CSS3

CSS3 – Médiatípusok használata

- A stíluslapok csatolási módjáról szóló bevezetőben már megismerkedtünk a @media szabállyal, amelynek segítségével a szabályok egyszerűen csoportosíthatók médiatípusok

Használható médiatípusok:

- *All*: összes eszköz
- *Speech*: beszédszintetizátor
- *Braille*: braille-kijelzők számára
- *Embossed*: braille-nyomtató számára
- *Handheld*: kézi, mobil eszközök (kis képernyő)
- *Print*: nyomtatás
- *Projection*: kivetítő
- *Screen*: képernyők
- *Tty*: fix szélességű karakteres megjelenítők
- *Tv*: televízió jellegű eszközök

Media Types	Media Groups			
	continuous/paged	visual/audio/speech/tactile	grid/bitmap	interactive/static
braille	continuous	tactile	grid	both
embossed	paged	tactile	grid	static
handheld	both	visual, audio, speech	both	both
print	paged	visual	bitmap	static
projection	paged	visual	bitmap	interactive
screen	continuous	visual, audio	bitmap	both
speech	continuous	speech	N/A	both
tty	continuous	visual	grid	both
tv	both	visual, audio	bitmap	both

Médiafüggő stíluslap megadása:

- 1. módszer - *Cél média megadása @media vagy @import szabályokkal.*

```
@import url("egyenibetutipus.css") screen;
@media print { /* nyomtatásra vonatkozó def. */ }
```

- 2. módszer - A cél médium megadása a dokumentumleíró (HTML) nyelvben

```
<LINK rel="stylesheet" type="text/css" media="print, handheld" href="nyomt.css">
```

Média csoportok

- A média típusok különböző szempontok szerint csoportosíthatóak is, és a stíluslap megadásánál ezen médiacsoportok nevét is megadhatnánk. (Lásd táblázat - ezen médiacsoportok megnevezését láthatjuk.)

CSS3 – Reszponzív arculat

A *Responsive Web Design (**RWD**)* egy olyan tervezési módszert jelent, amelynek célja, hogy optimális megjelenést biztosítson (egyszerű olvashatóság, könnyű navigálhatóság) a különböző eszközökön és platformokon (mobil eszközöktől a nagyobb felbontású monitorokig).

- *Viewport megadása*:

  A mobil eszközök böngészőprogramjai az oldalakat *virtuális ablakokban* jelenítik meg (ezt nevezzük *viewportnak*), amely általában szélesebb, mint a képernyő, és ezen területen a felhasználók több irányban barangolhatnak, illetve egyes területekre ráközelíthetnek, illetve eltávolodhatnak.

  Ha ezen *viewport* tulajdonságait meg szeretnénk adni a jobb felhasználói élmény biztosításának érdekében, egy <meta> taget kell használnunk, amelyet az oldal <head> részében kell elhelyeznünk.

  A *viewportnak* számos tulajdonsága lehet, amelyet vesszővel elválasztva adhatunk meg.

```
<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1">
```

  - *width* - a viewport szélessége. Ez lehet egy adott érték (pl. 500) vagy speciális érték is (*device-width*), amellyel az aktuális eszköz kijelzőjének szélességét tudjuk megadni (normál nagyítási szintre vonatkozóan).
  - A *height* - viewport magassága - itt is használható a *device-height* érték is.
  - Az *initial-scale* paraméterrel állíthatjuk be azt az alapértelmezett nagyítási szintet, ami az oldal első betöltésekor lesz érvényes.
  - A *maximum-scale* és *minimum-scale* paraméterrel a maximális és minimális nagyítási szint.

    Az egyes mobil eszközök eltérően viselkednek akkor, ha például álló helyzetből (portrait) fekvő helyzetbe (landscape) állítjuk a böngészőt. Például egy álló helyzetből fekvő helyzetbe váltás eredményezheti azt, hogy a böngésző megváltoztatja a nagyítási szintet, ahelyett, hogy úgy rendezné el az oldalt, mintha eleve fekvő helyzetbe töltődött volna be. Ezen nem kívánatos viselkedés miatt szokták a fejlesztők a maximum-scale paramétert 1-re állítani.
  - A *user-scalable* paraméterrel engedélyezhetjük (user-scalable="yes"), illetve letilthatjuk (user-scalable="no"), hogy a felhasználó nagyíthatja-e az alkalmazást vagy sem.

    pl.: a Google térkép esetén a user-scalable="no" beállítással találkozunk, mert ott hátrányos lenne, ha a térképre az eszköz nagyítás funkciójával közelítenénk rá, mert akkor a térkép pixelessé válna. Ehelyett maga az alkalmazás biztosítja nekünk a térképre történő kíváló minőségű nagyítást.

CSS3

CSS3 – Reszponzív arculat

- *Media Query használata:*
  Egy médialekérdezés egy média típus megadásból áll valamint nulla vagy több kifejezésből, amelyek lekérdezik az aktuális média egyes tulajdonságait. Egy médialekérdezés igazából egy logikai kifejezésnek tekinthető, ami igaz vagy hamis értékkel tér vissza. A lekérdezés igaz, ha a megadott média típus megegyezik az eszköz média típusával, és az összes kifejezés értéke igaz.

  Ez ahhoz szükséges hogy az adott médiához jól illeszkedő stíluslapot készíthessünk.

```
<link rel="stylesheet" media="screen and (color)" href="pelda.css">
```

vagy

```
@import url(color.css) screen and (color);
```

  Ezzel a kóddal a pelda.css stíluslapot színes képernyőn való megjelenítéshez linkeltük be. Az and operátorral több feltétel együttes meglétét kezeltük le.
  Ha az összes médiatípusra egyben szeretnénk hivatkozni rövidített módon:

```
@media all and (min-width:500px) {  }
@media (min-width:500px) {  }
```

  minden médiatípusra vonatkozik, amennyiben a megjelenítő minimális szélessége 500 képpont, az all érték viszont elhagyható

  - *Operátorok*
    A kifejezésekben különböző operátorokat használhatunk, ilyen a már példában szerepelt and (és) is. Most nézzük ezeket összegyűjtve:
    - *and* - logikai ÉS operátor →Pl.: *<link rel="stylesheet" media="speech and (min-device-width: 800px)" href="pelda.css">*
    - *not* - logikai NEM operátor → Pl.: *<link rel="stylesheet" media="not screen and (color)" href="pelda.css">*
    - *only* - arra használhatjuk, hogy a stíluslapot elrejtsük a régebbi böngészőprogramok elöl, mert azok rosszul értékelnék ki a kifejezést. → Pl.: *<link rel="stylesheet" media="only projection and (color)" href="pelda.css„>* az only kifejezés nélkül arégebbi böngésző a projection médiára állítaná be a stíluslapot, és figyelmen kívül hagyná az and (color) kifejezést.
    - *A vagy kapcsolatot is megadhatunk*, erre viszont a vessző szolgál, nem pedig a máshol megszokott or operátor.

      pl. *@media screen and (color), projection and (color) {}*

CSS3

CSS3 – Reszponzív arculat

- *Média tulajdonságok*
  - *aspect-ratio:* méretarány. A szélesség osztva a magassággal.
    - Értéke: tört (pl. 16/9)
    - Használható a min/max előtag: igen
    - Példa: <link rel="stylesheet" media="screen and (aspect-ratio: 4/3)" href="pelda.css">
  - *device-aspect-ratio*: méretarány. A device-width szélesség osztva a device-height magassággal.
    - Értéke: tört (pl. 16/9)
    - Használható a min/max előtag: igen
    - Példa: <link rel="stylesheet" media="screen and (device-aspect-ratio: 16/9)" href="pelda.css">
  - *color*: színmélység. A megjelenítő eszköz színmélységére jellemző szám (bitek száma színkomponensenként). Nulla esetén a kijelző nem színes.
    - Értéke: egész szám
    - Használható a min/max előtag: igen
    - Példa: <link rel="stylesheet" media="screen and (min-color: 1)" href="pelda.css">
  - *resolution*: A megjelenítő eszköz felbontása.
    - Értéke: felbontás (pl. 100dpi)
    - Használható a min/max előtag: igen
    - Példa: <link rel="stylesheet" media="screen and (min-resolution: 300dpi)" href="pelda.css">

CSS3

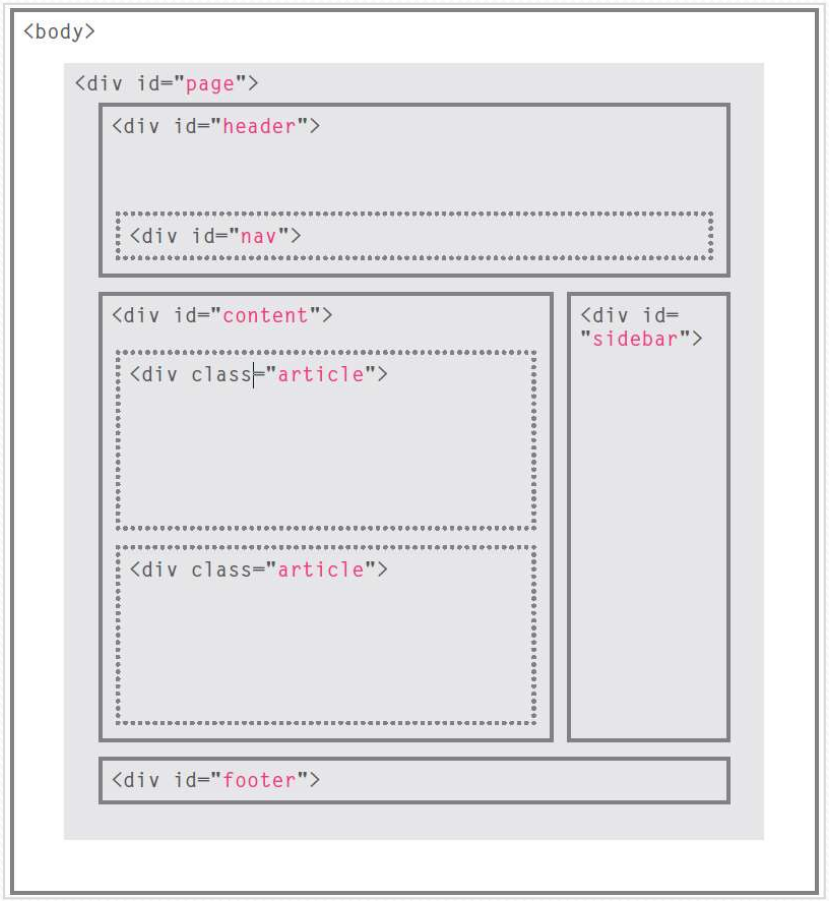
CSS3 – Reszponzív arculat

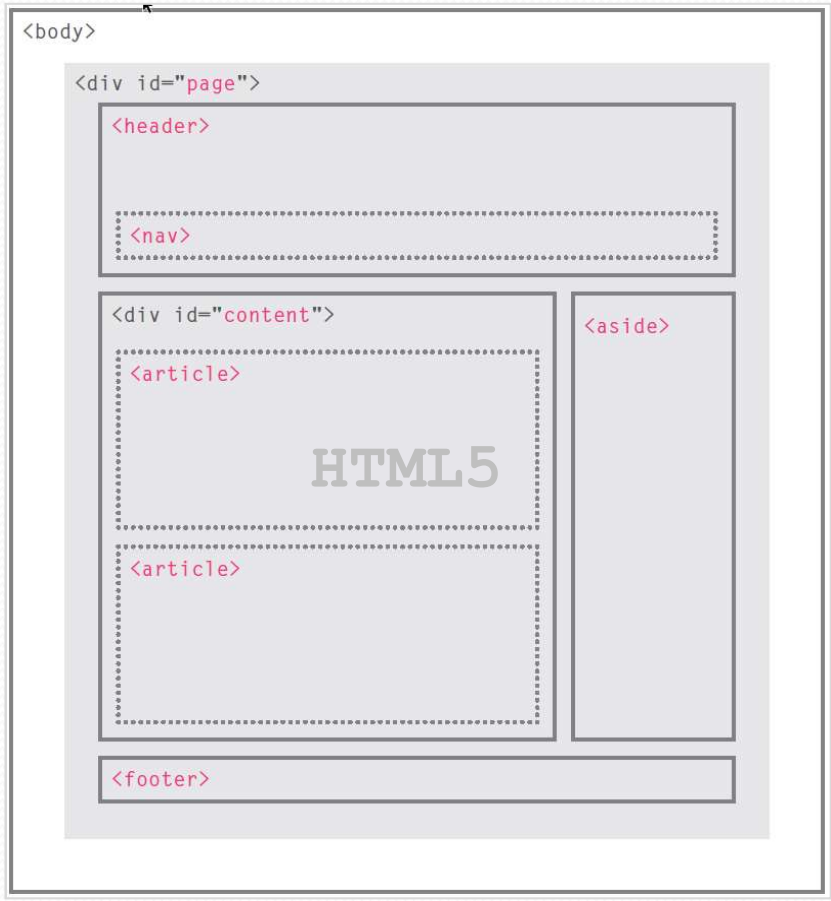
- *Média tulajdonságok*
  - *width: szélesség / height: magasság.* Folytonos média esetén a viewport szélességét / magasságát jelenti a gördítősávot (ha van) is beleértve. Lapozható média esetén ez az oldal doboz szélességét jelenti.
    - Értéke: hosszúság érték (pl. 300 px)
    - Használható a min/max előtag: igen
    - Példa: *<link rel="stylesheet" media="print and (min-width: 25cm)" href="nyomt.css>*
    - Példa: *<link rel="stylesheet" media="print and (min-height: 30cm)" href="nyomt.css">*
  - *device-width: szélesség / device-height: magasság.* Folyamatos média esetén a képernyő szélességét /magasságát jelenti, lapozható média esetén az oldallap szélességét/magasságát .
    - Értéke: hosszúság érték (pl. 300 px)
    - Használható a min/max előtag: igen
    - Példa: *<link rel="stylesheet" media="screen and (device-width: 600px)" href="pelda.css">*
    - Példa: *<link rel="stylesheet" media="screen and (device-height: 400px)" href="pelda.css">*
  - *orientation*: álló vagy fekvő elrendezést jelent. (álló (portrait) egy elrendezés, ha a magassága nagyobb vagy egyenlő a szélességgével. Ellenkező esetben landscape (fekvő).
    - Értéke: portrait | landscape
    - Használható a min/max előtag: nem
    - Példa: *<link rel="stylesheet" media="screen and (device-width: 600px)" href="pelda.css">*

HTML5

HTML – webhely elrendezések

- HTML5 verzió megjelenése előtt a weboldalak elemeket használták az oldal szerkezetének kialakítására
- Osztály- vagy id- attribútumokat használtak az elem szerepének megadására, jelzésére
- A HTML5 új elemkészletet vezet be a szerkezet kialakításához. Az új elemek „beszédes” nevei jelzik a bennük tárolt tartalom fajtáját.

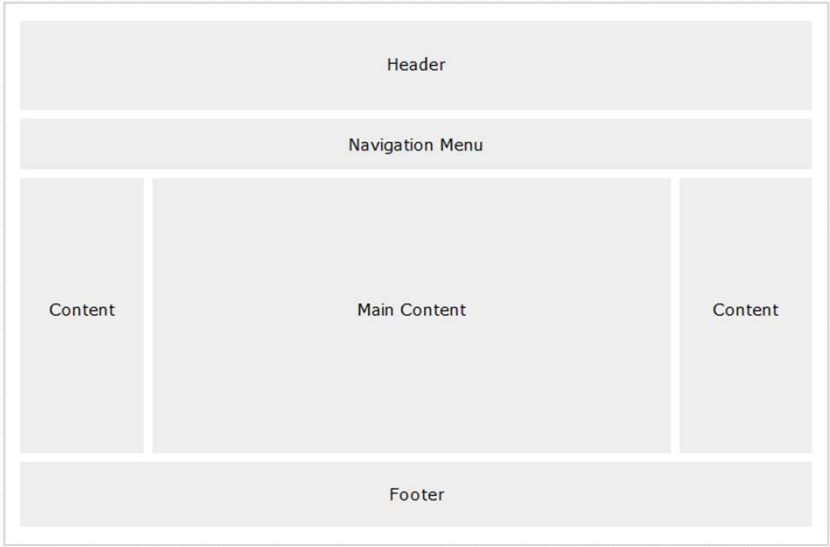




CSS3

CSS3 – webhely elrendezések

- A weblapok felületét általában több részre osztjuk, úgymint, fejléc, navigációs menü, tartalom, lábléc, amelyeket különböző módon formázunk meg.



- *Fejléc* – általában a navigációs menü fölött a lap tetején található, ami gyakran tartalmazz logót és a weblapcímet.

```
<div class="header">
    <h1>Elrendezés gyár</h1>
    <p> Méretezd át a böngésző oldalát az effektusokért!!! </p>
</div>
```

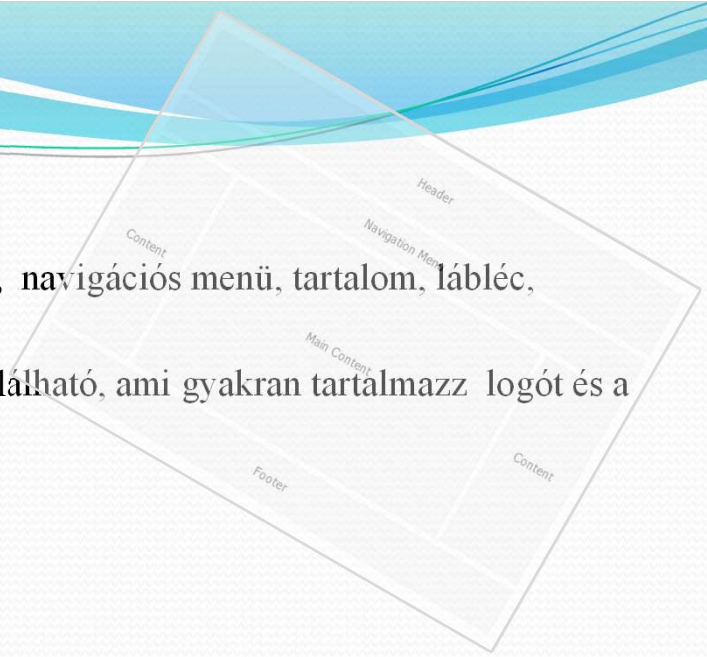
```
/* fejléc stílus*/
.header {
    background-color: #f1f1f1;
    padding: 20px;
    text-align: center;}
```

Elrendezés gyár

Méretezd át a böngésző oldalát az effektusokért!!!

HTML5

HTML– webhely elrendezések



- A weblapok felületét általában több részre osztjuk, úgymint, fejléc, navigációs menü, tartalom, lábléc, amelyeket különböző módon formázunk meg.
- *Fejléc* (header) – általában a navigációs menü fölött a lap tetején található, ami gyakran tartalmazz logót és a weblapcímet.

```
<div class="header">
    <h1>Elrendezés gyár</h1>
    <p> Méretezd át a böngésző oldalát az effektusokért!!! </p>
</div>
```

```
/* fejléc stílus*/
.header {
    background-color: #f1f1f1;
    padding: 20px;
    text-align: center;}
```

Elrendezés gyár

Méretezd át a böngésző oldalát az effektusokért!!!

- *Lábéc (footer*)– általában a lap alján található, gyakran tartalmaz szerzői jog és egyéb információkat

```
<div class="footer">
    <p>teszt.elek@tesztelek.hu</p>
</div>
```

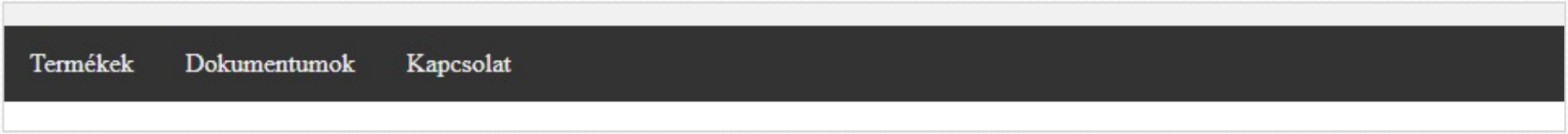
```
/* Lábléc stílus */
.footer {
    background-color: #f1f1f1;
    padding: 10px;
    text-align: left;
}
```

teszt.elek@tesztelek.hu

CSS3

CSS3 – webhely elrendezések

- *Navigációs menü* – Az oldalon való tájékozódást segítő linkek helye




```html
<div class="topnav">
    <a href="#">Termékek</a>
    <a href="#">Dokumentumok</a>
    <a href="#">Kapcsolat</a>
</div>
```

```css
.header {
    background-color: #f1f1f1;
    padding: 20px;
    text-align: center;}

/* nevigációs menü stílus */
.topnav {
    overflow: hidden;
    background-color: #333;}

/* Navigációs menő link stílus */
.topnav a {
    float: left;
    display: block;
    color: #f2f2f2;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none;}

/* Egér alatti rész stílus*/
.topnav a:hover {
    background-color: #ddd;
    color: black;}
```

CSS3

CSS3 – webhely elrendezések

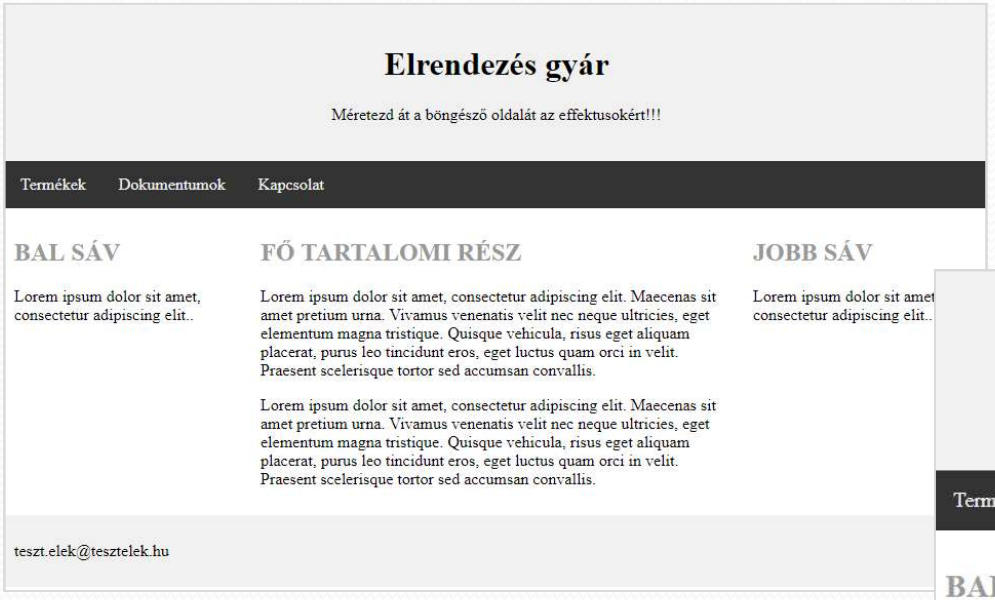
- *Tartalom*– Az oldalak ezen része gyakran függ attól, hogy kinek készítették az oldalt. Az alábbiak közül valamelyik, vagy ezek kombinációja


```
<div class="row">
    <div class="column side">
        <h2>Side</h2>
        <p>Lorem ipsum dolor sit amet, consectetur adipis
    </div>
    <div class="column middle">
        <h2>Main Content</h2>
        <p>Lorem ipsum dolor sit amet, consectetur adipis
        <p>Lorem ipsum dolor sit amet, consectetur adipis
    </div>
    <div class="column side">
        <h2>Side</h2>
        <p>Lorem ipsum dolor sit amet, consectetur adipis
    </div>
</div>
```

```
/* Három különböző oszlop stílus */
.column {
    float: left;
    padding: 10px;}
/* bal és jobb oszlop stílusa*/
.column.side {
    width: 25%;}
/* középső szlop stílus */
.column.middle {
    width: 50%;}
/* a lebegés tiltása az oszlopok után */
.row:after {
    content: "";
    display: table;
    clear: both;}
/* Alkalmazkodó megjelenés  -a három sozlopbókl egy lesz  600px alatti szélesség esetén */
@media (max-width: 600px) {
    .column.side, .column.middle {
        width: 100%;    } }
```

CSS3

- **CSS3 – webhely elrendezések**

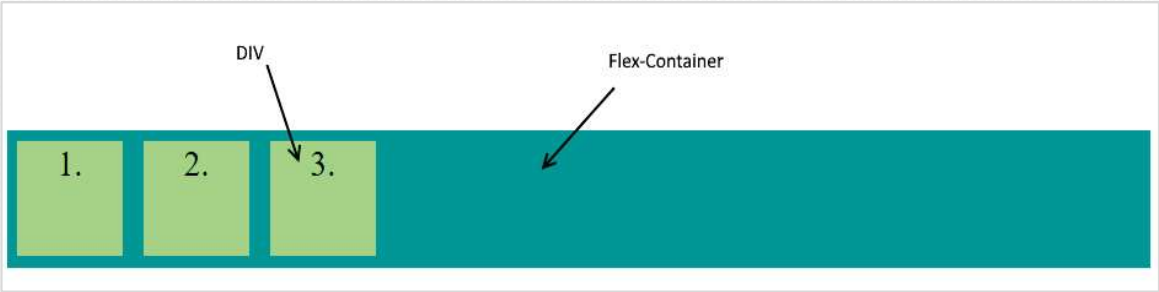


CSS3 – FlexBox (Flexibilis doboz)

- A Flexbox elrendezés modul könnyebbé teszi az oldalak Responsive (rugalmasan viselkedő) elrendezésének tervezését. Lehetővé teszi a konténeren belül a rugalmas elhelyezését és igazítását.
- A flexibilis elrendezés fő gondolata az, hogy lehetővé tegye a konténernek, hogy megváltoztassa elemeinek szélességét / magasságát (és sorrendjét is). Biztosítva ezzel a hogy az elemek a lehető legjobban kitöltsék a rendelkezésükre álló helyet anélkül, hogy túlcsordulnának (a különféle megjelenítő eszközhöz igazodva) . rugalmas tartály kibővíti az elemeket, hogy kitöltse a rendelkezésre álló szabad helyet, vagy zsugorítja azokat a túlcsordulás megakadályozása érdekében.
- A flexbox elrendezése irány-tagadó, szemben a szokásos elrendezésekkel (amelyekben a blokk, függőleges, az inline pedig vízszintes alapú szerkezett alakít ki). A szokásos elrendezésű az oldalak jól működnek, de hiányzik a belőlük a rugalmasság.
- A Flexbox elrendezés ideális webalkalmazások egyes összetevőinek megvalósítására, és kis méretű weblapok szerkezetének kialakításához. A nagyobb méretű elrendezésekhez Grid (rács) a megfelelőbb.
- Egy dimenziós elrendezést valósít meg , melyben az elemek egy tengely mentén helyezkednek el akár oszlopokról akár sorokról beszélünk

CSS3

CSS3 – FlexBox (Flexibilis doboz)



- A Flexbox elemei:
  - a Flexbox modell használatának megkezdéséhez először definiálni kell egy flex-container-t, ami egy szülőelem. ennek a display tulajdonsága : *flex* vagy *inline-flex*
  - A többi div ennek a gyereke (flex-item)
  - A gyerekek gyereke nem flex-tem

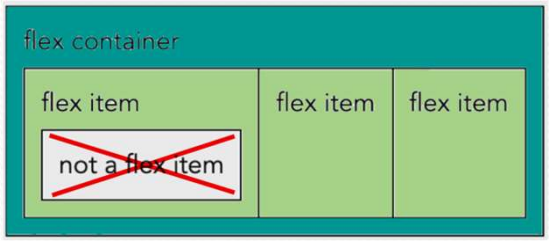
```
<!DOCTYPE html>
<html>
    <...5 lines />
    <body>
        <div class="flex-container">
            <div>1.</div>
            <div>1.</div>
            <div>1.</div>
        </div>
    </body>
</html>
```

```
.flex-container{
    display: flex;
    background-color: #009999;
}

.flex-container > div {
    width: 50px;
    height: 50px;
    text-align: center;
    vertical-align: central;
    background-color:  #99ff99;
    margin: 5px;
}
```

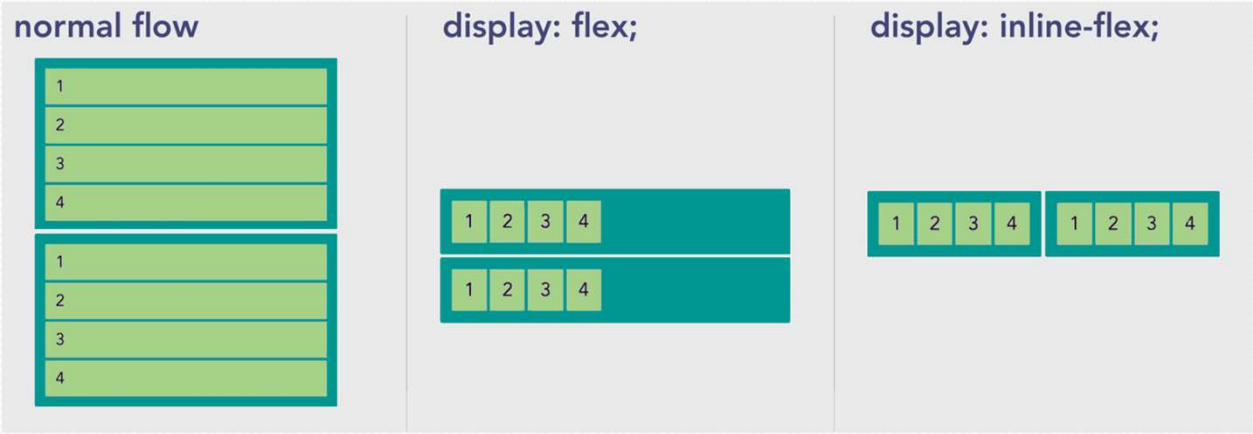
CSS3

CSS3 – FlexBox (Flexibilis doboz)

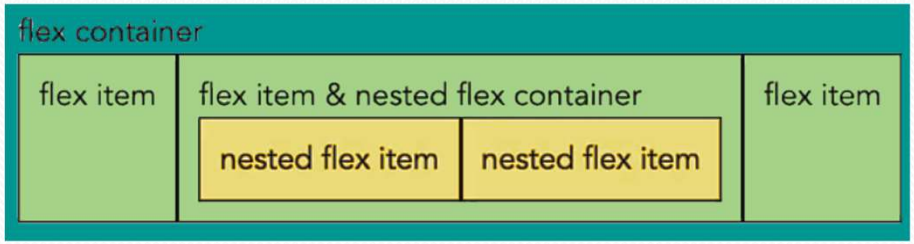
- Flex-container gyermekei



- *flex* vs. *inline-flex*



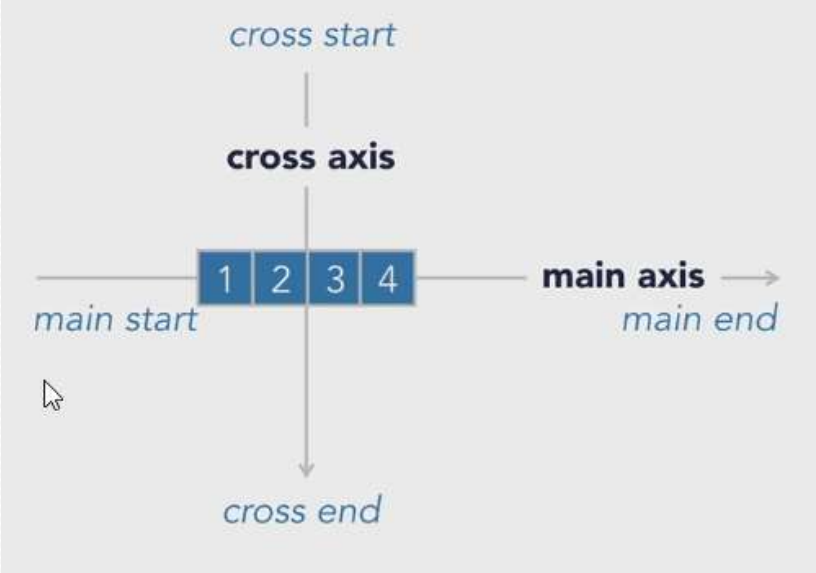
- *Beágyazott flex-container*



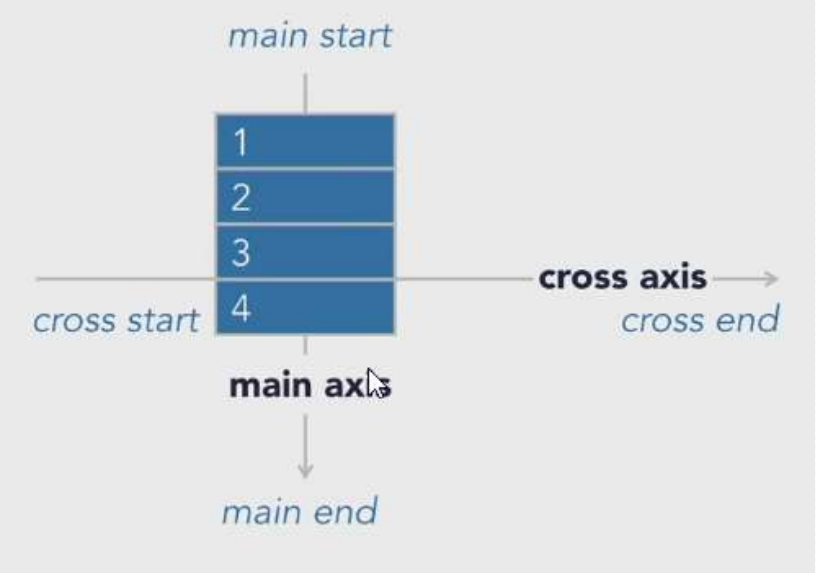
CSS3 – FlexBox (Flexibilis doboz)

- A flex-container elemeit két tengely (fő- és kereszt tengely) mentén helyezkednek el.
- Az elemek a tengelyek elejéhez és a végéhez igazodnak.
- A főtengelyt a *flex-direction* tulajdonság határozza meg .

flex-direction: row;



flex-direction:column;



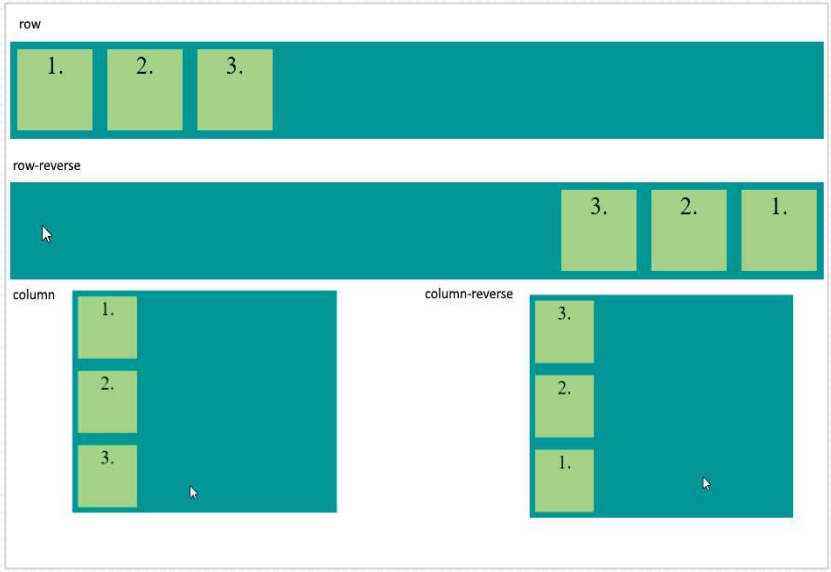
- A tengely elejét és a végét a dokumentum írásirány határozza ***ltr*** és ***rtl*** nyelvek esetén pont ellentétes.

CSS3

CSS3 – FlexBox

- A **flex-container** tulajdonságai:
- flex-direction *:{row|row-reverse|column|column-reverse}* A konténert milyen irányban töltik ki a flex elemek.

```
.flex-container{
    display: flex;
    background-color: #009999;
    flex-direction:  column-reverse;
}
```



- ***row***: Az elemek a szöveg irányával megegyezően helyezkednek el.
- ***row-reverse***: Az elemek a szöveg irányával ellentétes sorrendben helyezkednek el.
- ***column***: Az elemek fentről lefelé rendeződnek.
- ***column-reverse***: Az elemek lentről felfelé rendeződnek.

FONTOS

- *1. Vedd figyelembe, hogy ha az irányzékot ellentétesre (reversed) állítod, a 'start' és az 'end' értékek jelentése is megcserélődik.*
- *2. Vedd figyelembe, hogy ha a flex direction értéke column (oszlopos), akkor a justify-content már a függőleges, míg az align-items a vízszintes elrendezésre vonatkozik.*

CSS3

CSS3 – FlexBox

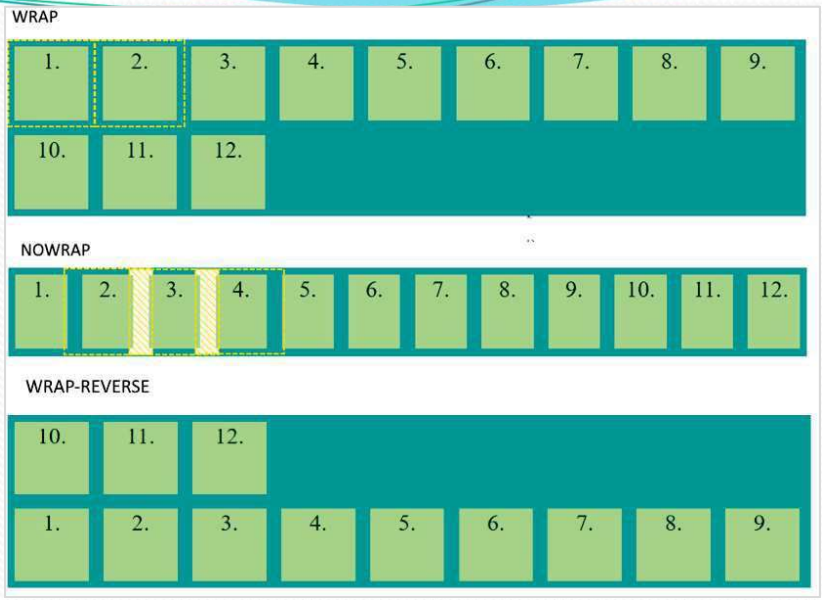
- flex-wrap :

  {wrap|nowrap|wrap-reverse}

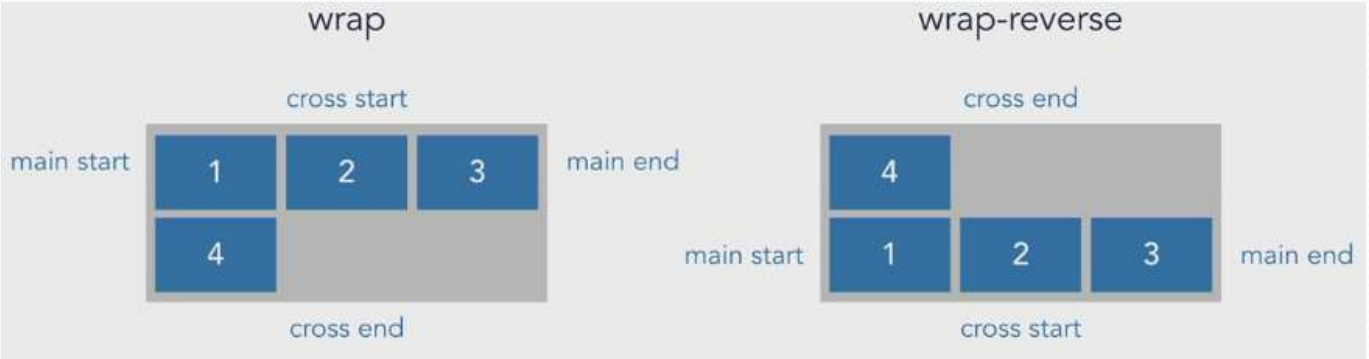
  A dobozok „sortörése” állítható be!

  Ha nincs sortörés, az elemek szükség esetén zsugorodnak

- **nowrap:** Minden elem egyetlen sorba tömörödik.
- **wrap**: Az elemek további sorokba törnek.
- **wrap-reverse:** Az elemek további sorokba törnek fordított irányban.




```
.flex-container{
    display: flex;
    background-color: #009999;
    flex-direction:  row;
    flex-wrap:  wrap-reverse;
}
```

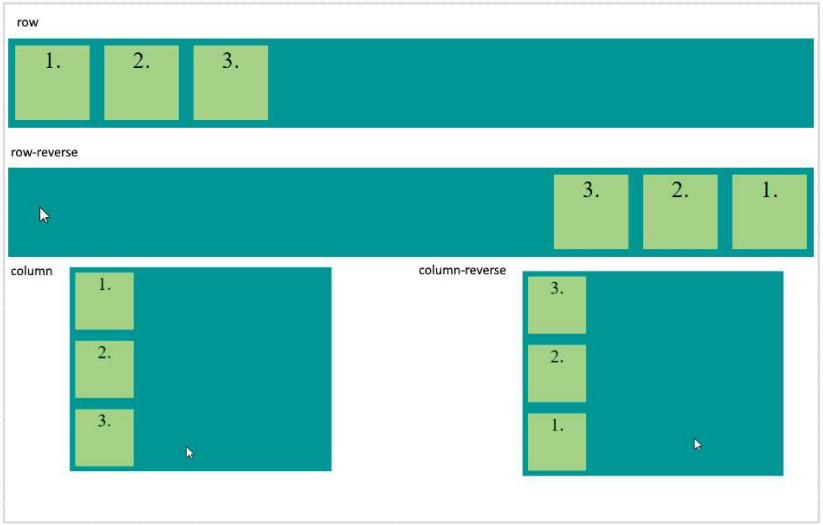


CSS3

CSS3 – FlexBox

- flex-flow: a flex-direction és flex-wrap gyors beállítására használható

```
.flex-container{
    display: flex;
    background-color: #009999;
    flex-flow: row wrap;
}
```



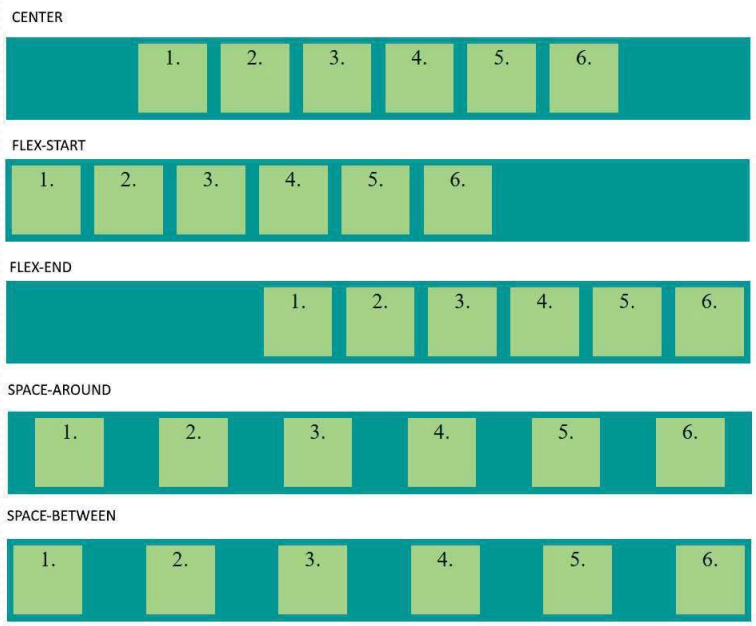
CSS3

CSS3 – FlexBox

- justify-content: A konténer elemeinek vízszintes igazítása

  {center|flex-start|flex-end|space around, space-between| space-evenly}

```
.flex-container{
    display: flex;
    background-color: #009999;
    flex-flow: row wrap;
    justify-content: space-between;
}
```



- **flex-start:** Az elemek a konténer bal oldalára igazodnak .
- **flex-end**: Az elemek a konténer jobb oldalára igazodnak.
- **center**: Az elemek a konténer közepére igazodnak.
- **space-between**: Az elemek úgy igazodnak, hogy köztük a hely egyenlő mértékben oszlik meg.
- **space-around**: Az elemek úgy igazodnak, hogy a körülöttük lévő hely egyenlő maradjon.

CSS3

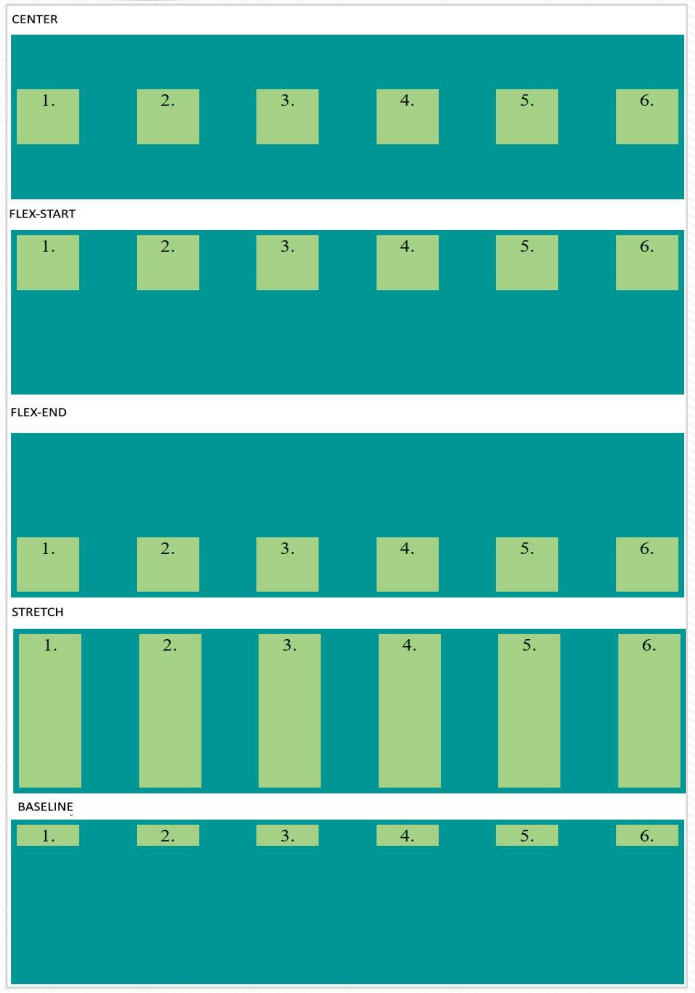
CSS3 – FlexBox

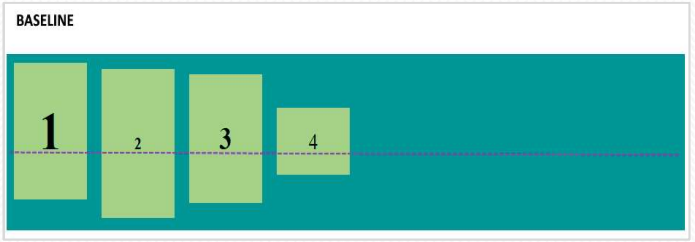
- align-items - A konténer eleminek függőleges igazítása

```
.flex-container{
    display: flex;
    background-color: #009999;
    flex-flow: row wrap;
    height: 150px;
    justify-content: space-between;
    align-items: ;
}
```

```
baseline   align-items
center     align-items
flex-end   align-items
flex-start align-items
stretch    align-items
```

- **flex-start**: Az elemek a konténer tetejére igazodnak.
- **flex-end**: Az elemek a konténer aljára igazodnak.
- **center**: Az elemek a konténeren belül függőlegesen középre igazodnak.
- **baseline**: Az elemek a konténerben a szöveg alapvonalához igazodnak.
- **stretch**: Az elemek széthúzódnak, kifeszülnek, hogy kitöltsék a konténert.





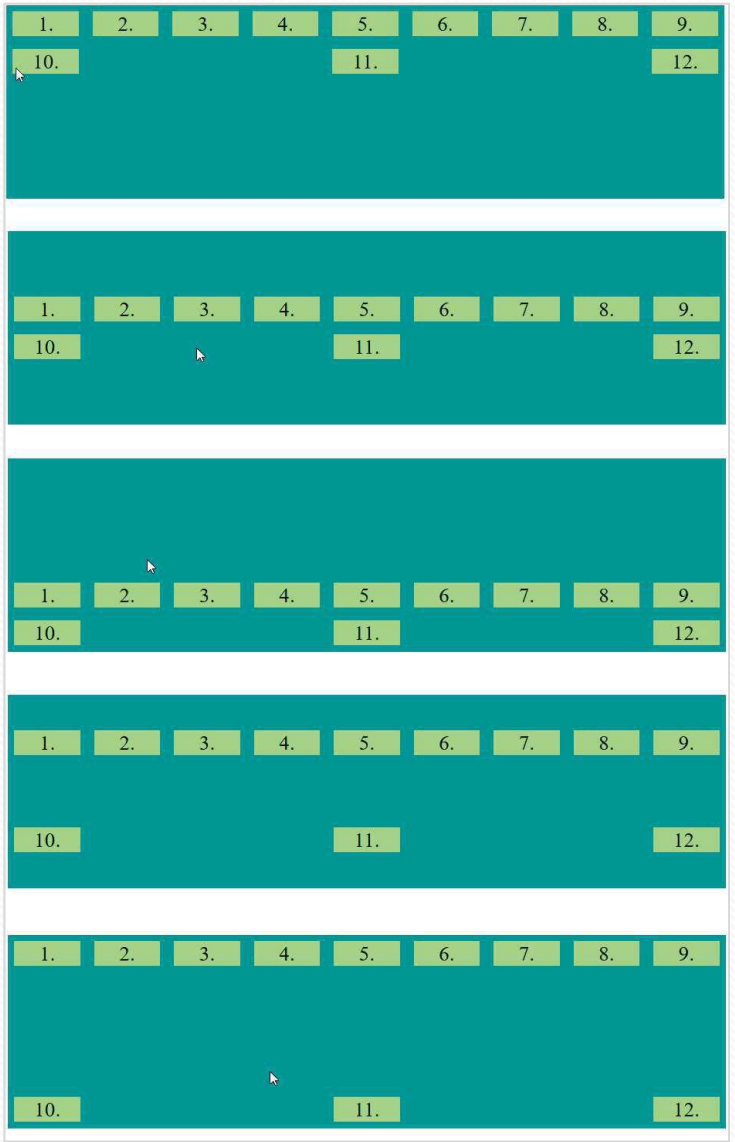
CSS3

CSS3 – FlexBox

- align-content: A konténer sorainak (a sorok közötti térnek) igazítása – hasonló a *justify-content*-hez csak függőlegesen

```
.flex-container{
    display: flex;
    background-color: #009999;
    flex-flow: row wrap;
    height: 150px;
    justify-content:space-between;
    align-items:  baseline;
    align-content:
}
```

```
center        align-content
flex-end      align-content
flex-start    align-content
space-around  align-content
space-between align-content
stretch       align-content
```



- **flex-start**: A sorok a konténer tetejére rendeződnek.
- **flex-end**: A sorok a konténer aljához rendeződnek.
- **center**: A sorok a konténeren belül függőlegesen középre rendeződnek.
- **space-between**: A sorok közötti tér kiegyenlítetten oszlik el.
- **space-around**: A sorok körülötti tér egyenlően oszlik el.
- **stretch**: A sorok széthúzódnak, hogy kitöltsék a konténerüket.

CSS3

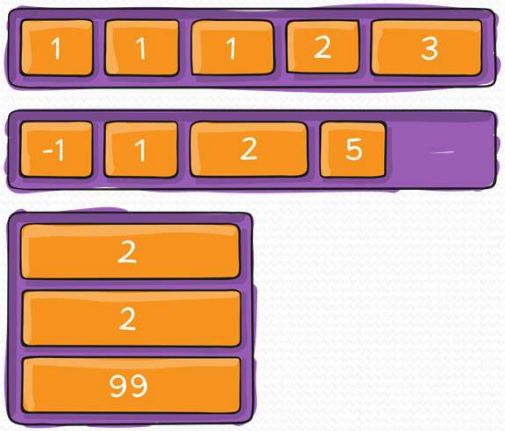
CSS3 – FlexBox

- A konténerbeni elemeket gyerek elemeknek nevezzük, ezek az elemek öröklik a flexibilitást a szülő elemtől.

- A gyerek elemek tulajdonságai:

  - **Order** – az elemek sorrendjét határozza meg (aé: 0)

```
order: <integer>; /* default is 0 */
```



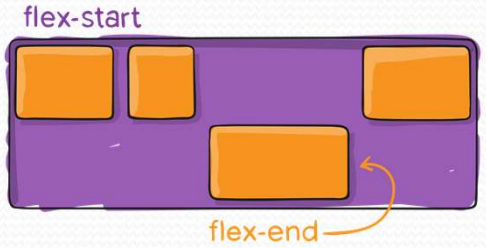
Ha a sorok vagy az oszlopok irányának megfordítása nem elégséges, akkor alkalmazhatjuk az **order** tulajdonságot az egyes elemekre sorrendjének, helyének módosítására.

Alapértelmezettként az elemek a o értéket kapják, de lehetőségünk van negatív (~előbbre) vagy pozitív (~hátrébb) egész számot megadni értékként.

Az elemek az **order** tulajdonságuknak megfelelő sorban jellennek meg a konténeren belül

  - **align-self** – felűlírja az elem alapértelmezett align-items tulajdonságát

```
align-self: auto | flex-start | flex-end | center | baseline | stretch;
```

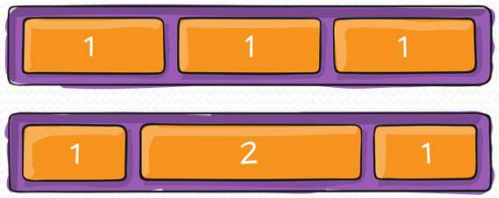


CSS3

CSS3 – FlexBox

- *1. flex-grow* – azt adja meg, mekkora lesz a flex elem a többi elemhez képest ha lehetséges . (aé: 0)

```
flex-grow: <number>; /* default 0 */
```

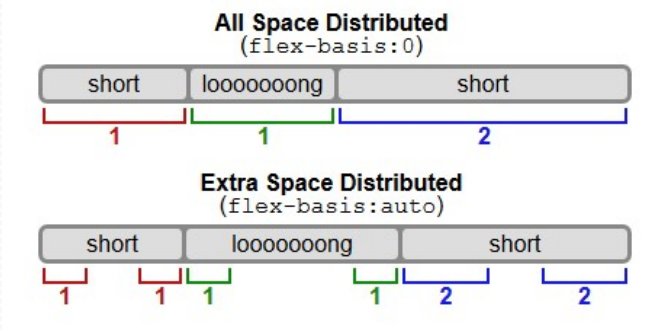


- *2. flex-shrink* – azt adja meg, hogy mekkorára zsugorodjon az elem a többi elemekhez képest ha szükséges. (aé: 1)

```
flex-shrink: <number>; /* default 1 */
```

- *3. flex-basis* – az elem kiindulási hosszát adja meg a fennmaradó terület kiosztása előtt (content, min-content, max-content)

```
flex-basis: <length> | auto; /* default auto */
```



- ***flex*** – az előző három tulajdonság rövid megadására szolgál a fenti sorrendben!!!

CSS3

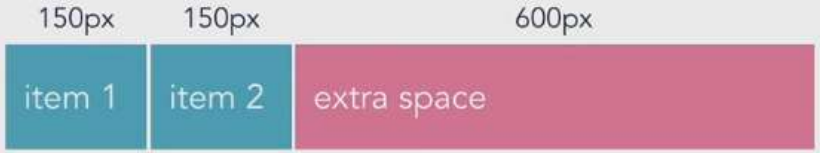
```
/* Keyword values */
flex: auto;
flex: initial;
flex: none;

/* One value, unitless number: flex-grow */
flex: 2;

/* One value, width/height: flex-basis */
flex: 10em;
flex: 30%;
flex: min-content;

/* Two values: flex-grow | flex-basis */
flex: 1 30px;

/* Two values: flex-grow | flex-shrink */
flex: 2 2;

/* Three values: flex-grow | flex-shrink | flex-basis */
flex: 2 2 10%;

/* Global values */
flex: inherit;
flex: initial;
flex: unset;
```

```
.flex-items {
  flex: 0 1 200px;
}
```


```
.flex-items {
  flex: 1 1 200px;
}
```

200px (flex-basis) + 200px grow space = 400px

```
.flex-item-one {
  flex: 1 1 150px;
}
.flex-item-two {
  flex: 2 1 150px;
}
```



flex-grow: 1 flex-grow: 2

flex-grow: 1 = 1/3 of extra space
150px (flex-basis) + 200px grow space

flex-grow: 2 = 2/3 of extra space
150px (flex-basis) + 200px grow space + + 200px grow space

CSS3 – FlexBox - Prefixek

- A Flexbox használata esetén a különböző böngészőknek megfelelő megjelenéshez böngésző specifikus előtagot (prefixet) szükséges használni
- A böngészőspecifikus előtagokon túl vannak csak egyedi böngészők által használt tulajdonságok is
- A böngésző specifikus előtagok kézzel is létrehozhatók, de az előállításukhoz praktikus kiegészítők is hasznélhatók. Pl : VsCode, Atom → autoprefixer
- Az Autoprefixer elemzi a CSS fájlokat, és hozzáadja a szükséges előtagokat a CSS szabályokhoz.

```css
.wrapper {
  display: flex;
  flex-flow: row wrap;
  font-weight: bold;
  text-align: center;
}
```

```css
.wrapper {
  display: -webkit-box;
  display: -ms-flexbox;
  display: flex;
  -webkit-box-orient: horizontal;
  -webkit-box-direction: normal;
      -ms-flex-flow: row wrap;
          flex-flow: row wrap;
  font-weight: bold;
  text-align: center;
}
```

CSS3 - Példa

CSS3 – FlexBox - Layout