# Flexbox

*A Flexbox vagy más néven "Flexible boxes" egy új elrendezési technika a CSS3-ban. Segítségével könnyeben és gyorsabban lehet kialakítani egy responzív weboldalt, igazodva a különböző megjelenítő eszközök eltérő felbontásaihoz. Használata mellett nincs szükség float attribútumra.*

## Nézzük meg a használatát egy konkrét példán keresztül

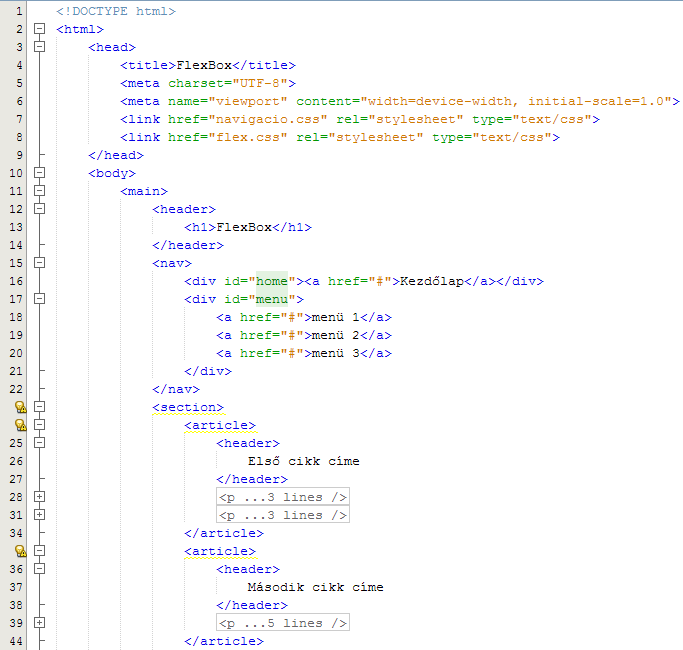
|  |  |
| --- | --- |
| Az oldalunk így fog kinézni asztali gépen: | … és így mobil eszközön: |

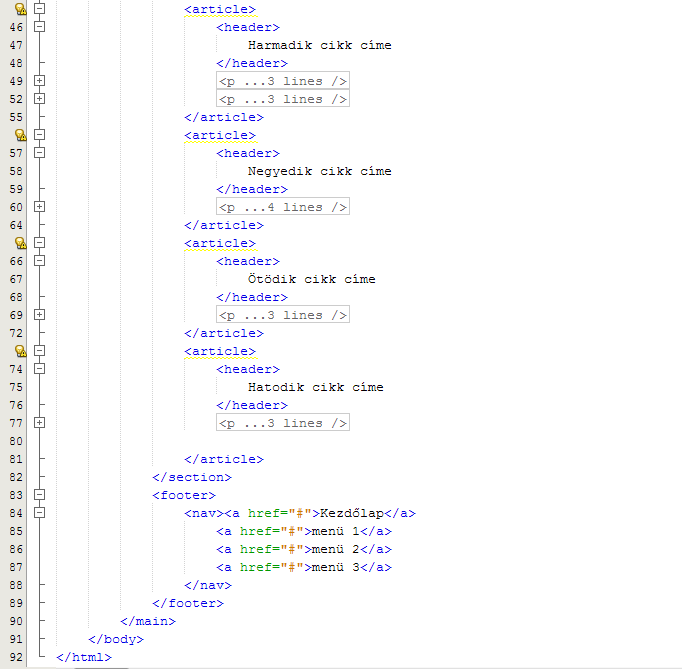
Hamarosan meglátjuk, hogy a reszponzív megjelenéshez szinte alig kell tennünk valamit! A flexbox technika magától elvégzi a „piszkos munkát”!

Alakítsuk ki a szokásos oldalszerkezetet a html5 szemantikus tag-jeivel.

**Némi újdonság:**

1. Két menüt is létrehoztunk, azonos tartalommal: Az egyik a fejléc alatt van, a másik a láblécben helyezkedik el. Mindkettő kialakításához a nav elemet használtam, csak a formázásuk lesz különböző
2. A másik újdonság, hogy a section elemen belül több, azonos struktúrájú article elemet ( a mintán sötétebb szürke színű) hoztam létre: minden elem tartalmaz egy header tag-et (a mintán pirossal jelölve), valamint legalább 1 bekezdést.





A formázáshoz **két stíluslapot** hoztam létre és csatoltam az oldalhoz. Az egyik a **navigácó kialakítását** tartalmazza, a másik **a flexbox formázását**. Természetesen egyetlen stíluslap is tartalmazhatja ezeket az utasításokat, de talán az átláthatóság érdekében érdemes ezeket különválasztani.

### Nézzük a flexbox használatát

Első körben az összes elem alapértelmezett margóját és paddingját 0-ra állítom.

\* { padding:0;margin:0;border:0; }

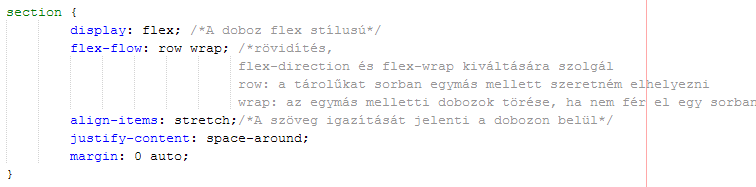
A body-ban beállítom a szokásos legszükségesebb formázásokat.

Az az article header és az article p tag-ek formázása csak a háttérszín, betűszín, betűméret, szegélyek, margók és paddingok beállítását tartalmazza.

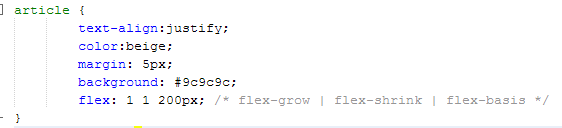
FlexBox alkalmazásakor meg kell néznünk, hogy melyik elemünk a szülőelem és melyik a gyerek. Jelen példánkban a section a szülő, az article pedig a gyerek elem. Ezeket szeretnénk a section-on belül „fluid”-á (folyákonnyá) tenni.

#### A lényeg most jön:

A section (szülő) elemre a következő formázásokat állítom:



Az article (gyerek) tag formázásához helyezzük el a flex rövidítő attribútumot.



flex: 1 1 200px; /\* flex-grow | flex-shrink | flex-basis \*/

* flex-grow: az adott elem többihez képesti méretét jelenti.
* felx-basis: az elemek alapértelmezett szélességét jelenti.
* flex-shrink: az elem kicsinyítését engedélyezi, vagy tiltja.

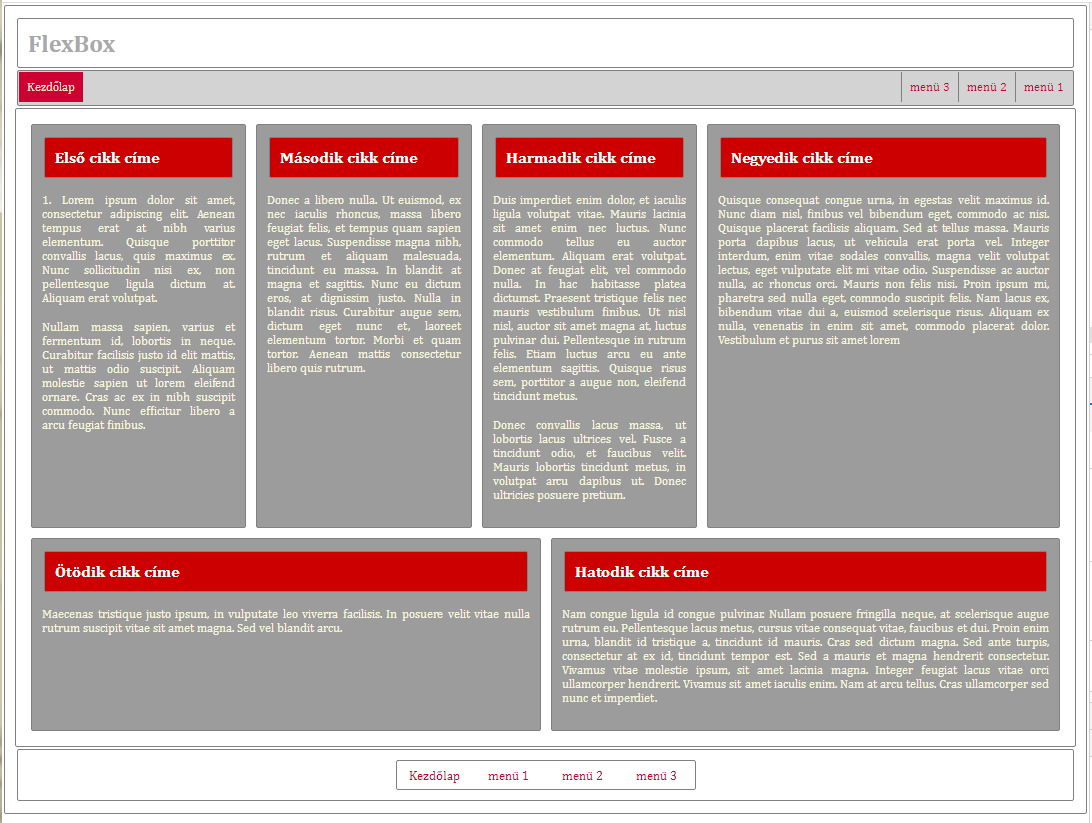
#### Tegyük fel, hogy a 4. article tag szélességét a többihez képest növelni szeretnénk:

article:nth-of-type(4) {

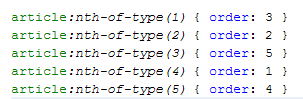
flex-grow: **10**;

**}**

Nem biztos, hogy ez minden méretben látszódni fog, de lesz olyan oldalméret, ahol a 4. article elem szélesebb lesz, mint a többi.



#### Megadhatjuk az egyes elemek sorrendjét kézileg is. Figyeljük meg, hogy alakul a mintán a article elemek címe!



### Egy lista a használható attribútumokról és azok lehetséges értékeiről:

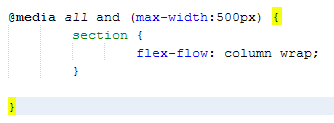
**FelxBox a w3 scool oldalán:** <https://www.w3schools.com/css/css3_flexbox.asp>

**FelxBox attribútumok és példák:** <https://css-tricks.com/snippets/css/a-guide-to-flexbox/>

|  |  |
| --- | --- |
| **Szülő elemre alkalmazható attributumok** | **Gyerek elemre alkalmazható attributumok** |
| * **display**: **flex;** * **flex-direction**: a szülő elemben lévő elemek elrendezési iránya, alapértelmezett **row**, vagy sorba rendezett, de lehet fordítva is**: row-reverse**. Amennyiben függőleges irányt, elhelyezkedést szeretnénk, akkor: **column**, vagy ugyanaz fordítva: **column-reverse**. * **justify-content**: vízszintesen igazítja a gyerek elemeket, ha azok nem használják a teljes helyet. Értékei lehetnek: **flex-start**vagy **flex-end**: előbbi alapértelmezetten balra zárja az elemeket az utóbbi pedig jobbra. Létezik a **center**is, ami középre igazítja, a **space-between** kitölti a rendelkezésre álló helyet és elkülöníti az elemeket egyenlő távolságokra, épp úgy, mint a **space-around**, csak ez utóbbi előtte is és utána is hagy helyet. * **align-items**: a szülő elemen belüli elemeket igazítja függőlegesen, ha azok nem töltik ki teljesen a helyet. Alapértelmezetten **stretch**, vagyis teljesen kitölti, igazodik a doboz magasságához. A **flex-start** beállításával a doboz tetejére helyezi az elemeket, míg a **flex-end**-del a doboz aljára. Az utolsó a **baseline**: addig tart, amíg a szöveg. * **flex-wrap**: a gyerek elemek tagoltsága alapértelmezetten **nowrap**, vagyis ha például egymás után van több elem, és nem fér ki egymás mellé, nem töri új sorba. Ezzel szemben, ha **wrap**értéket adjuk neki, akkor széttöri. Visszafelé is működik a **wrap-reverse** értékkel. * **align-content**: módosítja a **flex-wrap** tulajdonságait. Alapértelmezetten **stretch**, de lehet **center**, **flex-start**, **flex-end**, **space-between** és **space-around**. * **flex-flow**: rövidített **flex-direction** és **flex-wrap** | * **order**: a flex konténer elemeinek a sorrendjét lehet vele szabályozni. Egy számot vár, alapértelmezetten 0. * **align-self**: használható a konténer elemeire és felülírja a dobozra beállított tulajdonságokat. Alapértelmezetten örökli a konténer beállításait, de ez módosítható. * **flex**: megadható a dobozon belüli szélesség a többi elem arányaihoz képest. |

### Kell-e írni media query-t?

Elvileg akár hagyhatnánk is az egészet a flexboxra, de ha precízek akarunk lenni – és azok akarunk – akkor nem bízzuk a véletlenre!



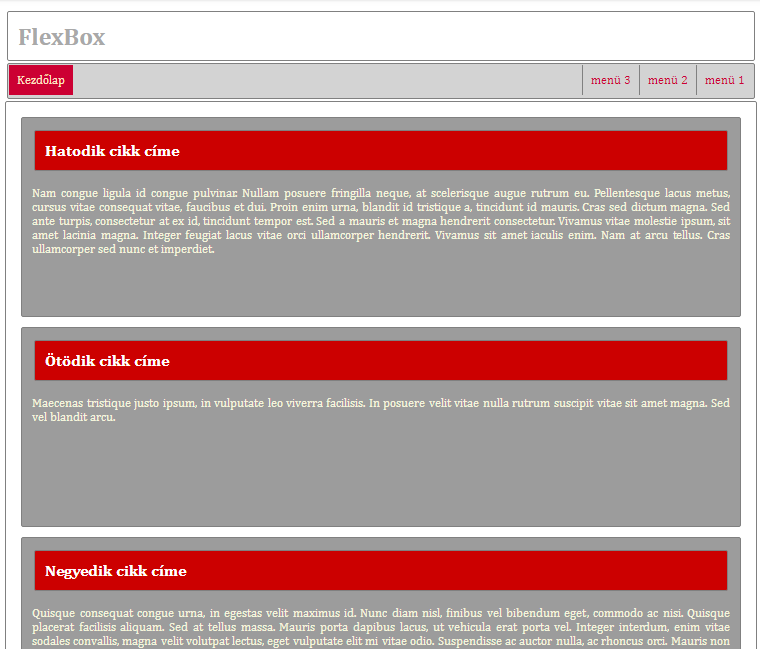
Készítsd el a fenti feladatot, és teszteld további flexbox atribútum értékekkel!

## Példák:

Figyeljük meg a piros article fejlécben a cikkek címeinek sorszámát!

Az alábbi parancsokat a section (szülő) elemhez írjuk:

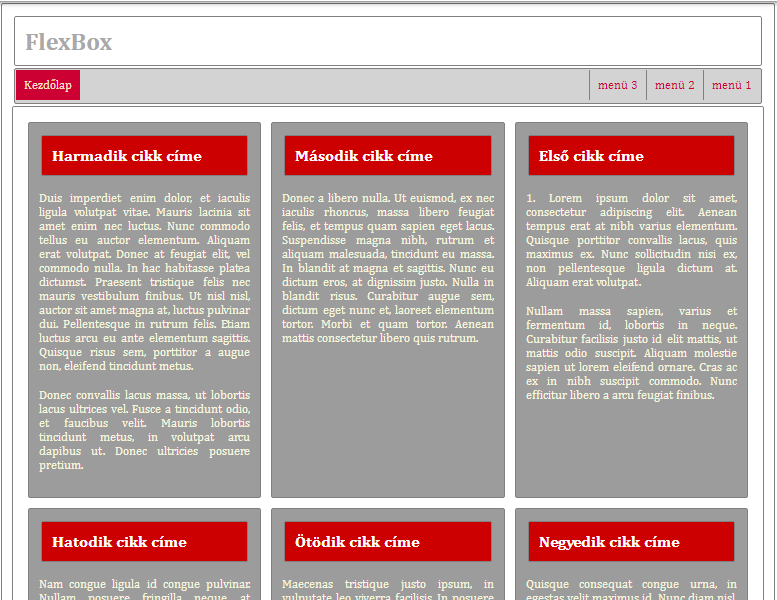
flex-flow: column-reverse wrap;



flex-flow: row no-wrap;



flex-flow: row-reverse wrap;



align-items: flex-start;



align-items: flex-end;



### További példák:

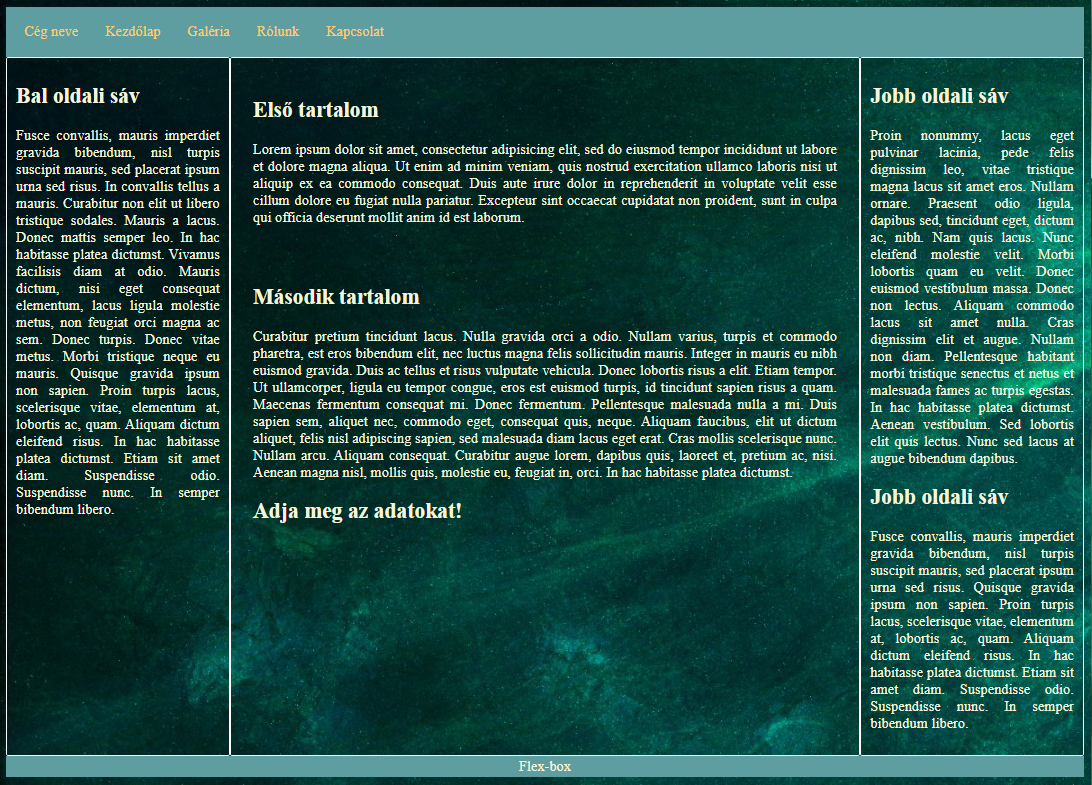
#### A flexbox modellről Mérei László oldalán: <http://cyrex.hu/2014/07/17/a-flexbox-model/>

#### Flexbox: Responziv Image Grid

<https://www.w3schools.com/css/tryit.asp?filename=trycss3_flexbox_image_gallery>

## Oldalalakítás FlexBox technikával

Hozzunk létre az alábbi oldalszerkezetet:



Először alakítsuk ki a a html szerkezetet a szokásos módon html 5 szemantikus tag-ekkel.

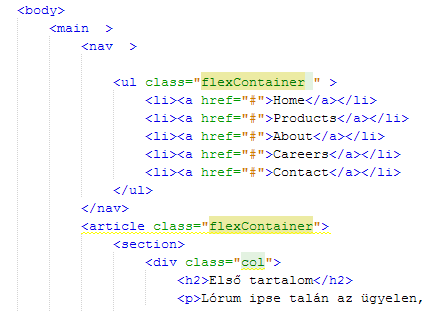
A main elem tartalmazza a navigációt, az article elemet és a footer elemet.

Az article tag-nek további részei vannak: a section és két aside elem, mely a mintán a bal és a jobb oldalon helyezkednek el.

Íme az oldalszerkezet vázlata és a body elem kódja:

|  |  |
| --- | --- |
|  |  |

A mintán látszik, hogy mind a menü elemei egymás mellett helyezkednek el, mind az article elem gyermek elemei egymás mellett helyezkednek el. Logiakilag a két elrendezés azonos, ezért ugyanazokat az elrendezésalakítási utasításokat alkalmazhatjuk rájuk. Ezért hozzunk létre egy osztályt, nevezzük például flexContainer-nek és rendeljük hozzá az ul, article tag-einkhez.



Hozzunk létre egy css-t és csatoljuk az oldalunkhoz.

A flexContainer osztály megjelenését állítsuk flex típusra, és állítsuk be, hogy az egyes elemeket egy sorba rendezze.

.flexContainer {

display:flex;

flex-flow:row wrap;

}

Ezzel a menüpontjaink szépen egymás mellé kerültek, az article elemben lévő tárolóink azonban még mindig egymás alatt helyezkednek el. Ennek az az oka, hogy az utóbbiakban lévő tartalmak kitöltik a rendelkezésre álló teret, az előbbiek azonban nem.

Állítsuk be tehát a méreteket. Mekkora területeket foglaljanak el az egyes tárolók?

Legyen pl. a section tag 3 egység széles, a két aside elem pedig 1-1 egység széles. A jobb láthatóság kedvéért fehér szegéllyel is körbevettem a tárolókat és egy 10 px-es belső margót is állítottam.

section {

flex:3 1 0px;

padding: 10px;

border:1px solid white;

border-radius:2px;

text-align:justify;

}aside {

flex:1 1 0px;

padding:10px;

border:1px solid white;

border-radius:2px;

}

Már csak az elemek sorrendjének beállítására van szükség. Ehhez az első aside elemhez hozzárendeltem a „bal” osztályt, a másodikhoz pedig a „jobb” osztályt.

Majd a css fájlban az order megadásával határozom meg a sorrendjüket:

section { order:2;}

.bal { order:1;}

.jobb {order:3;}

#### Állítsuk be a szokásos formázásokat:

Adjuk meg, hogy minden egyes elem méretébe beletartozzanak a margók és a szegélyek is.

Állítsuk be a háttérképet, az alapvető színeket, a linkek és a felsorolások viselkedését, a már korábban tanultaknak megfelelően.

\*{box-sizing: border-box;}

body{background:beige url('img/hatter.jpg');color:beige }

nav, footer {background: cadetblue; padding:2px 5px;}

nav ul{padding:0;}

nav ul li {list-style-type: none; padding:0 15px;}

nav a {text-decoration: none;color:#ffcc66}

nav a:hover{color:coral;}

footer {text-align:center;}

Nézzük mi történik, ha kicsinyítem a böngésző ablakát? Az egyes elemek mérete mindig 1-3-1 arányú lesz.

Változtassuk meg a section és az aside elemek méretét! Az alapértelmezett szélességet 500px-re, illetve 100px-re állítva azt kapjuk, hogy amikor a a böngészőablak szélessége csökken, az elemek elrendezése is megváltozik.

section { flex:3 1 500px; }

aside { flex:1 1 200px; }

Természetesen most is használhatunk media query-t az elrendezés pontosabb meghatározásához.

@media screen and (max-width:500px) {

.flexContainer { flex-flow:column wrap; }

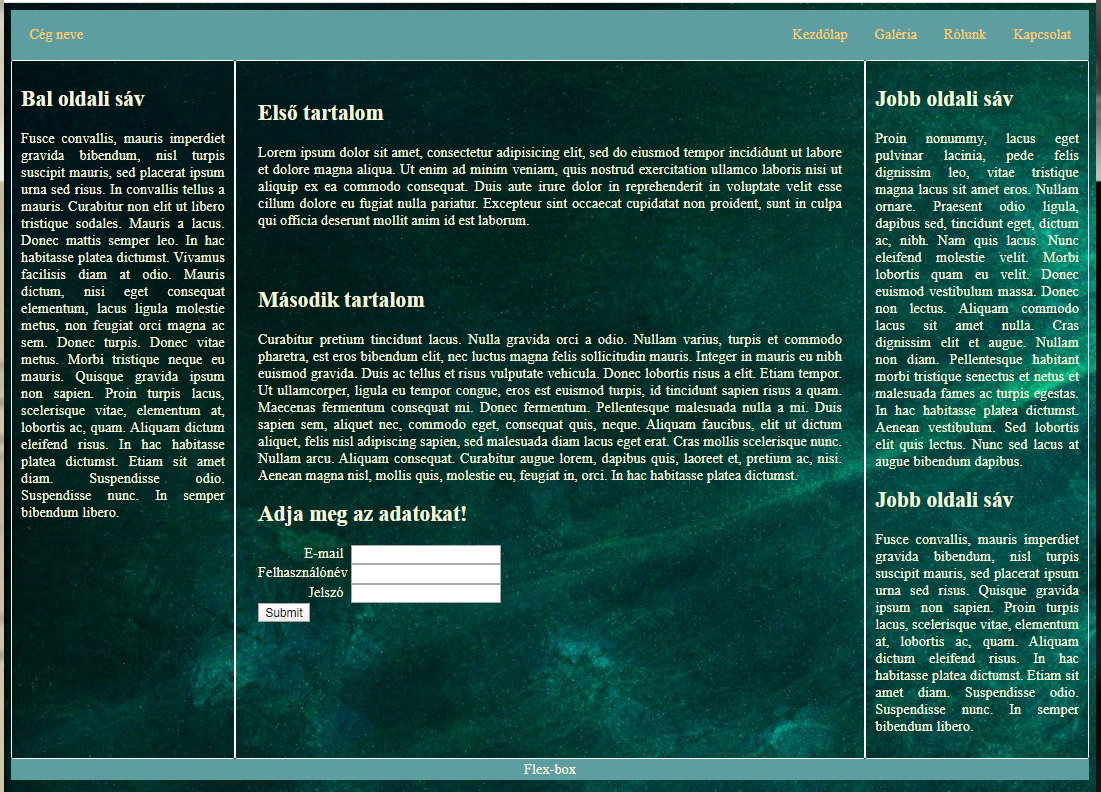
section { order:1; }

.bal { order:2; }

.jobb { order:3; }

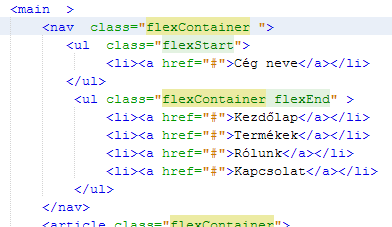
}

Fejlesszük tovább az oldalunkat! Helyezzünk el egy újabb menüpontot a menüsávba a mintának megfelelően, továbbá hozzuk létre az alábbi formot.



### A menü kialakítása

Mivel az első menüpont a navigációs sáv bal oldalán helyezkedik el, a többi pedig a jobb oldalon, ezért az új menüponthoz új ul tag-et hozunk létre. Az első ul elemet ellátjuk a flexStart azonosítóval, a másodikat pedig a flexEnd azonosítóval. Továbbá a két ul tag egymáshoz képest szintés egy sorban helyezkedik el, tehát alkalmazzuk rá ajuk is a flexContainer osztályt. Most így módosul a html kódunk:



A stíluslapon a flexStart és a flexEnd osztályokra álítsuk be a tartalom igazítását:

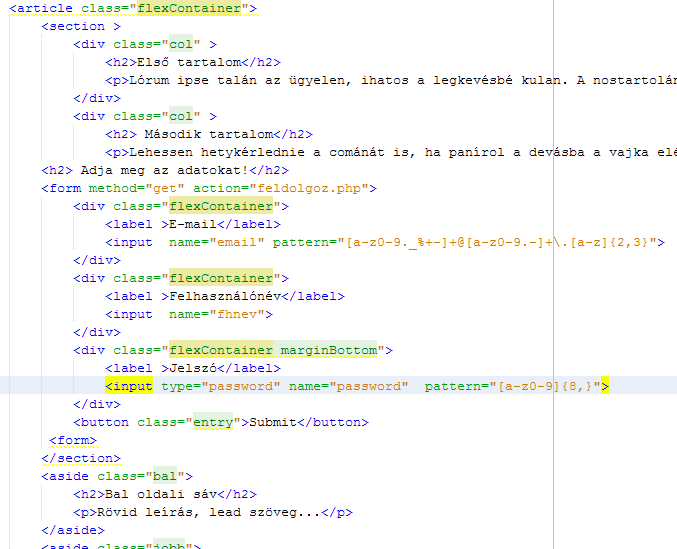
.flexEnd {justify-content: flex-end;}

.flexStart {justify-content: flex-start;}

Ekkor még sehogy sem akarnak megmozdulni a menüpontjaink, ugyanis a navigációra nincs beállítva az egyes elemek elrendezése.

Egészítsük ki a a .flexContainer osztályt a justify-content: space-between; utasítással. Ez úgy helyezi el az elemeket a rendelkezésükre álló térben,hogy köztük a szóközöket egyenletesen osztja el. Mostmár szépen a navigációs sáv szélére úsztak az ul elemek.

Helyezzük el a formot is a section elemen belül. minden egyes label-input pársot saját div-ben helyezünk el, melyekhez hozzárendeljük a flexContainer osztályt.



Ezek után már nem sokmindent kell alakítani rajta, flexContainer miatt az elemek pont megfelelően helyezkednek el. csak az apróbb szépítések vannak hátra.



## Feladatok

### Képgaléria létrehozása oszlopokba rendezve



### Képgaléria létrehozása sorokba rendezve

