Mobil- és webes szoftverek

Bootstrap, LESS és Flexbox

BME AUT
Automatizálási és
Alkalmazott
Informatikai Tanszék

Gincsai Gábor
gincsai@aut.bme.hu

Bootstrap

Ne kezdjük nulláról írni a CSS-t!

Mit ad nekünk a Bootstrap?

- Először is formázást definiál jobbára szemantikus taghez, mint a `<blockquote>`, a `<mark>` vagy a headingek, és a `<body>` tipográfiájára is ad egy ízléses alapot.
- Számos shortcut osztályt tartalmaz (pl. `success`, `warning`, passzoló színekkel és ikonokkal)
  - Bár ezek egy része nem igazán „szemantikus”, pl. `text-center`
- Sok segítséget ad az űrlapok és táblázatok formázásához.
- Főleg kiindulási alapnak használjuk, leginkább a reszponzív CSS grid miatt, amit nyújt.

A Bootstrap grid

- **Mobile first:** mobilra mondjuk meg az elrendezést, majd ezt a képernyő növekedésével fokozatosan felülbírálhatjuk.
  - Alapvetően adaptív struktúrájú, de a töréspontok között bekapcsolhatunk folytonos méretezést
- A képernyőt sorokra és oszlopokra bontja, ahol minden sor 12 azonos méretű oszlopból áll, a sorok száma és mérete azonban tetszőleges.
  - Természetesen egy konkrét tartalmi elem több oszlopon is átnyúlhat, de pl. "féloszlopon" alapvetően nem.
  - A sorok magasságát jellemzően nem "kézzel méretezzük", hanem a legmagasabb konkrét cellája határozza meg.
    - Emiatt időnként clearfixelni kell...

A grid használata

- A grid gyökere a `container` vagy `container-fluid` osztály
  - Ebbe `row` osztállyal dekorált konténereket teszünk…
  - …amikbe pedig kizárólag `col-X-*` osztályú elemek kerülnek.

	Extra small devices Phones (<768px)	**Small devices** Tablets (≥768px)	**Medium devices** Desktops (≥992px)	**Large devices** Desktops (≥1200px)
Grid behavior	Horizontal at all times	Collapsed to start, horizontal above breakpoints		
Container width	None (auto)	750px	970px	1170px
Class prefix	`.col-xs-`	`.col-sm-`	`.col-md-`	`.col-lg-`
# of columns	12			
Column width	Auto	~62px	~81px	~97px

A col-X-* osztályok viselkedése

- Először egy prefixszel megadjuk, melyik töréspont **fölött** kerül alkalmazásra
  - pl. a `col-sm-*` tableteken *és desktopokon is* érvényesül, de mobilon nem.
- A specifikusabb osztály felülírja a kevésbé specifikusat
  - pl. a `col-sm-*` felülírja a `col-xs-*`-t, mert az kevesebb képernyőméretre érvényes.
- A * adja meg, hány oszlopot foglaljon el a cella
  - Az oszlopok száma *mindig* 12, mobilon is.
  - Az alapértelmezett cellaméret viszont 12.
    - Pl. ha egy cellára csak `col-sm-*`-ot aggatunk, úgy mobilon az alapértelmezett 12 lesz a mérete, tehát teljes szélességű lesz (ez elég tipikus stratégia)

Mobil és desktop elrendezés

.col-xs-12 .col-md-8		.col-xs-6 .col-md-4
.col-xs-6 .col-md-4	.col-xs-6 .col-md-4	.col-xs-6 .col-md-4
.col-xs-6	.col-xs-6	

Itt mit rontottunk el?

.col-xs-9

.col-xs-4
Since 9 + 4 = 13 > 12, this 4-column-wide div gets wrapped onto a new line as one contiguous unit.

.col-xs-6
Subsequent columns continue along the new line.

HTML elemek reszponzív elrejtése / megjelenítése

	Extra small devices Phones (<768px)	**Small devices** Tablets (≥768px)	**Medium devices** Desktops (≥992px)	**Large devices** Desktops (≥1200px)
`.visible-xs-*`	Visible	Hidden	Hidden	Hidden
`.visible-sm-*`	Hidden	Visible	Hidden	Hidden
`.visible-md-*`	Hidden	Hidden	Visible	Hidden
`.visible-lg-*`	Hidden	Hidden	Hidden	Visible
`.hidden-xs`	Hidden	Visible	Visible	Visible
`.hidden-sm`	Visible	Hidden	Visible	Visible
`.hidden-md`	Visible	Visible	Hidden	Visible
`.hidden-lg`	Visible	Visible	Visible	Hidden

Reszponzív képek

- **Cél:** a képek töltsék ki a szülőjük nyújtotta teret, de a képarány megtartása mellett.
- Megoldás:

```
max-width: 100%;
height: auto;
display: block;
```

- Bootstrapben az `img-responsive` class-t használjuk (ami egy shortcut ugyanerre).

DEMO

Bootstrap használata

BME AUT
Automatizálási és
Alkalmazott
Informatikai Tanszék

LESS

Miért jobb a CSS-t generálni mint megírni?

Miért LESS-t használunk CSS helyett

- A LESS célja a dinamikusabb, újrafelhasználható CSS előállítása.
- A LESS a CSS kiegészítése új funkciókkal
  - Változók
  - Mixinek
  - Szabályok egymásba ágyazása
  - Függvények és operátorok

LESS fordítás CSS-re

- A kódot .less-ben írjuk, de .css-t linkelünk be a HTML oldalba.

- Fordítás VS 2015-ben
  - > Web Compiler extension segítségével.
  - > https://visualstudiogallery.msdn.microsoft.com/3b329021-cd7a-4a01-86fc-714c2d05bb6c
- .less alatt látható a lefordított .css is VS-ben.
- Hogy egy fájlba forduljon minden, definiáltuk a nexon.less fájlt amibe beimportálunk mindent.

Bootsrap CSS vagy LESS legyen?

- Van .less verzió is, amit globálisan testreszabhatunk, a **variables.less**-ben

```
//== Colors
//
//## Gray and brand colors for use across Bootstrap.

@gray-base:              #000;
@gray-darker:            lighten(@gray-base, 13.5%); // #222
@gray-dark:              lighten(@gray-base, 20%);   // #333
@gray:                   lighten(@gray-base, 33.5%); // #555
@gray-light:             lighten(@gray-base, 46.7%); // #777
@gray-lighter:           lighten(@gray-base, 93.5%); // #eee

@brand-primary:         darken(#428bca, 6.5%); // #337ab7
@brand-success:         #5cb85c;
@brand-info:            #5bc0de;
@brand-warning:         #f0ad4e;
@brand-danger:          #d9534f;
```

Változók használata

- Értékek (pl.: színek) újrahasznosítása úgy, hogy csak egy helyen kelljen megadni a színt.
- Definiáljuk a colors.less-ben

```
@cyan-bg: #289FA6;
```

- Felhasználjuk a form.less-bem

```
.default-value {
    margin-top: 5px;
    color: @cyan-bg;
    padding-left: 0;
}
```

Változók használata

Definiálás

```
//Layout méretek
@nxn-page-min-width: 1200px;
@nxn-page-min-height: 800px;
```

Használata

```
html {
    min-width: @nxn-page-min-width;
    min-height: @nxn-page-min-height;
    height: 100%;
}
```

Mixinek

- A mixinekkel lehetőségünk van, hogy egy osztályba szervezzünk több tulajdonság beállítását, amit később egy sorban újra hasznosítsuk, akár úgy is, hogy bemenő paramétert kap.

helpers.less

```
.nxn-border(@color) {
    border: 1px solid @color;
}
```

table.less

```
tr {
    .nxn-border(@cyan-border);
}
```

Szabályok egymásba ágyazása

- Az átláthatóbb kód érdekében egymásba ágyazhatók a szabályok, ami valójában egy-egy összetett szelektort fognak képezni.

```
.form-horizontal {
    .form-group {
        padding-top: 5px;
}
// Ez lesz belőle
.form-horiziontal .form-group{
     padding-top: 5px;
}
```

- Ebben az esetben nem szoktuk a &-t (parent selectort) használni.

Szabályok egymásba ágyazása

- Az átláthatóbb kód érdekében egymásba ágyazhatók a szabályok, ami valójában egy-egy összetett szelektort fognak képezni.

```
.form-horizontal {
    > form-group {
        padding-top: 5px;
}
// Ez lesz belőle
.form-horiziontal > .form-group{
      padding-top: 5px;
}
```

- Ebben az esetben nem szoktuk a &-t (parent selectort) haszánlni.

Szabályok egymásba ágyazása

```
.form-horizontal {
    &.form-group {
        padding-top: 5px;
}
// Ez lesz belőle
.form-horiziontal.form-group{
     padding-top: 5px;
}
```

- Figyeljük meg, hogy itt nincs szóköz a & és . között, így teljesen más szabályt kapunk.

Szabályok egymásba ágyazása

```
.form-horizontal {
    &-group {
        padding-top: 5px;
}
// Ez lesz belőle
.form-horiziontal-group{
      padding-top: 5px;
}
```

- Itt kötőjelet használunk a & után, az osztály nevének része lesz.

Szabályok egymásba ágyazása

```
.form-horizontal {
    &::before, &::after {
        content: '';
}
// Ez lesz belőle
.form-horiziontal:before,
.form-horiziontal:after {
      …
}
```

- Itt kötőjelet használunk a & után, az osztály nevének része lesz.

Szabályok egymásba ágyazása

```
.btn {
    & &-default {
        background-color: blue;
}
// Ez lesz belőle
.btn .btn-default {
      background-color: blue;
}
```

- Itt kétszer is használjuk egy kifejezésben a &-t.

Egy összetettebb példa

```
@nxn-window-space: 20px;
.portal {
 &-sushi {
   display: block;
   width: 100%;
 }
 &-container {
   height: ~"calc( 100% - 235.5px - @{nxn-window-space} + 50px )";
   margin: @nxn-window-space auto;
   &-single {
     height: ~"calc( 100% - 46.5px - 2 * @{nxn-window-space} )";
   }
 }
}
```

Ez lesz belőle

```
.portal-sushi {
  display: block;
  width: 100%;
}
.portal-container {
  height: calc( 100% - 235.5 - 20px + 50px );
  margin: 20px auto;
}
.portal-container-single {
  height: calc( 100% - 46.5px - 2 * 20px );
}
```

Beépített függvények

- A színek kezelését megkönnyítő beépített függvények is találhatók a less-ben.

```
//Sötétebb

@sushi-menu-selected-start:
darken(@sushi-menu-bg-start, 5%);

// Világosabb

@sushi-menu-selected-end: lighten(@sushi-
menu-bg-end, 2%);
```

Függvények és operátorok

- Lehetőség van alapműveletek elvégzésére (+, -, *, /)

```
@sushi-menu-item-width: 180px;

.generate-positions(@n, @i: 0) when (@i < @n) {
    &.sushi-position-@{i} {
        left: (@i * @sushi-menu-item-width + 18px);
    }

    .generate-positions(@n, (@i + 1));
}
.generate-positions(4);
```

DEMO

LESS demo

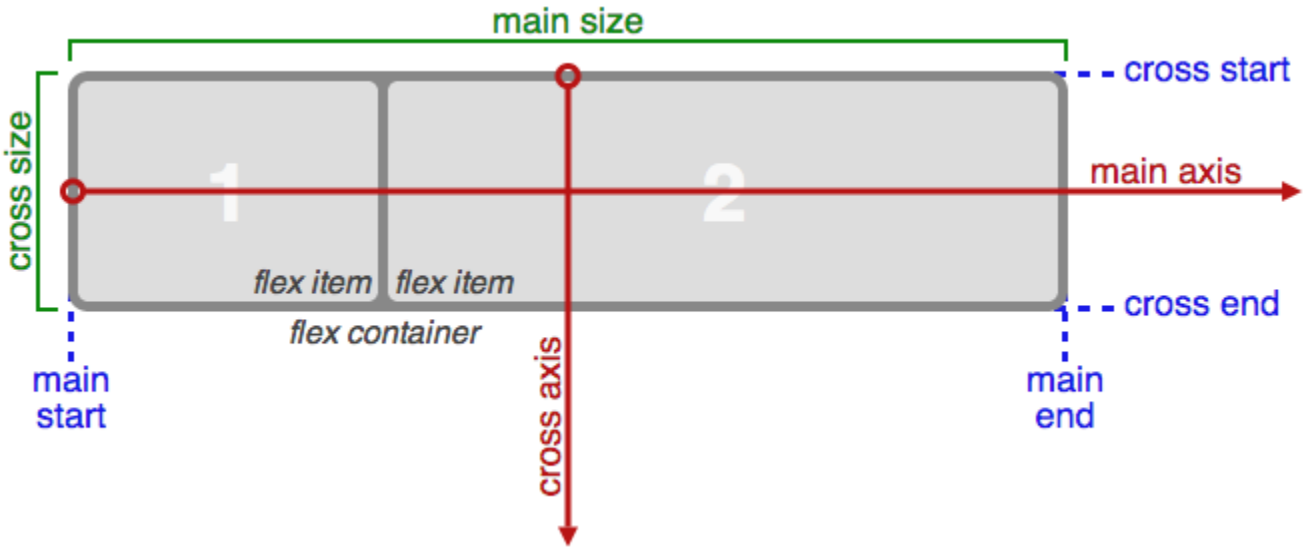
BME AUT
Automatizálási és
Alkalmazott
Informatikai Tanszék

Flexbox

Modern Layout készítése

Mit old meg a flexbox?

- Teljes(ebb) kontrollt ad egy konténer gyerekei felett, reszponzív módon
  - Elemek igazítása egymáshoz és a konténer széleihez képest
  - Elemek automatikus méretezése arányosan
  - Elemek sorrendjének variálása



Flexbox tulajdonságok (konténer)

- Először is `display: flex` a konténeren, hogy bekapcsoljuk
- `flex-direction`: gyerekek elrendezése vízszintes (`row`) vagy függőleges (`column`) tengely mentén.
- `flex-wrap`: Sor-/oszloptörés engedélyezése.
- `justify-content`: elemek rendezése, tagolása a főtengely mentén

justify-content

Flexbox: justify-content és flex-grow

A PEN BY Gincsai Gábor

Save | Fork | Settings | Change View

HTML

```
<div id="main">
  <div style="background-color:coral;"></div>
  <div style="background-color:lightblue;"></div>
  <div style="background-color:khaki;"></div>
  <!-- Ő kapjon flex-grow: 1-et -->
  <div style="background-color:pink;"></div>
</div>
```

CSS

```
#main {
    width: 400px;
    height: 100px;
    border: 1px solid #c3c3c3;
    display: flex;
    justify-content: space-between;
    /*justify-content: space-around;*/
    /*justify-content: space-evenly;*/
}
#main div {
    width: 70px;
    height: 70px;
}
```

https://codepen.io/gincsai/pen/POPMBg

A flex-start és a flex-end értelmezése

	flex-start	flex-end
row	balra	jobbra
row-reverse	jobbra	balra
column	fentre	lentre
column-reverse	lentre	Fentre

További konténer tulajdonságok

- `align-items`: a gyerekek igazítása a főtengelyre merőlegesen.
  - Hasonlít a `justify-content`-re, de fontos különbség, hogy nem a *gyerekek között megmaradt* helyet osztja be, hanem a gyerekek és a *flexbox széle* közötti helyet.
  - Tehát ez akkor is működik, ha valamelyik gyerek „nyúlós” (mivel az a főtengely mentén nyúlik).
  - A gyerekek a kereszttengelyen is nyújthatók, erre van a `stretch` opció
    - Gyakori hiba azt hinni, hogy a `justify-content` rendelkezik `stretch` értékkel, de nincs neki, mivel a főtengelyi menti nyúlást a „gyerekek döntik el” (`flex-grow`, ld. később)

align-items

HTML

```
<div id="main">
  <div style="background-color:coral;">
    Első rövid oszlop.
  </div>
  <div style="background-color:lightblue;">
    Második kicsivel hosszabb szövegű oszlop.
  </div>
  <div style="background-color:khaki;">
    Harmadik
  </div>
  <!-- Ő kapjon flex-grow: 1-et -->
  <div style="background-color:pink;">
    Negyedik oszlop, aminek a szövege nagyon-nagyon nagy.
  </div>
</div>
```

CSS

```
#main {
    width: 400px;
    height: 150px;
    border: 1px solid #c3c3c3;
    display: flex;
    align-items: flex-start;
    /*align-items: center;*/
    /*align-items: flex-end;*/
    /*align-items: baseline;*/
    /*align-items: stretch;*/
}
#main div {
    width: 70px;
    min-height: 60px;
}
```

https://codepen.io/gincsai/pen/YEwKaK

További konténer tulajdonságok

- `align-content`: többsoros flexbox *sorainak* igazítása a kereszttengely mentén.
  - Egysoros flexboxra nincs hatása.
  - Értékei kb. ugyanazok és kb. ugyanazt jelentik, mint a `justify-content` esetén, azonban itt van külön `stretch` érték is (ami a default), hiszen itt nem a főtengelyről, hanem a kereszttengelyről van szó, tehát nem a gyerekektől függ a „nyúlás”.

Flexbox gyerek tulajdonságai

- `flex-grow`: szabályozza, hogy a gyerek nyúljon-e – és a többiekhez képest milyen arányban –, ha marad hely a főtengely mentén. Alapértelmezett: 0 (nincs nyúlás)
- `flex-shrink`: összenyomható-e a gyerek, ha kevés a hely, és nem lehet sort törni; és ha igen, milyen arányban.
  - Alapértelmezett: 1
- `flex-basis`: a gyerek alap mérete nyújtás/összenyomás előtt, alapértelmezetten `auto`.
- `flex`: shorthand az előző háromra.
  - `flex: 1 1 auto` – teljesen „rugalmas”
  - `flex: 0 0 auto` – teljesen „rugalmatlan”

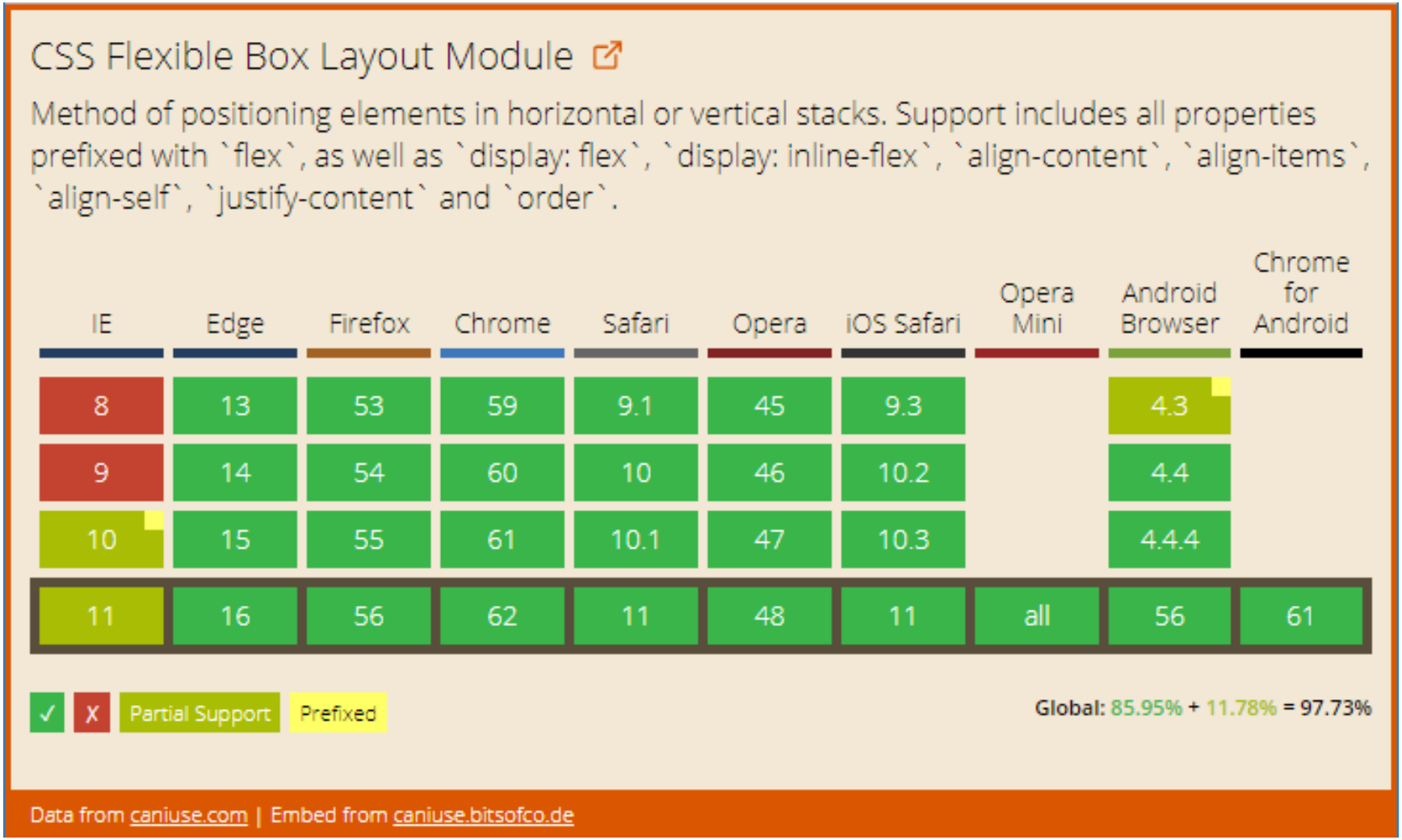
flex-basis részletek

- Alapértelmezett értéke az `auto`, ami a gyerek főtengely menti méretét (**`width`** vagy **`height`**) értékét veszi fel.
  - Ha az *is* `auto`, akkor a tartalomhoz igazodik a méret.
- Megadhatunk konkrét értéket is (pl. 25px, 33%), ezzel effektíve felülírjuk a főtengely menti méretet.
  - Ez akkor lehet hasznos, ha variáljuk a tengelyt (pl. mobil nézet), és emiatt simán a **`width`** vagy a **`height`** felülírása nem lenne elég rugalmas.

További flexbox gyerek tulajdonságok

- `align-self`: a gyerek „felülírhatja” saját magára a konténeren beállított `align-items` értéket.
- `order`: a gyerekek alapvetően abban a sorrendben jelennek meg, ahogy az a HTML-ben szerepel. Az `order-rel` ezen variálhatunk.
  - Negatív szám is lehet, alapértelmezetten 0.
- Van néhány „hagyományos” CSS tulajdonság, ami nem alkalmazható flexbox gyerekekre.
  - `float`, `clear`, `vertical-align`

Használhatom a flexboxot?



CSS Flexible Box Layout Module

Method of positioning elements in horizontal or vertical stacks. Support includes all properties prefixed with `flex`, as well as `display: flex`, `display: inline-flex`, `align-content`, `align-items`, `align-self`, `justify-content` and `order`.

IE	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Chrome for Android
8	13	53	59	9.1	45	9.3		4.3	
9	14	54	60	10	46	10.2		4.4	
10	15	55	61	10.1	47	10.3		4.4.4	
11	16	56	62	11	48	11	all	56	61

✓ X Partial Support Prefixed

Global: 85.95% + 11.78% = 97.73%

Data from caniuse.com | Embed from caniuse.bitsofco.de

DEMO

Flexbox használata

https://codepen.io/gincsai/pen/eeJYmb

align-items, justify-content, flex-wrap, flex-shrink, flex-grow

BME AUT
Automatizálási és
Alkalmazott
Informatikai Tanszék