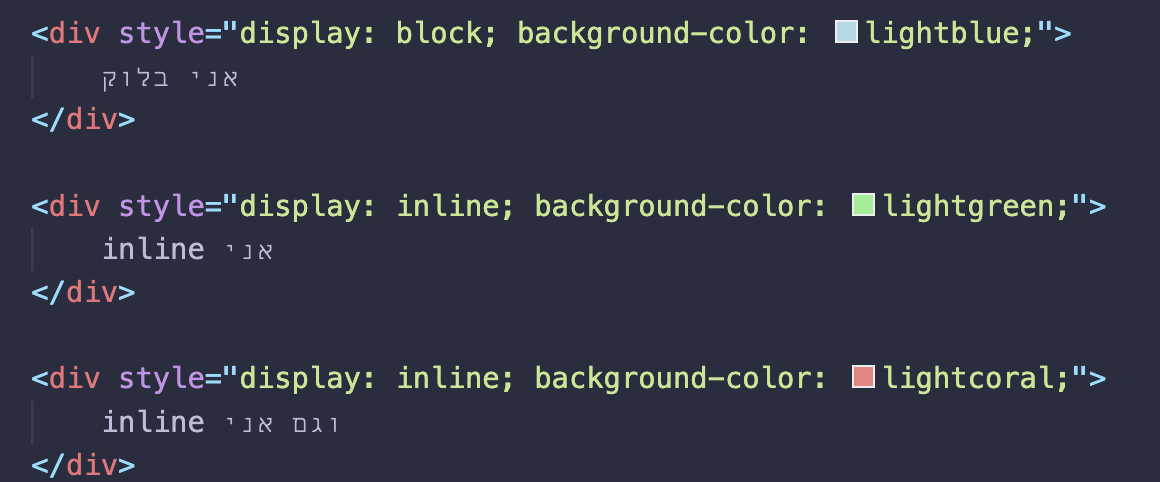
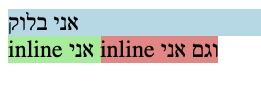
| שיטה | איפה כותבים? | מתי משתמשים? | יתרון | חסרון |
| --- | --- | --- | --- | --- |
| Inline | בתוך התגית (<p style=”color: red”>) | כשצריך לעצב רק אלמנט אחד | הכי מהיר | קשה לשנות אחר כך;  מפר SoC  **Separation of concerns**  הפרדת דאגות |
| Internal | בתוך <style>בקובץ ה־HTML | כשיש עמוד **אחד** בלבד | הכול באותו קובץ | לא נוח כשיש הרבה עמודים 🔁 **DRY = Don't Repeat Yourself[[4]](#footnote-4)** |
| External | בקובץ CSS נפרד עם <link> | כשיש כמה עמודים באתר | מסודר ונוח לשינויים | צריך קובץ נוסף |

## 6. כאשר הוספנו ל־ div selector את הפרופרטי ,display: inline התצוגה על המסך קטנה מאוד – רק ריבוע קטן בצד. מדוע?

כאשר מוסיפים ל־ div את display: inline, האלמנט מתנהג כמוinline element (כמו span>>) .כלומר:

* הוא **לא מקבל רוחב וגובה** כמו בלוק רגיל
* הוא לא שובר שורה
* הוא מתכווץ לגודל התוכן בלבד

מאחר ש־div הוא היה אלמנט חסר תוכן, הוא **מצטמצם לריבוע קטן** שמופיע בצד (כמו מילה בשורה.(



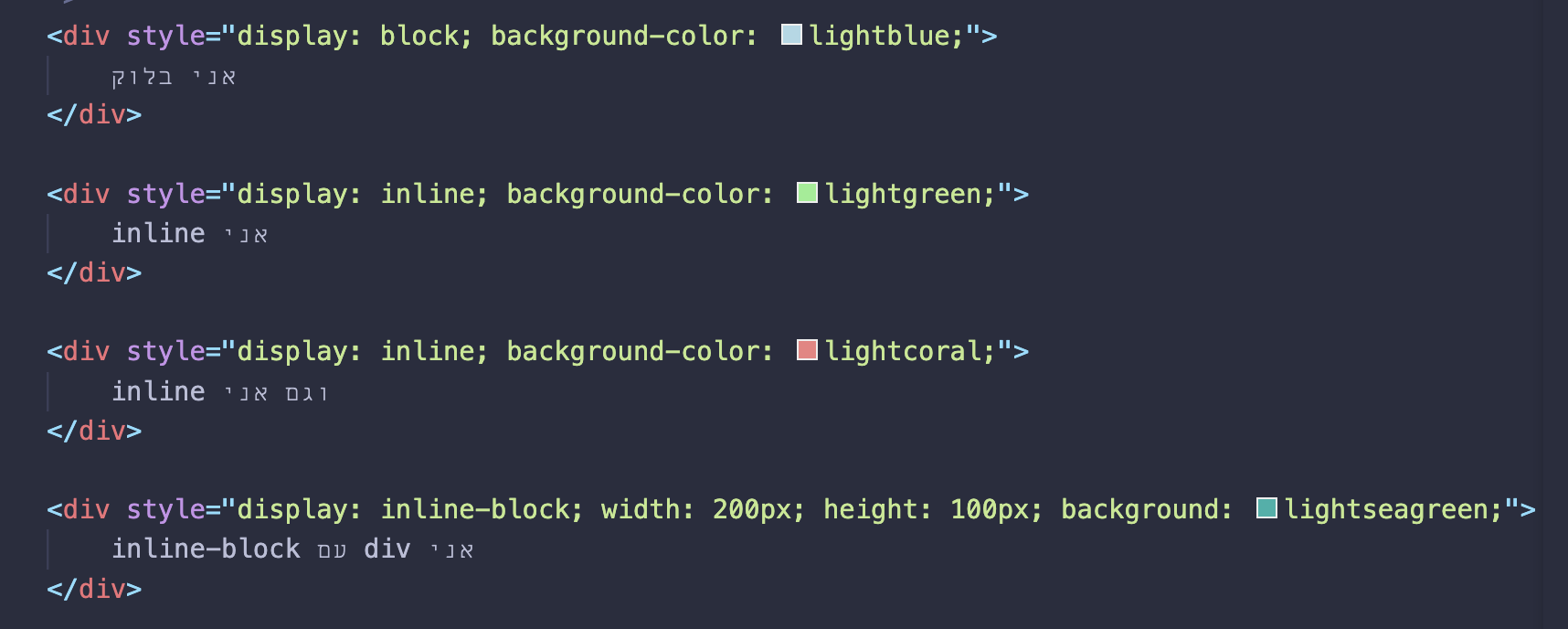
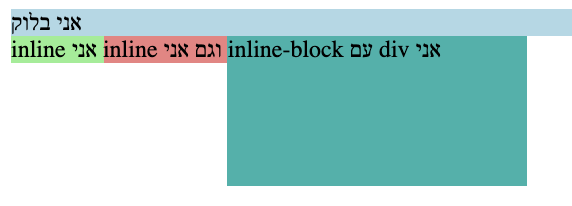
## 7. אם אני רוצה לתת ל־ div את התכונות של inline אבל מבלי שהתצוגה תקטן לגדלים הלא רצויים, מה עליי לעשות?

צריך להשתמש ב־display: inline-block.

📌הסבר:

inline-block משלב את הטוב משני העולמות: מתנהג כמו **inline,** לא שובר שורה;

אבל מאפשר להגדיר **רוחב (width) וגובה (height)** כמו .block



# תרגיל בית #4

1. צור תיקייה A והכנס לתוכה 2 משאבים, האחד משאב C שיהיה קובץ html והשני תיקייה B ובה משאב D שיהיה קובץ .html קרא לתיקיות A, B ולקובץ ה html בשמות משמעותיים.  
   צור 2 לינקים ב D שיפנו ל C כך שהלינק הראשון יהיה relative url והשני יהיה absolute url
2. א. צור קובץ html שידמה את האתר הבא:

Home Page

* html
* css
  1. inline
  2. internal
  3. external
* js

full stack  
front end  
back end  
data base

back end  
nodejs  
php  
python

הכנס את הקובץ שיצרת לתוך תיקיה בשם topics ולאחר מכן הכנס את topics לתיקייה A שיצרת בתרגיל 1. צור קישור מהקובץ C משאלה 1 לקובץ שיצרת ומהקובץ שיצרת נוכל לפנות חזרה לקובץ C כאשר נלחץ על הלינק Home Page.

ב. השתמש ב pseudo class כדי לצבוע את כל ה li ו dd הזוגיים בצבע כחול וכל li ו dd שלישי בצבע זהב.

ג. השתמש ב relational selectors על מנת שכל ה dd שבאים אחרי dt הראשון יהיו בפונט בגודל .px45

## 4. מה מסמל ה-1,1,0 selector specificity?

**מהי ספציפיות (Specificity) בCSS?**

ספציפיות היא הדרך שבה הדפדפן מחליט איזו הוראת CSS תחול כאשר כמה חוקים מכוונים לאותו אלמנט.

ככל שהסלקטור (הבוחר) יותר מדויק וספציפי – כך העדיפות שלו גבוהה יותר.

הערך 1,1,0 בספציפיות (Specificity) של סלקטור ב־CSS מתאר עד כמה סלקטור מסוים "חזק" ומשפיע על אלמנט ביחס לסלקטורים אחרים.

הפורמט הכללי של ספציפיות הוא:  
**a,b,c**  
כאשר:

– **a** מספר סלקטורים לפי id

– **b** מספר סלקטורים לפי class, attribute ו־pseudo-class (כמו hover, :focus)

– **c** מספר סלקטורים לפי תג (element) ו־ pseudo-element (כמו ::before, ::after)

|  |  |  |
| --- | --- | --- |
| Selector Part | Type of Selector | Specificity Value |
| #id | ID selector | 1,0,0 |
| .class, [attr], :hover | Class, attribute, or pseudo-class selector | 0,1,0 |
| div, p, h1, ::before | Type (element) or pseudo-element selector | 0,0,1 |
| Inline style (e.g. style="...") | Inline CSS style | 1,0,0,0 *(treated as strongest)* |

## 5. נתון דף html עם אלמנט p ולו ה־attributes הבאים: id="main" class="p1". בקובץ הסגנון המקושר ל־html, נתון סגנון ל־id ששמו main כך שצבע הטקסט של האלמנט שהוא מייצג יהיה כחול. בקובץ ה־html יוגדר כלל סגנון לכל האלמנטים שמסווגים ל־p class1 כך שצבע הטקסט שלהם יהיה זהב. מה יהיה צבע הטקסט של האלמנט ?p

**תשובה:** צבע הטקסט של האלמנט יהיה **כחול**.

**הסבר:** ב־CSS קיימת היררכיית ספציפיות ,(Specificity) לפיה נבחר הסגנון שיש לו עדיפות גבוהה יותר.

* כלל שמבוסס על id )כמו #main ) נחשב לבעל **ספציפיות גבוהה יותר** מכלל שמבוסס על class .
* במקרה הזה יש התנגשות בין שני כללים:
  + #main { color: blue; } – ספציפיות: (1,0,0)
  + .p1 { color: gold; } – ספציפיות: (0,1,0)
* הכלל עם ה־id גובר, ולכן צבע הטקסט של האלמנט יהיה **כחול**.

# תרגיל בית #5

## 1-4. נתון קטע הקוד. מה יהיה צבע והמידות של האלמנטים?

### 1. div.container

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Element** | **div1** | **p1** | **div** | **p2** | **p3** | **div2** | **p4** |
| **height** | 300px+2px border=302px | 16px default font-size+2px= 18px line-height | 18px | 18px | 18px | 18px | 18px |
| **width** | 300px+2px border=302px | 300px | 300px | 300px | 300px | screen width | screen width |
| **color** | blue with a black border | blue (border is not an inheritable CSS property) | blue | blue | blue | default | default |

### 2. div .container

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Element | div1 | p1 | div | p2 | p3 | div2 | p4 |
| **height** | 18px line-height \*3 + 16px margine\*2 = 86px | 18px | 18px | 18px | 18px | 18px | 18px |
| **width** | screen width | screen width | screen width | screen width | screen width | screen width | screen width |
| **color** | default | default | default | default | default | default | default |

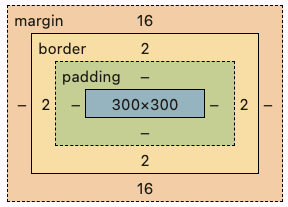
### 3. div p

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Element | div1 | p1 | div | p2 | p3 | div2 | p4 |
| **height** | 302px (p height) \*3 + 16px margin \*2 = 938px | 300px+2px border=302px | 300px+2px border=302px | 300px+2px border=302px | 300px+2px border=302px | 300px+2px border=302px | 300px+2px border=302px |
| **width** | screen width | 300px+2px border=302px | screen width | 300px+2px border=302px | 300px+2px border=302px | screen width | 300px+2px border=302px |
| **color** | default | blue with a black border | default | blue with a black border | blue with a black border | default | blue with a black border |

### 4. container > p

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Element | div1 | p1 | div | p2 | p3 | div2 | p4 |
| **hight** | 302px + 16px margine + 18px line-height + 16px margine + 302px = 654px | 300px + 2px = 302px | 18px | 18px | 300px + 2px = 302px | 18px | 18px |
| **width** | screen width | 300px+2px = 302px | screen width | screen width | 300px + 2px = 302px | screen width | screen width |
| **color** | default | blue with a black border | default | default | blue with a black border | default | default |

## 5. מהם ארבעת המרכיבים של ה- box model?

**📦 ארבעת המרכיבים של ה־Box Model:**

1. **Content (תוכן)**   
   זהו האזור שבו נמצא הטקסט או התמונה של האלמנט.
2. **Padding (ריפוד)**   
   הרווח בין התוכן לבין הגבול (border).   
   יוצר "מרווח פנימי" מסביב לתוכן.
3. **Border (גבול)**   
   מסגרת שמקיפה את ה־padding ואת התוכן.
4. **Margin (שוליים)**   
   הרווח **החיצוני** שמפריד את האלמנט מאלמנטים אחרים בדף.

## 6. האם האלמנט כולל את כל ארבעת המרכיבים?

❌ Margin (שוליים) לא כלול בגבולות הפיזיים של האלמנט.

Margin אינו חלק מהאלמנט עצמו, אלא מרווח חיצוני שמפריד בינו לבין אלמנטים אחרים.

## 7. מה תפקיד של הסטייל? box-sizing הסבר את שתי האפשרויות.

ה־ CSS property בשם **box-sizing** קובע **איך הדפדפן מחשב את הגודל הכולל של האלמנט** (רוחב וגובה) – כלומר, האם ה־padding וה־ border כלולים בתוך ה־width וה־ height או שמתווספים אליהם.

### שתי האפשרויות העיקריות:

|  |  |  |
| --- | --- | --- |
| Property | content-box (default) | border-box |
| **Box-sizing** | content-box | border-box |
| **How size is calculated** | Width & height = content only.  Padding and border are added on top. | Width & height include padding and border. |
| **Example CSS** | .box {  width: 200px;  padding: 10px;  border: 2px solid black;  box-sizing: content-box;  } | .box {  width: 200px;  padding: 10px;  border: 2px solid black;  box-sizing: border-box;  } |
| **Actual total size** | 200 + 10×2 + 2×2 = 224px | 200px exactly (padding & border included) |

## 8. למה ה־content (תיבת התוכן) תמיד גדול מה־font-size?

כי content box כולל גם רווח סביב הטקסט, לא רק את האותיות עצמן.

סיבות עיקריות:

* line-height (ריווח טבעי) – מוסיף שטח מעל ומתחת לאותיות, גם אם לא הגדרנו אותו בעצמנו.
* חלקים של אותיות שיורדים או עולים (כמו g, y, f, l – ) הדפדפן מוסיף מקום כדי שלא ייחתכו.
* ה־font-size מודד רק את גובה האותיות, לא את כל הרווח סביבן.

## 9. נתון קטע הקוד. מה יהיו המידות של האלמנטים?

|  |  |  |
| --- | --- | --- |
| Element | p | div |
| **hight** | 30px padding\*2 + 10px border\*2 + 16px line-height (1em is equal to the font-size) = 96px | 96px p-hight\*2 + 20px\*1 (margin collapsing) = 212px |
| **width** | 30px padding\*2 + 10px border\*2 + 300px content-width = 380px | screen width (by default) |

## 10. נתון קטע הקוד. מה יהיו המידות של האלמנטים?

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Question | 10.1 | 10.2 | 10.3 | 10.4 | 10.5 |
| **CSS** | div {  width: 300px;  height: 300px;  background-color: gold; } | body { width: 600px; }  div {  width: 50%;  /\*The <div> will be 50% of its parent, which is <body>.\*/  height: 50%;  /\*Here’s the trick: percentage-based height requires the parent to have a defined height.  In the code, the <body> does not have a defined height,  so: 👉 height: 50% has no effect — the div will have height 0\*/  background-color: gold;  border: 1px solid black; } | body {  width: 600px;  /\*The body { width: 600px; } does not affect the <div> in this case, because the <div> uses viewport units – not relative to its parent.\*/  }   div {  width: 50vw;  /\*vw = viewport width  1vw = 1% of the browser window’s width\*/  height: 50vh;  /\*vh = viewport height  1vh = 1% of the browser window’s height\*/  background-color: gold;  border: 1px solid black;  } | body {  width: 600px;  /\*The body { width: 600px; } does not limit the <div>.\*/  }    div {  font-size: 10px;  /\*em is a relative unit in CSS. It’s used to size elements (like font-size, padding, margin, width, etc.) based on the current font size.    1em = 100% of the current font size  If the font size is 16px, then 1em = 16px  2em = 32px, 0.5em = 8px, etc.\*/  width: 50em;  height: 50em;  background-color: gold;  border: 1px solid black;  } | body {  width: 600px;  }  /\*rem stands for root em  It’s a relative unit, just like em, but instead of being based on the element’s font size, it’s always based on the root element's font size — typically the <html> element.  font-size: 16px is default in most browsers \*/    div {  font-size: 10px;  /\* The <div> uses rem, so it ignores its own font-size (10px) for sizing.  It calculates based on root font-size (16px)\*/  width: 50rem;  /\* The <body> is only 600px wide, but the <div> will overflow it horizontally because it's 800px wide. \*/  height: 50rem;  background-color: gold;  border: 1px solid black;  } |
| **hight** | 300px | 0 body hight\*50%+2px border=2px | 50% of the full browser height +2px border | 10px font-size\*50+2px border=502px | 16px defauil font-size for html\*50+2px border=802 |
| **width** | 300px | 600px body width\*50%+2px border=302px | 50% of the full browser width +2px border | 10px font-size\*50+2px border=502px | 16px defauil font-size for html\*50+2px border=802 |

## 11. Positioning

צור קובץ סטייל נפרד לכל סעיף מהסעיפים הבאים.

|  |  |
| --- | --- |
| 11. צור div בגודל 600x600 (בפיקסלים) ובתוכו שלושה div’s שגודלם 100x100 (בפיקסלים), תן ל־div האב border כלשהו, ולשלושת בניו צבעי רקע שונים.  <div class="conteiner">  <div id="first"></div>  <div id="second"></div>  <div id="third"></div>  </div> |  |
| **11.1 הזז את הריבוע השני 15 פיקסלים משמאל לימין ומלמעלה למטה באופן שלא יסתיר את אחיו.**  **כלומר**: להשתמש ב־margin , ולא ב־position.  #second {  margin-left: 15px;  margin-top: 15px; } |  |
| 11.2 קבע את הריבוע השני בפינה הימנית העליונה של האב. **כלומר**: להשתמש ב־position: absolute ביחס לאב.  .conteiner {  position: relative[[5]](#footnote-5); }  #second {  position: absolute;  top: 0;  right: 0; { |  |
| 11.3 קבע את הריבוע השני 200 פיקסלים מראש ושמאלה מעמוד האינטרנט כך שגם בגלילה הוא יישאר שם.  **כלומר**: להשתמש בposition: fixed -.   * position: fixed **מקבע** אלמנט **למסך**(viewport) ולא למיקום בעמוד עצמו. * האלמנט **נשאר תמיד באותו מקום** במסך, גם כשגוללים את העמוד למעלה או למטה.   #second {  position: fixed;  top: 200px;  left: 200px;  } |  |

# תרגיל בית #6

## 1. תאר מהי בעיית parent collapsed ב־float?

**Float**מתאים כאשר רוצים **להצמיד תמונה** לצד שמאל או ימין של טקסט, כדי שהטקסט **יעטוף** את התמונה. **(לא מומלץ להשתמש ב־float** לצורך **מבנה עמוד שלם**.)

כאשר ילדים בתוך אלמנט משתמשים ב־float , ההורה לא "מרגיש" את הגובה שלהם (כאילו אין בתוכו כלום), ולכן הגובה של אלמנט האב קורס לגובה 0 – וזו ״קריסת ההורה״ [(collapsed parent)](https://css-tricks.com/all-about-floats/#aa-the-great-collapse).



### 1.1 הפטרון האפשרי: clearfix.

/\* clearfix גורם להורה "להכיר" בגובה של ילדים עםfloat , מבלי לשנות את מבנה ה־HTML \*/

.clearfix::after {

content: ""; /\* Creates an invisible pseudo-element \*/

display: block; /\* Makes it behave like a block element \*/

clear: both; /\* Forces it to clear all floated siblings above (left & right) \*/

}

<div class="container clearfix">

<div style="float: left;">Box 1</div>

<div style="float: right;">Box 2</div>

</div>

## 2. למה ומתי גובה ה־container מושפע מה־margin של הילדים.

צור div container ובתוכו element p עם 20 מילים. כאשר נוסיף border ל־container הגובה שלו יגדל. מה אפשר להסיק מזה ביחס למה שלמדנו בהרצאה 5 שגובה ה־container לא מושפע מה־margin של הילדים? בונוס: תן הסבר לתופעה ([questions](https://stackoverflow.com/questions/27829250/why-doesnt-a-childs-margin-affect-a-parents-height)).

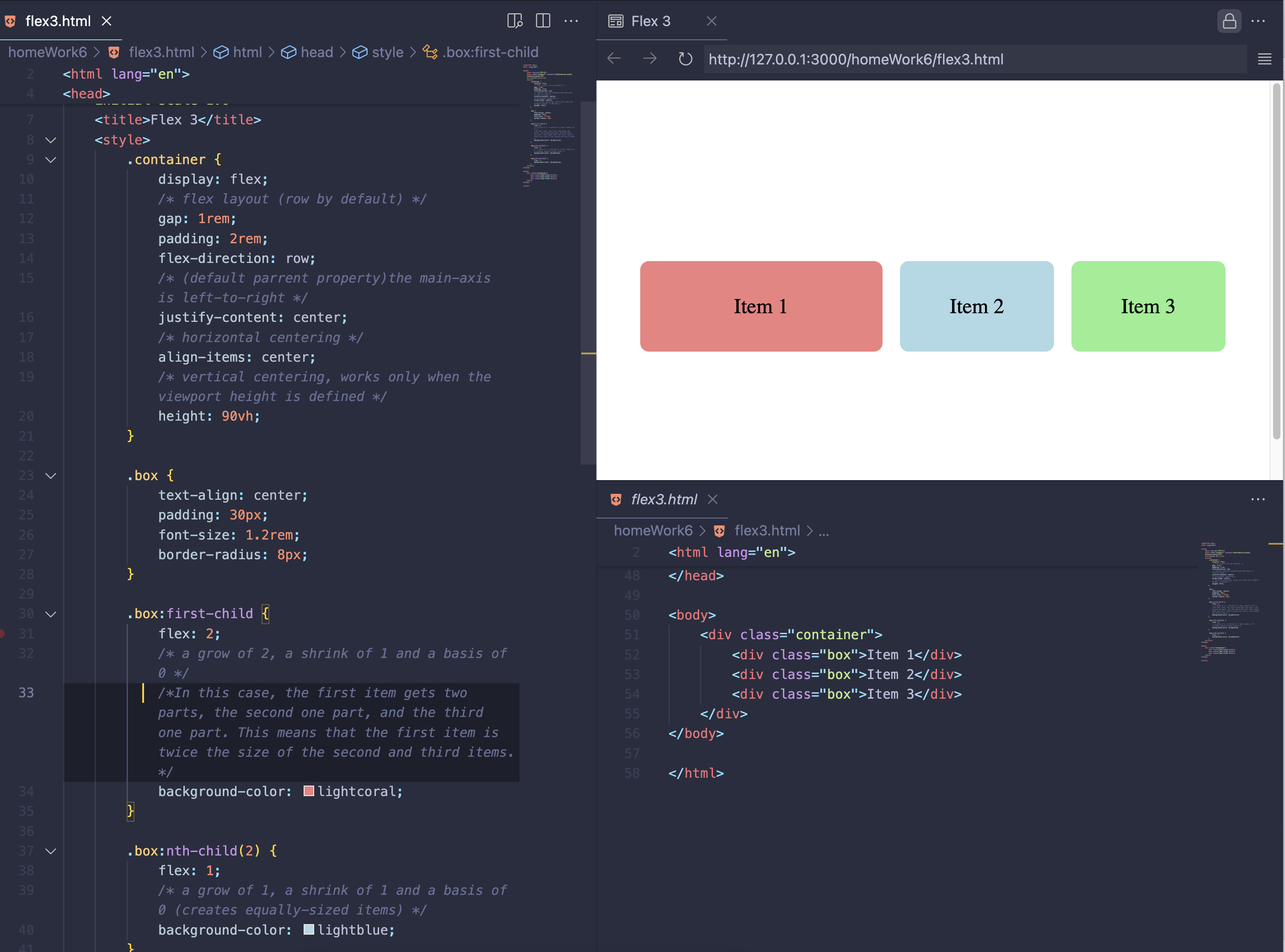
**תשובה**: למה הגובה של ה־ div משתנה כשמוסיפים לו border? הסיבה היא התנהגות מיוחדת שנקראת   
**קריסת מרווחים.(margin collapse)**

|  |  |
| --- | --- |
| ✅ עם border | ❌ בלי border |
| ברגע שמוסיפים borderאו אפילו padding הדפדפן מונע מה־ margin של הילד לברוח החוצה.  עכשיו ה־margin נלקח בחשבון בתוך ההורה.  התוצאה: הגובה של ההורה גדל בהתאם ל- margin . | אם יש רק ילד אחד עםmargin עליון/תחתון (ל-<p> margin היא תכונה דיפולטיוית) וההורה לא מכיל תוכן נוסף, padding או border  אז ה־margin ״בורח החוצה״ מההורה (קורס).  **התוצאה**: הגובה של ההורה לא משתנה, כאילו ה־margin לא קיים. |
|  |  |

## 3. פריסות רספונסיביות ב-row באמצעות Flexbox

צור flex ובתוכו שלושהdivs , במידות רספונסיביות (מותאמות לגודל המסך) ובצבעי רקע כבחירתך. סדר אותם בשורה ושיוצבו במרכז העמוד (מלמעלה ומהצדדים). דאג לכך שהגודל של הבן הראשון יהיה גדול יותר משאר הבנים ובכיוון ה- flex-direction, לשם כך אין להשתמש ב- height ו־width.

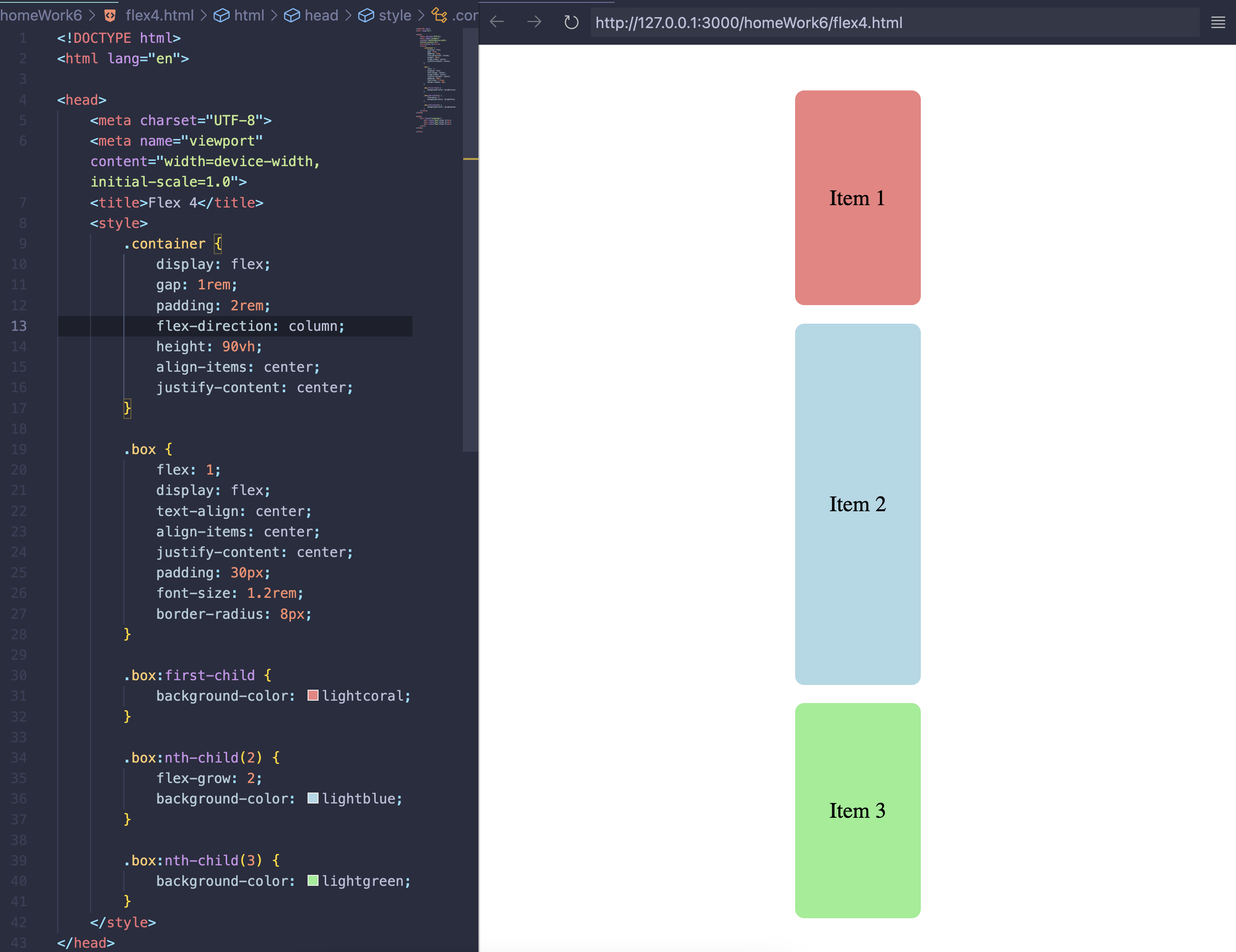
|  |  |  |  |
| --- | --- | --- | --- |
| Property | שייך ל... | הערות | משמעות ושימוש |
| display: flex | .container | חובה להציב על הקונטיינר | מפעיל מצב Flex ומאפשר סידור גמיש של הילדים |
| flex-direction: row | .container | ברירת מחדל | מסדר את הילדים בשורה, משמאל לימין או הפוך |
| justify-content: center | .container | צריך רוחב עודף בקונטיינר | ממרכז את הילדים לאורך הציר הראשי main  (אופקי במקרה של row) |
| align-items: center | .container | אם לא מציביםheight על הקונטיינרalign-items - לא ייראה בפועל. | ממרכז את הילדים על הציר המשני cross  (אנכי במקרה של row) |
| flex: 2 | .item | flex: X הוא קיצור ל flex-grow: X; flex-shrink: 1; flex-basis: 0; | גורם לילד לתפוס פי 2 גודל בהשוואה לאחרים  (flex-grow) |
| flex: 1 | .item | הקונטיינר חייב להיות flex | מגדיר גידול יחסי שווה בין שאר הילדים |



## 4. פריסות רספונסיביות ב-column

סדר אותם divs בעמודה. דאג לכך שהבן השני יגדל פי שניים מאחיו עם הגדלת גודל המסך.

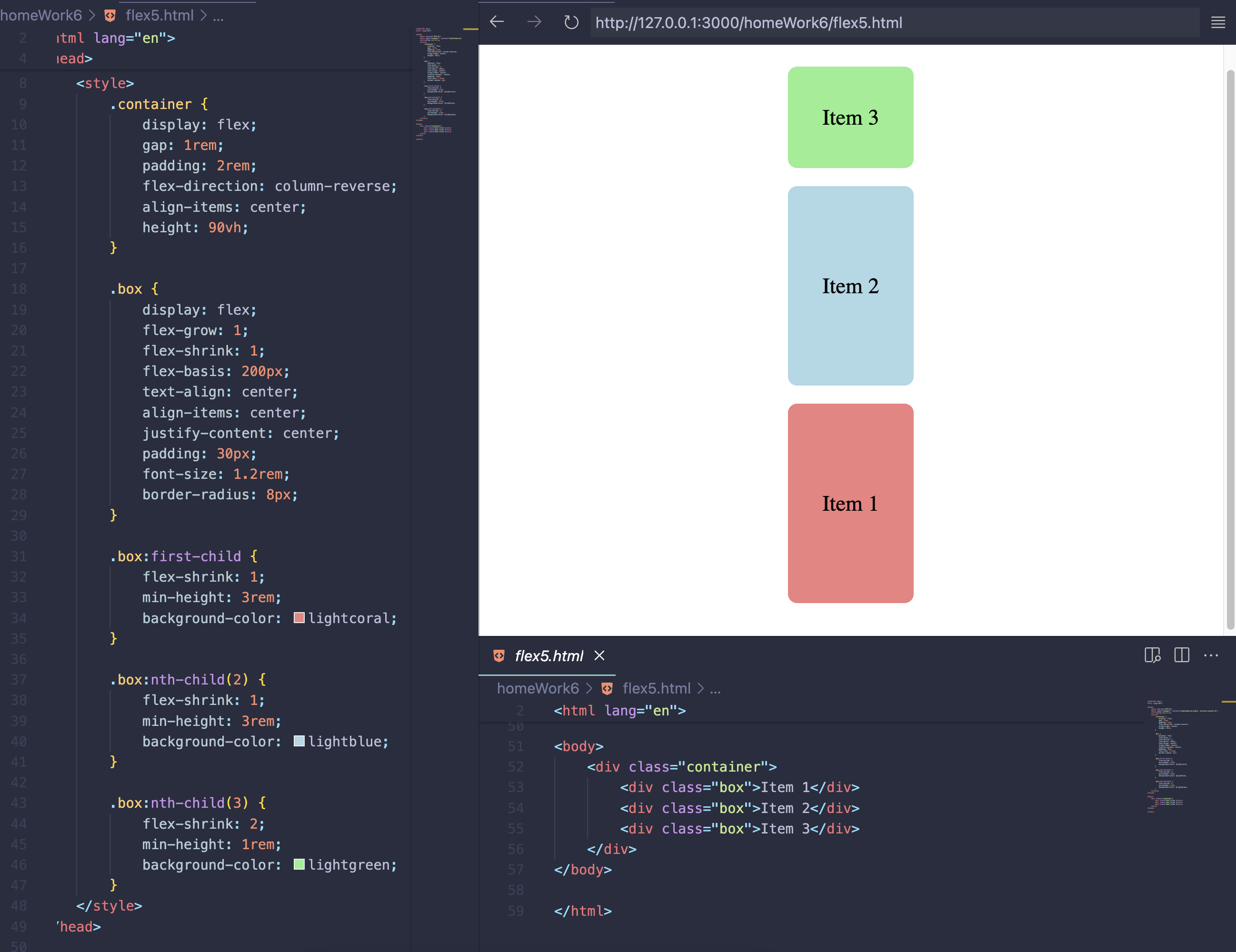
|  |  |  |  |
| --- | --- | --- | --- |
| Property | שייך ל... | תנאים להפעלה | מה זה עושה? |
| flex-direction: column | אלמנט אב (container) | יש להפעיל display: flex על הקונטיינר | מסדר את הילדים בעמודה (מלמעלה למטה),  והופך את הציר הראשי לאנכי |
| flex-grow: 2 | אלמנט בן (child) | הקונטיינר חייב להיות Flexbox (display: flex) | גורם לילד לתפוס פי 2 מקום יחסית לאחרים לאורך הציר הראשי (שכעת הוא אנכי בגלל column) |



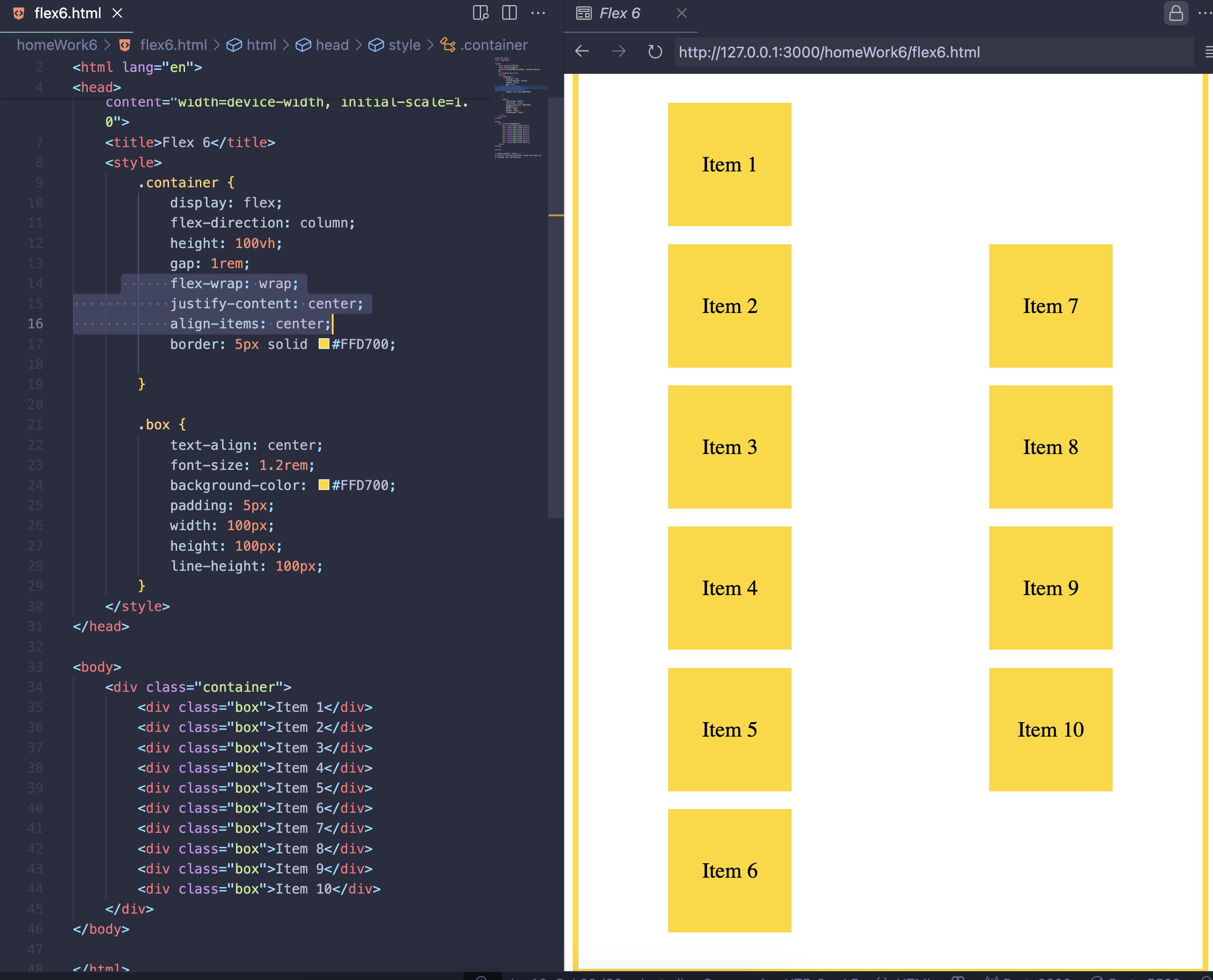
## 5. סידור בעמודה הפוכה + הקטנת גובה עם flex-shrink.

כנ"ל והפוך את הסדר שלהם. דאג לכך שהבן השלישי יקטן פי שניים ביחס לאחיו עם הקטנת גודל המסך.

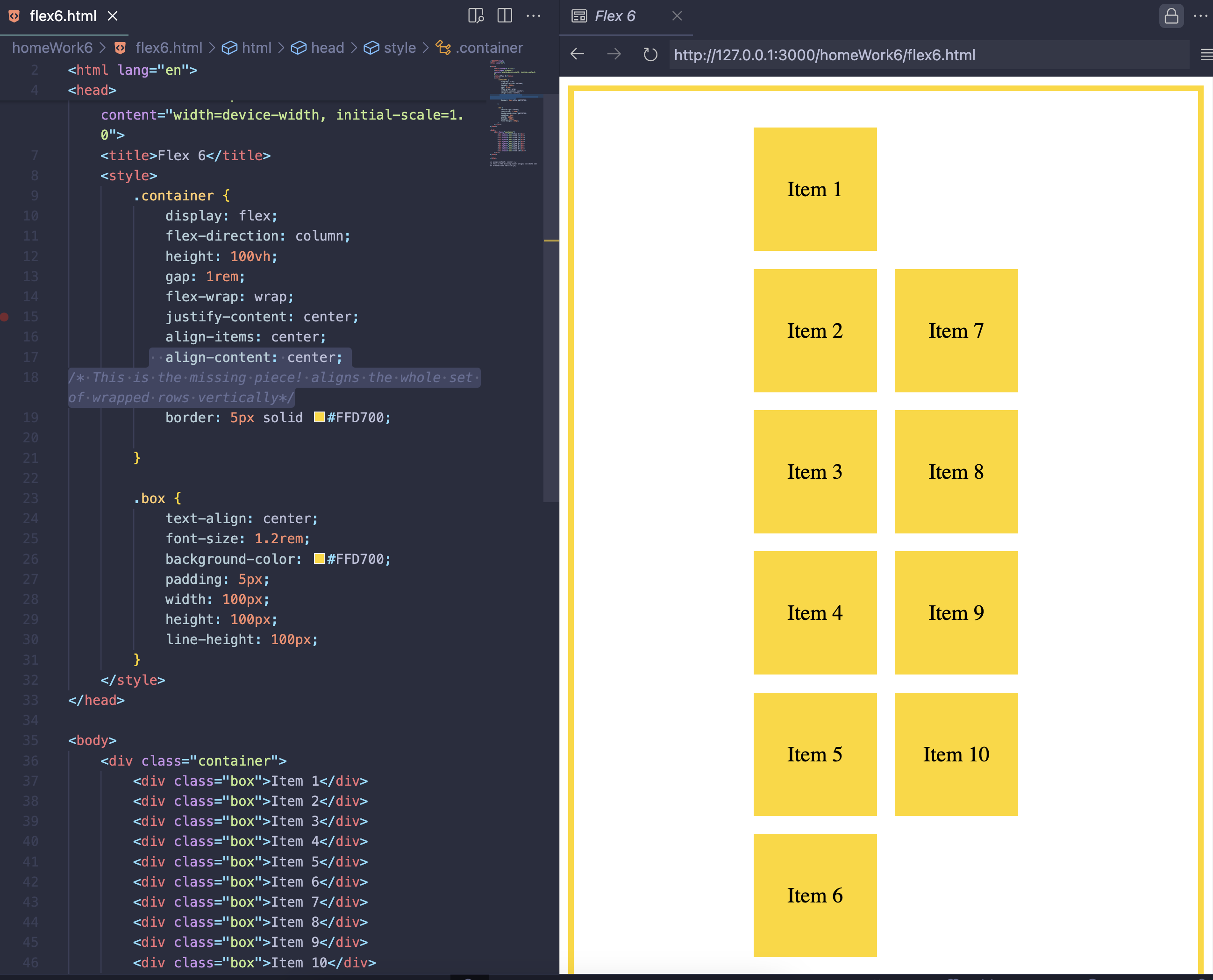
|  |  |  |
| --- | --- | --- |
| Property | שייך ל... | מה זה עושה בפועל? |
| flex-direction: column-reverse | .container | מסדר את הילדים בעמודה מהסוף להתחלה (מלמטה למעלה) |
| flex-grow: 1 | .item | מאפשר לכל ילד לגדול באופן שווה כשיש מקום |
| flex-basis: 200px | .item | קובע גובה התחלתי אחיד של 200px לכל תיבה |
| flex-shrink: 1 | בן רגיל | כל ילד יקבל גובה מוקטן ביחס שווה |
| flex-shrink: 2 | בן שלישי | גורם לבן השלישי לקטון פי 2 מהר יותר מהשאר |
| min-height: 1rem | בן שלישי | מגדיר את הגובה המינימלי האפשרי לתיבה |



## 6. צור flex ובתוכו עשרהdivs , כך שיסודרו באופן הבא:



### 6.1 מנו הסטייל שיגרום לסידור להיות כך:



7. ב-grid, מה יותר נכון:70% 30% grid-template-column: או 70fr, 30fr grid-template-column:?

**ברוב המקרים עדיף להשתמש ב־fr** כי היא גמישה יותר, מתנהגת חכם לפי תוכן, וקלה לתחזוקה.

ההבדל המרכזי:

* = **%** מתאים כאשר צריך שהעמודות יתפסו חלק יחסי **מהקונטיינר כולו**. (70% אומר שהעמודה הראשונה תמיד תהיה 70% מרוחב הקונטיינר, בלי קשר לתוכן).
* = **fr (**fraction**)** מתאים כשרוצים חלוקה יחסית **בין עמודות**, בלי קשר לרוחב הקונטיינר. (70fr קובע שהעמודה הראשונה תהיה 70 מתוך 100 חלקים של השטח הפנוי).

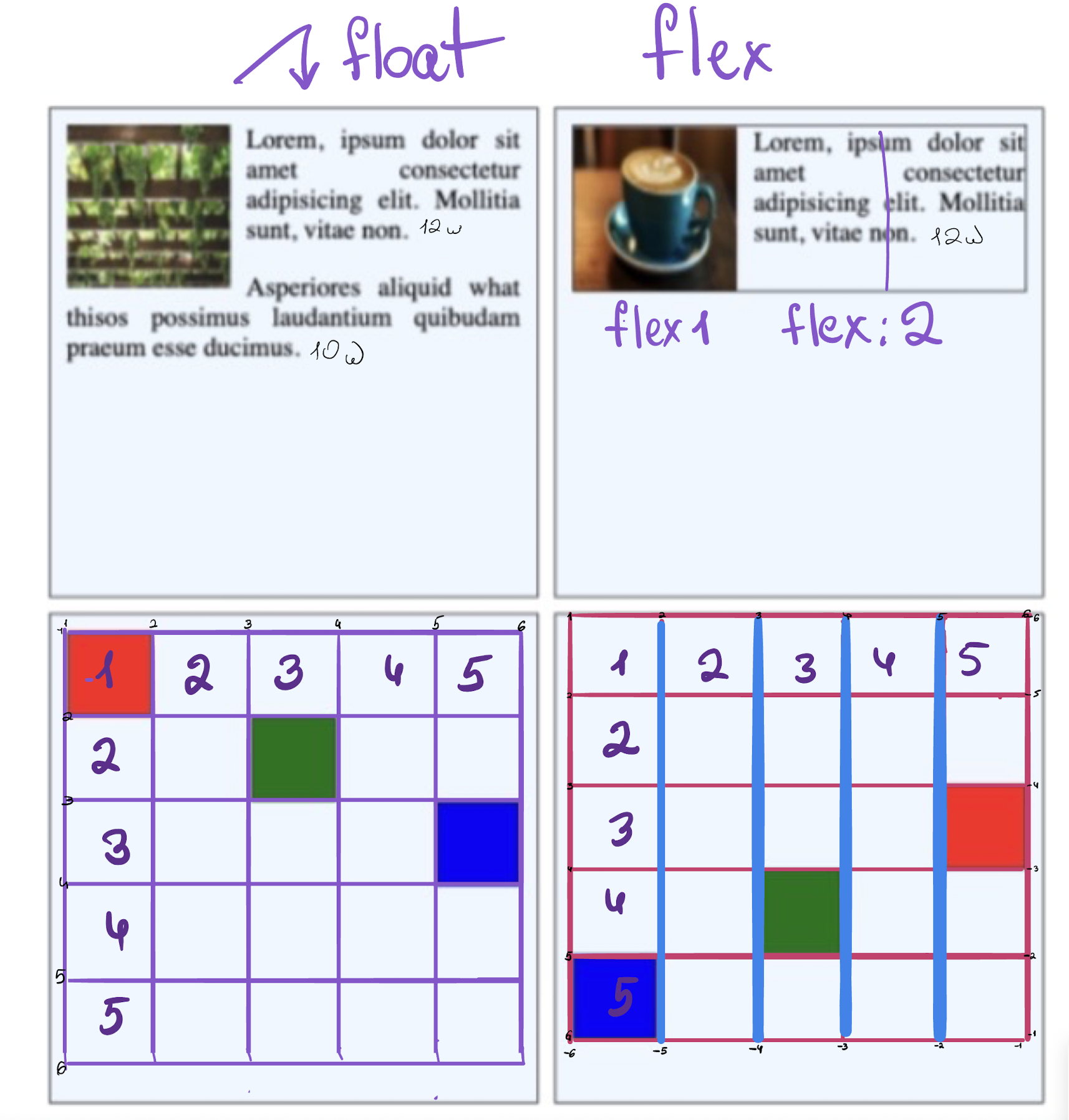
## 8. ב- grid מה תפקודים של align-items, justify-items, align-content, justify-content?

= items מיישר **פריטים** בתוך כל תא.

= content מיישר **את כל מערכת הגריד** בתוך הקונטיינר.

|  |  |  |  |
| --- | --- | --- | --- |
| פרופרטי | על מה הוא משפיע | כיוון | תיאור |
| **justify-items** | על **כל פריט** בגריד | אופקי (X) | מיישר את תוכן התאים האופקית (כגון שמאלה, ימינה או באמצע). לדוג׳: start, center, end, stretch. |
| **align-items** | על **כל פריט** בגריד | אנכי (Y) | מיישר את תוכן התאים אנכית בתוך השורה. לדוג׳: start, center, end, stretch. |
| **justify-content** | על **הרשת כולה** (grid) | אופקי (X) | מיישר את **הרשת כולה** בתוך הקונטיינר אם יש מקום עודף. לדוג׳: start, center, space-between, space-around. |
| **align-content** | על **הרשת כולה** (grid) | אנכי (Y) | מיישר את **הרשת כולה** אנכית בתוך הקונטיינר כאשר יש גובה עודף. לדוג׳: center, start, stretch, space-evenly. |

## 9. צור את האתר הבא: על התצוגה להיות במרכז העמוד מהצדדים ובסוף העמוד מלמלה. יש לכלול float, flex, grid.

1. 2⁰ = 1, 2¹ = 2, 2² = 4, 2³ = 8, 2⁴ = 16, 2⁵ = 32, 2⁶ = 64, 2⁷ = 128, 2⁸ = 256, 2⁹ = 512, 2¹⁰ = 1024 [↑](#footnote-ref-1)
2. ### ✅ ****הסבר** **לפי** **חלקים:****

   #### <meta name="ר

   [↑](#footnote-ref-2)
3. HEX: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10=A, 11=B, 12=C, 13=D, 14=E, 15=F [↑](#footnote-ref-3)
4. DRY = **Don’t Repeat Yourself**  
   The goal: **Write your style rules once** and **reuse** them instead of copying the same code multiple times. [↑](#footnote-ref-4)
5. כדי ש-position: absolute יעבוד **בתוך אלמנט מסוים** הדפדפן שואל: "יחסית למה למקם את האלמנט הזה?"

   אם אין אף **אב** עם position: relative הדפדפן יתייחס לעמוד כולו (body)

   אבל אם נוסיף position: relative לאלמנט אב (למשל (.container אז האלמנט עם position: absolute ימוקם **בתוך אותו אב** — לפי הפינות שלו (למשל: ימני עליון בתוך (.container [↑](#footnote-ref-5)