wantsome
The friendly IT Academy

Saptamana 4

Partea 1

Programare Front-End

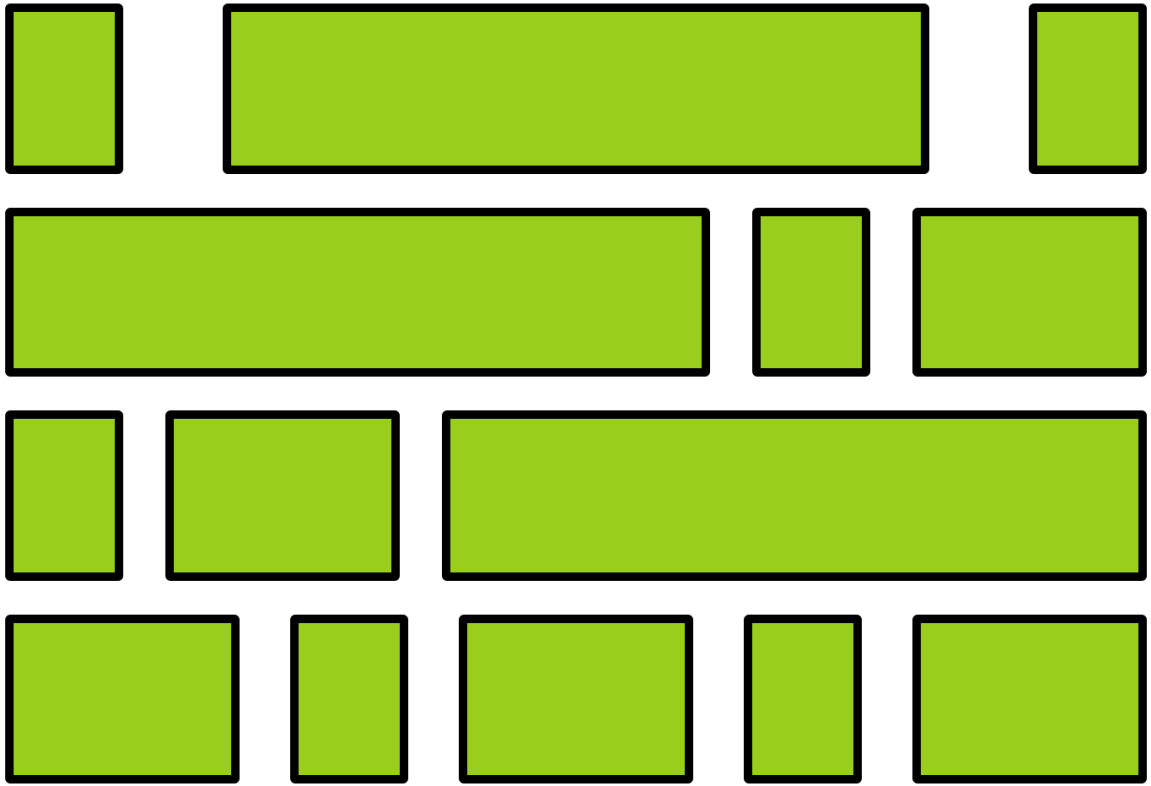
1. Complex display types and grid systems

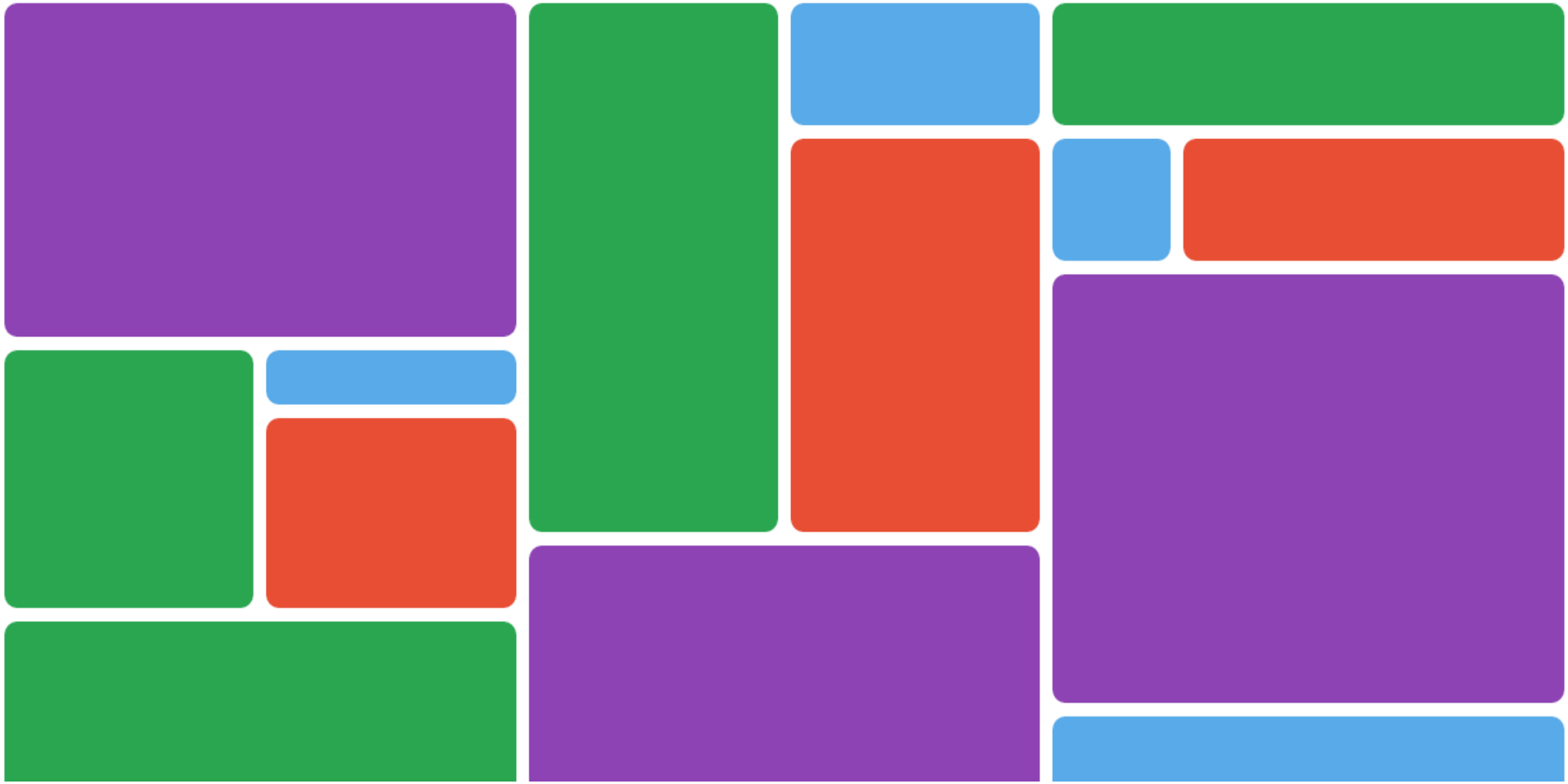
Problematica

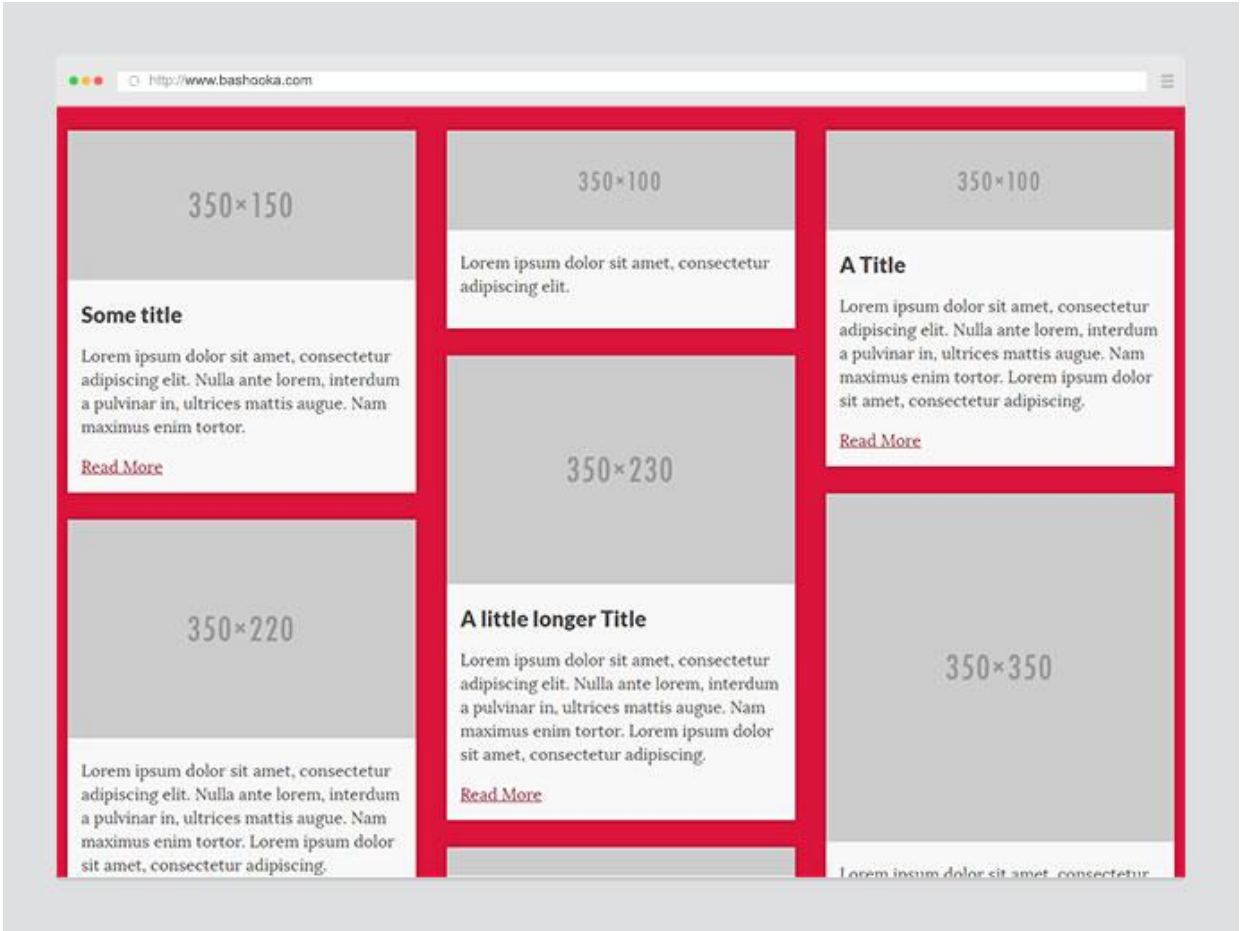
- In ultimii anii, aplicatiile web au devenit din ce in ce mai complexe, iar design-ul si layout-urile necesare au tinut pasul din acest punct devedere

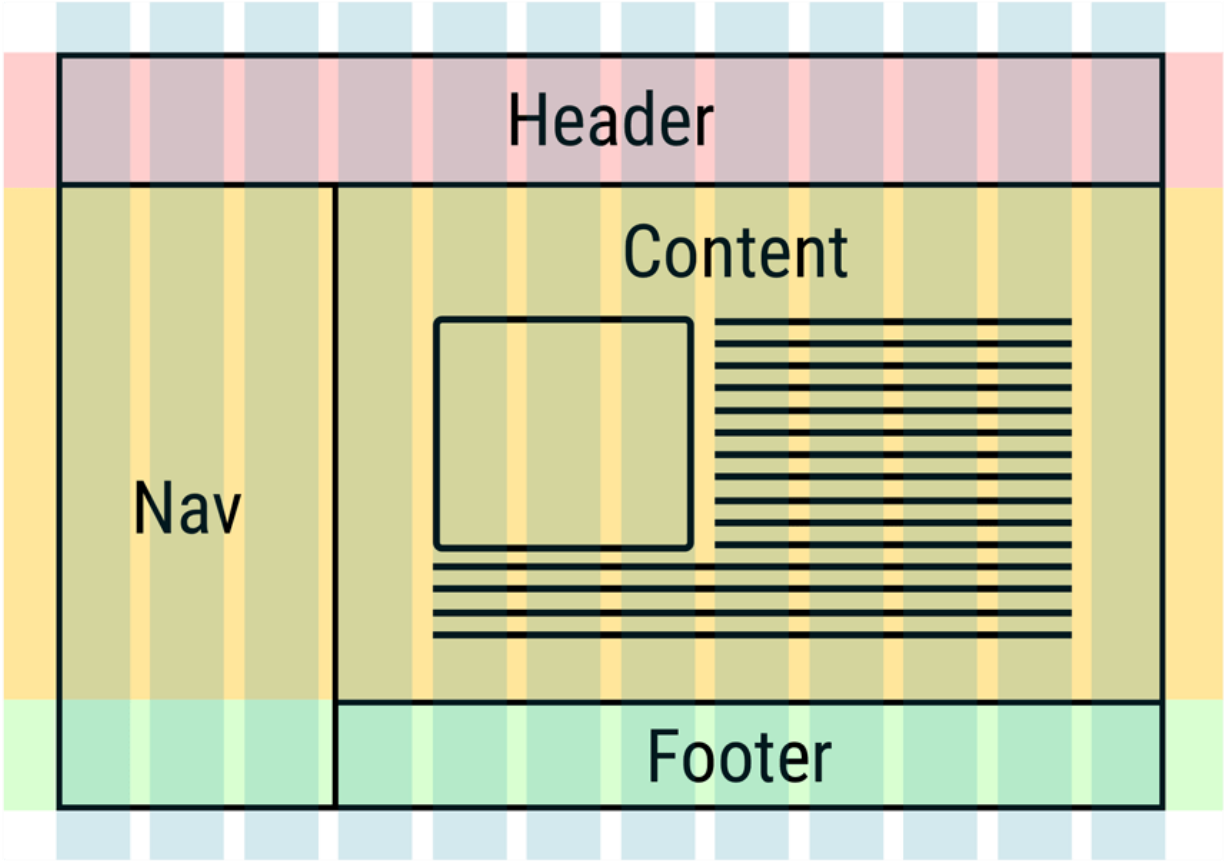
- Orice spatiu disponibil in cadrul unei interfete este foarte important si poate fi folosit cu folos

- Nevoia de metode cat mai simple pentru a specifica si dezvolta structuri ale unor interfete web care presupun afisarea a multiple elemente impartite pe multiple coloane si randuri cu continut adaptiv in functie de ecranul utilizatorului, a devenit vitala









Vertical...
Top tile

...tiles
Bottom tile

Middle tile
With an image

Wide tile
Aligned with the right tile

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin ornare magna eros, eu pellentesque tortor vestibulum ut. Maecenas non massa sem. Etiam finibus odio quis feugiat facilisis.

Tall tile
With even more content

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut. Morbi maximus, leo sit amet vehicula eleifend, nunc dui porta orci, quis semper odio felis ut quam.

Suspendisse varius ligula in molestie lacinia. Maecenas varius eget ligula a sagittis. Pellentesque interdum, nisl nec interdum maximus, augue diam porttitor lorem, et sollicitudin felis neque sit amet erat. Maecenas imperdiet felis nisi, fringilla luctus felis hendrerit sit amet. Aenean vitae gravida diam, finibus dignissim turpis. Sed eget varius ligula, at volutpat tortor.

Integer sollicitudin, tortor a mattis commodo, velit urna rhoncus erat, vitae congue lectus dolor consequat libero. Donec leo ligula, maximus et pellentesque sed, gravida a metus. Cras ullamcorper a nunc ac porta. Aliquam ut aliquet lacus, quis faucibus libero. Quisque non semper leo.

Solutii ?

2.1 CSS Flexbox

Flexbox - *display: flex*

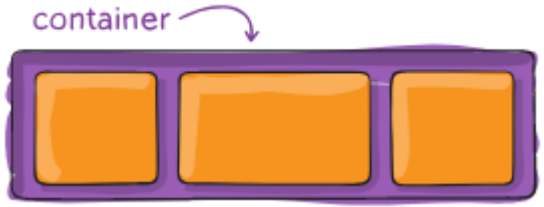
Modulul Flexbox Layout (Flexible Box) intentioneaza sa ofere o cale mult mai eficienta de a structura, alinia si distribui spatiul alocat elementelor dintr-un *container*, chiar si atunci cand dimensiunea lor este necunoscuta sau dinamica (de aici si denumirea de *"flex"*).

Ideea principala din spatele *flex layout* este sa ofere unui *container* abilitatea de a altera dimensiunile elementelor pe care le contine in asa fel incat sa distribuie cel mai eficient spatiul disponibil; Un *container* de tip flex este capabil sa largeasca sau sa ingusteze elementele pentru a ocupa spatiul sau a nu depasi spatiul disponibil in interiorul sau.

Spre deosebire de obisnuitele layout-uri de tip block destinate distribuirii elementelor pe verticala sau inline pentru orizontala, flex nu depinde de directie. Cu toate ca acestea pot fi foarte utile pentru structurarea paginilor web, atunci cand vine vorba de situatii complexe sunt foarte limitate (in special cand vine vorba despre schimbarea orientarii unui device, *resize, stretching, shrinking,* etc.).

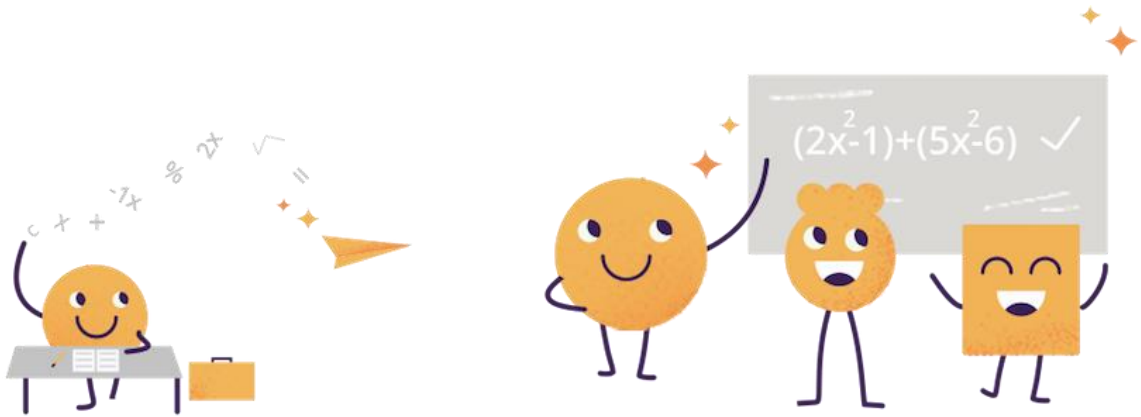
Flexbox - proprietati specifice

https://css-tricks.com/snippets/css/a-guide-to-flexbox/



PRACTICE: **Flexbox**

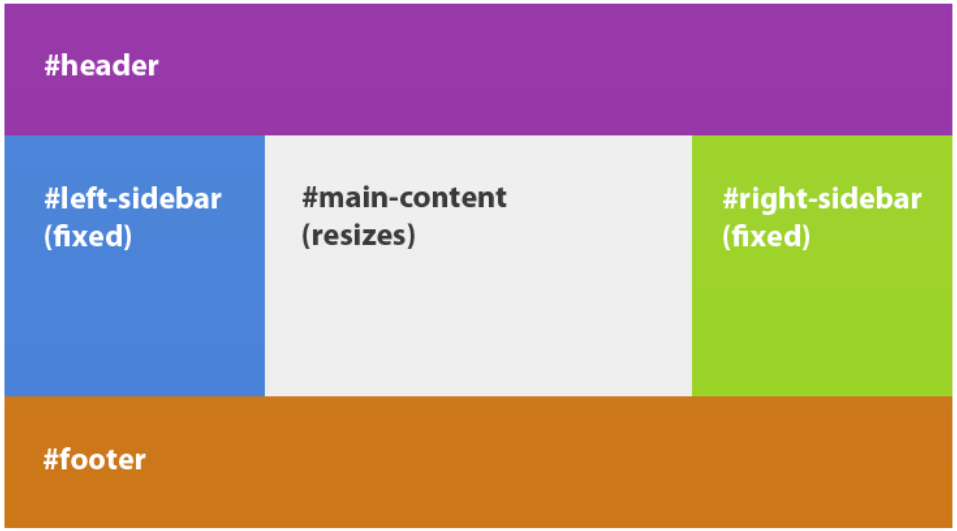
https://flexboxfroggy.com/

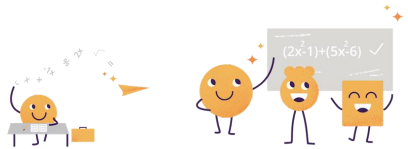


PRACTICE: Flexbox

Cerinte:

1. *Implementati layout-ul urmator folosind **flexbox**:*





wantsome
The friendly IT Academy

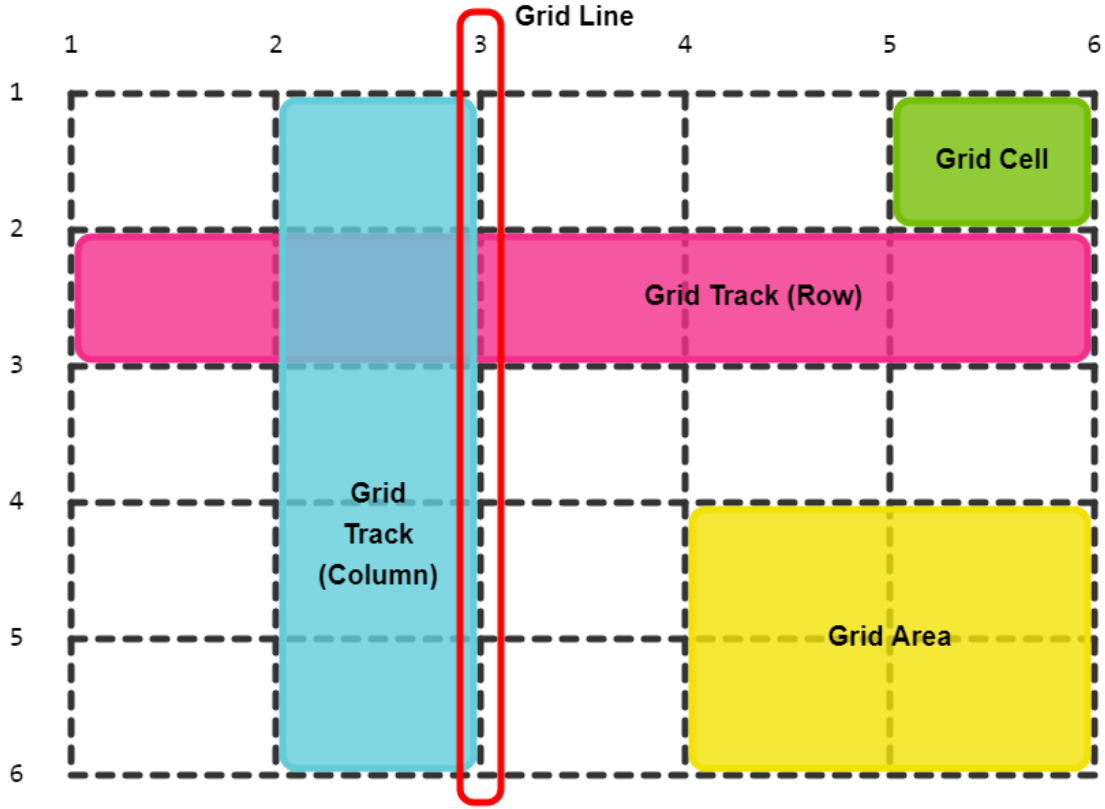
2.2 CSS Grid

Grid - *display: grid*

CSS Grid Layout, sau Grid, asa cum este cel mai numit, este un sistem de structurare a layout-urilor a carui intentie este sa usureze pe deplin modalitatea de implementare a interfetelor gandite sub forma de grila, fara a fi nevoiti sa apelam la tabele (care nici nu sunt *responsive*), *floating*, tehnici foarte complicate, complexe.

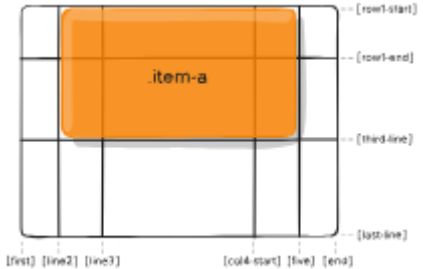
Flexbox poate ajuta foarte mult in acest sens, insa este uni-dimensional spre deosebire de Grid care este bi-dimensional - lucreaza atat pe coloane cat si randuri in acelasi timp - (pot lucra foarte bine impreuna).

Implementarea de interfete bazate pe un *grid* a fost o foarte mare bataie de cap pana la implementarea acestui modul.

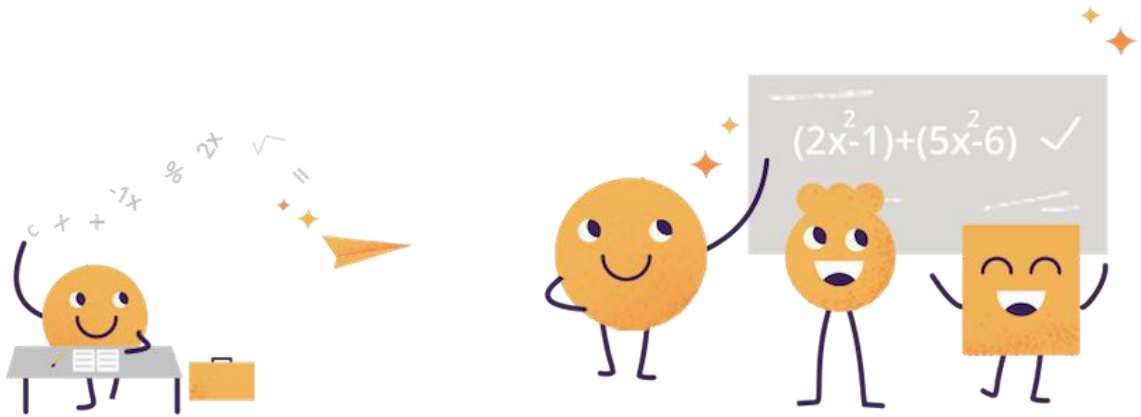


Grid - proprietati specifice

https://css-tricks.com/snippets/css/complete-guide-grid/



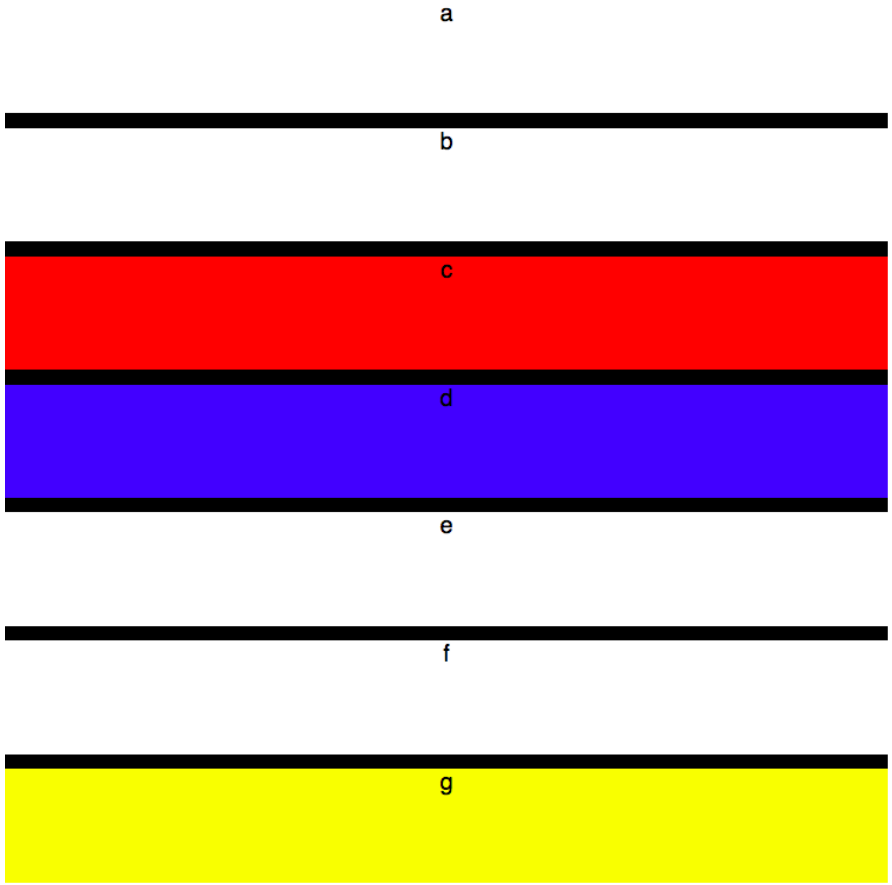
PRACTICE: **Grid**

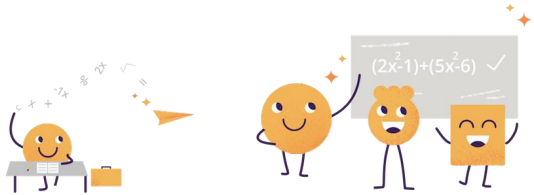


PRACTICE: Grid

Cerinte:

1. *Implementati layout-ul urmator folosind **grid**:*
- *bordurile negre sunt de fapt **grid-gap***

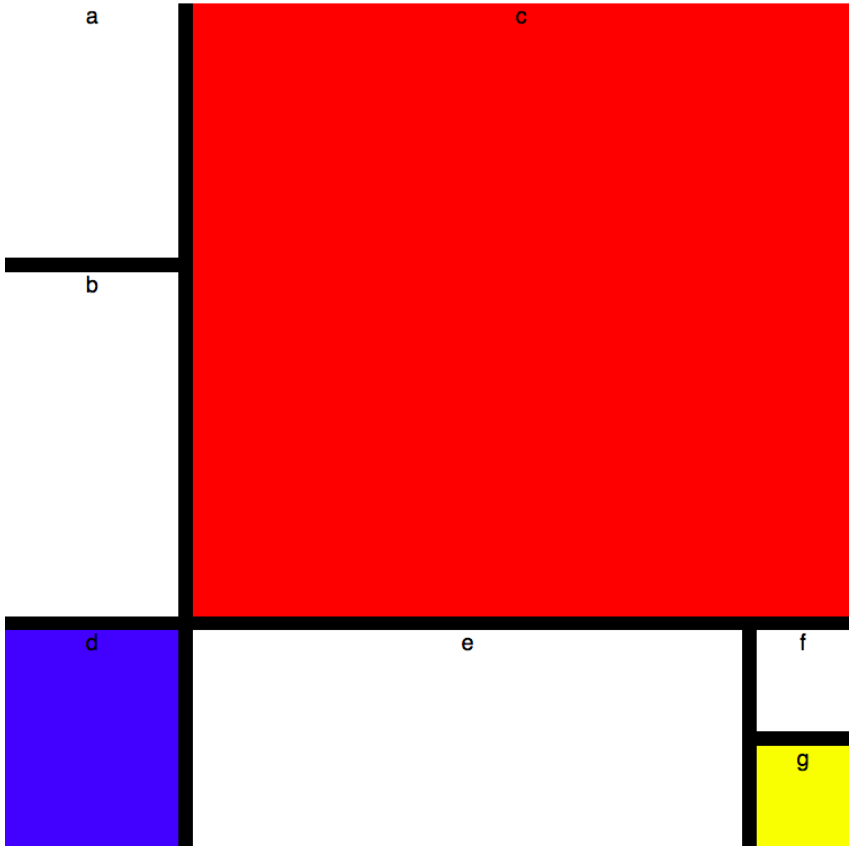


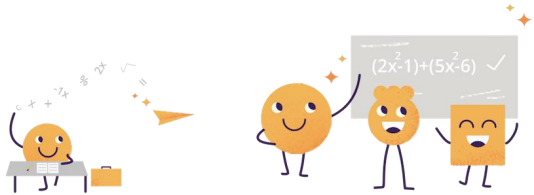


PRACTICE: Grid

Cerinte:

1. *Modificati css-ul pentru layout-ul anterior astfel incat sa arate ca in imaginea alaturata. Grid-ul are o dimensiune de 10 x 10, fara dimensiuni fixe.*

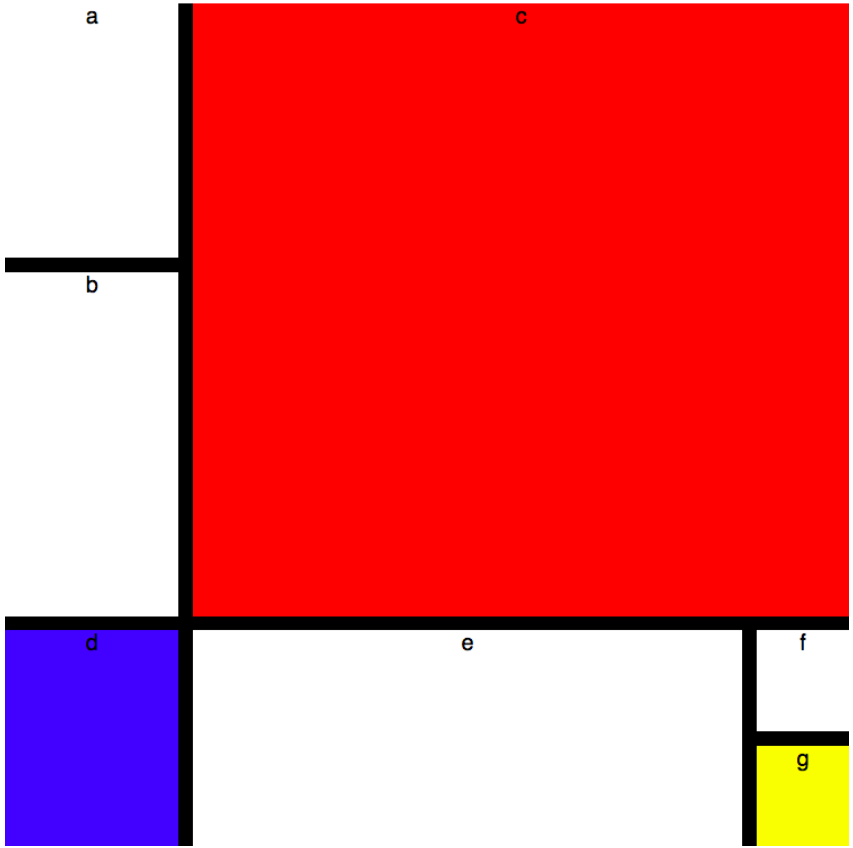


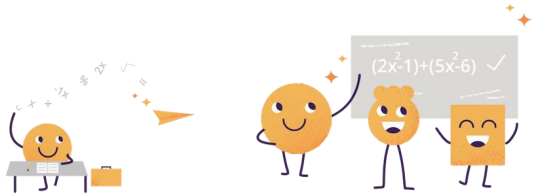


PRACTICE: Grid

Solutie:

https://codesandbox.io/s/1r80vjl81j





Flexbox vs Grid, Flexbox + Grid

- Ambele module ofera functionalitati foarte puternice si care ne usureaza structurarea interfetelor complexe ale aplicatiilor web

- In general, **Flexbox** este folosit pentru interfete uni-dimensionale care nu presupun un numar foarte mare de elemente (*small-scale*)

- **Grid** ne ajuta in momentul in care complexitatea structurii unei interfete creste (*large-scale*) si apare nevoia de a distribui elementele bi-dimensional

- De cele mai multe ori, cele doua module sunt folosite impreuna datorita particularitatilor lor

2. Responsiveness - Cross-device compatibility

Problematica

- Utilizatorii aplicatiilor web folosesc o foarte mare varietate de dispozitive mobile sau desktop pentru a naviga pe internet, care dispun de diferite platforme, dimensiuni ale ecranului, orientare

- Dezvoltarea de aplicatii de sine statatoare pentru fiecare caz in parte ar fi un impediment major

- O aplicatie web care functioneaza doar pe desktop poate pierde un numar foarte mare si important de utilizatori

Ce solutie avem ?

2.1 RWD – Responsive Web Design

Responsive **W**eb **D**esign

Responsive Web Design este metoda care presupune ca design-ul si dezvoltarea unei interfete web sa fie adaptive in functie de comportamentul utilizatorului raportat la marimea ecranului, platforma si orientare.

Practica in sine consista in folosirea unui mix de *grids*, *layouts*, imagini si o utilizare inteligenta unei tehnici *CSS* numita *media queries*. Daca utilizatorul foloseste iPad-ul in loc de laptop, site-ul in cauza trebuie sa se adapteze automat in functie de noua rezolutie, dimensiunea potrivita a imaginilor si capabilitatile device-ului (este posibil ca unele functionalitati implementate pentru browsere-le de pe desktop sa nu functioneze pentru cele mobile). Deasemenea, trebuie luate in considerare si setarile utilizatorului de pe acel dispozitiv.

2.1 CSS Media Queries

CSS Media Queries

- modul care permite definirea unor reguli pentru aplicarea anumitor proprietati CSS doar daca o anumita conditie este indeplinita - *@media*
- conditiile pot include: tipul de device (all, print, screen, speech) si diferite caracteristici, parametri (precum rezolutia, marimea ecranului, marimea viewport-ului, etc.) insotite de diferiti operatori (and, not, only)
- multiple reguli pot fi combinate utilizand “,”

Exemple:

```
@media screen and (min-width: 30em) and (orientation: landscape) { ... }

@media (min-height: 680px), screen and (orientation: portrait) { ... }
```

Referinte: positioning explained

Media Queries - conditii, parametri, operatori

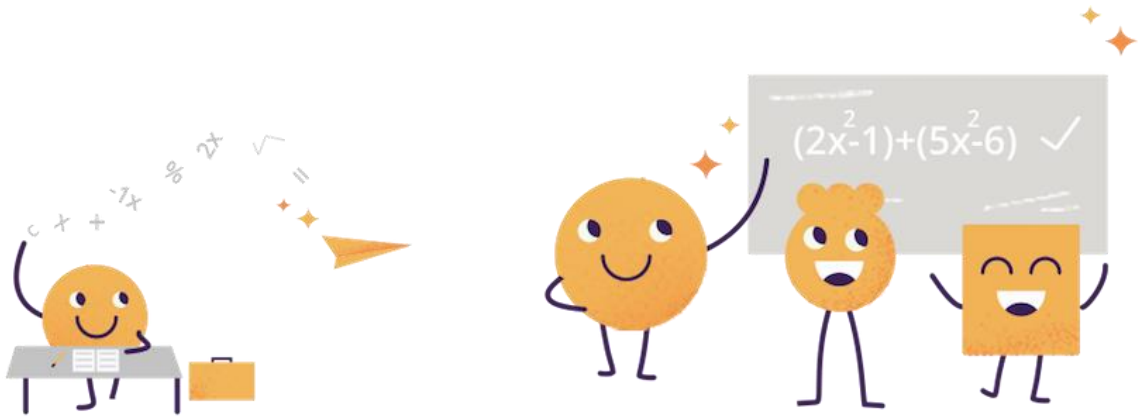
Spec

https://developer.mozilla.org/en-US/docs/Web/CSS/Media_Queries/Using_media_queries

Collection of Media Queries for Standard Devices

https://css-tricks.com/snippets/css/media-queries-for-standard-devices/

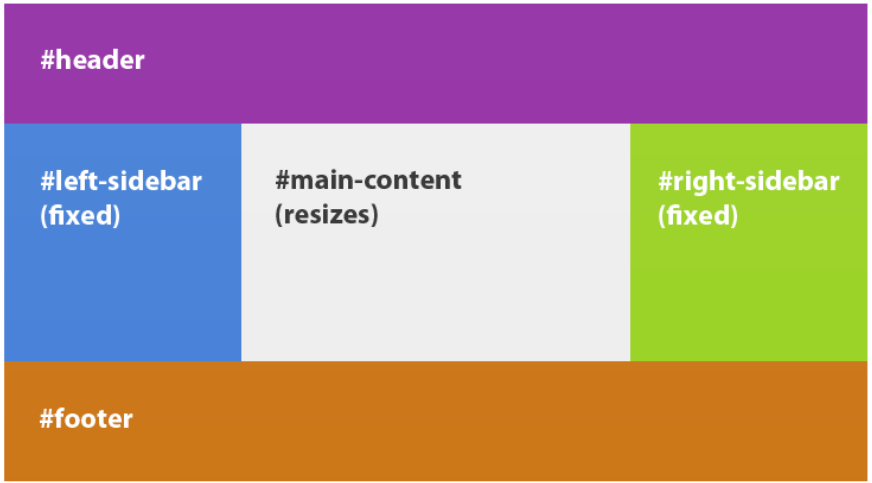
PRACTICE: **CSS Media Queries**

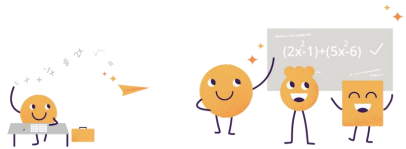


PRACTICE: Media Queries

Cerinte:

1. *Pentru layout-ul implementat la sectiunea **flexbox**, creati varianta mobile (phone to tablet); Culorile trebuie sa fie diferite pentru varianta mobile*





wantsome
The friendly IT Academy

HOMEWORK:
CSS: Backgrounds, Flex, Grid, Media Queries

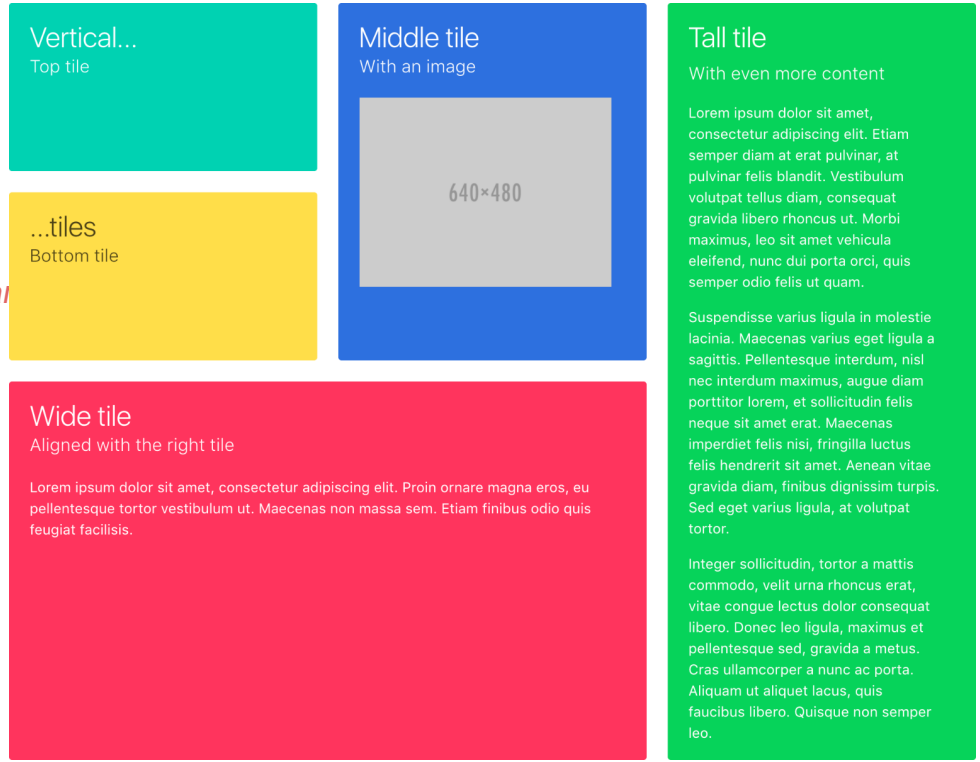
Cerinte:

1. *Creati o pagina HTML care sa aiba o structura valida. Background-ul paginii trebuie sa aiba o culoare deschisa, la alegere, specificata folosind notatie **hexadecimala** (ex.: #3399ff). Pagina trebuie sa includa un header si un footer cu height de 70px si background de culoare verde specificata folosind o notatie **RGB**. Footer-ul trebuie sa fie asezat in partea de jos a paginii. Adaugati o sectiune main care sa includa 3 blocuri de dimensiune 250x250 si care sa aiba ca background-uri 3 imagini diferite. Background-ul sectiunii main trebuie sa aiba aceeasi culoare ca header-ul si footer-ul dar cu o transparenta redusa cu 30% (**rgba**). Imaginile trebuie sa se ajusteze conform dimensiunilor blocurilor. Aliniati cele trei blocuri inline si distribuiti in mod egal spatiul gol dintre ele (primul bloc nu trebuie sa inceapa chiar din stanga, trebuie sa existe un spatiu inainte). Orice aliniere se va face folosind **flexbox**.*

HOMEWORK:
CSS: Backgrounds, Flex, Grid, Media Queries

Cerinte:

2. *Implementati urmatorul design. Pastrati spatierea, marginile, culorile, sa si folositi **Grid**.*



HOMEWORK:
CSS: Backgrounds, Flex, Grid, Media Queries

Cerinte:

3. *Creati o pagina HTML valida. Folositi https://uigradients.com/ pentru a alege un **gradient**. Aplicati-l ca background paginii nou create. In cadrul paginii, recreati structura din imaginea alaturata. Pagina trebuie sa se vada corespunzator pe orice tip de dispozitiv.*