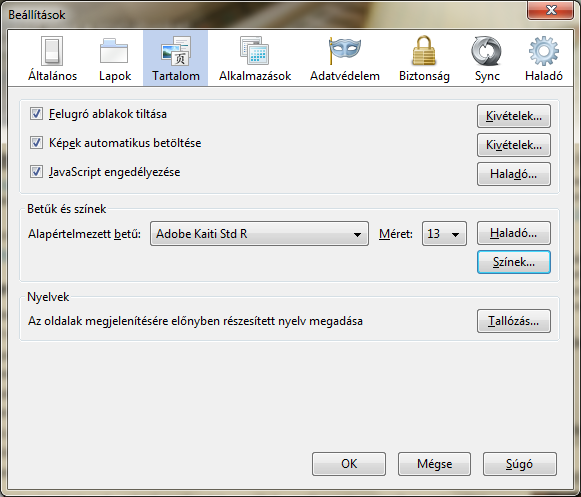
1. gyakorlat – Formázzuk meg oldalunkat stílusok segítségével!   
   A CSS alapjai

Az előző gyakorlatban elkészített oldalunk szépen megjelenik a böngészőprogramban, mivel a böngészőben a h1, h2, p cimkék betűtípusához, méretéhez, szinéhez hozzá volt rendelve egy alapértelmezett érték. Ennek az alapértelmezett értéknek megfelelően jelenítette meg a program a kódunkat.

A böngésző is stíluslapokat használ egy olyan oldalhoz is, amihez mi semmilyen stílust nem adtunk meg. A böngésző ebben a stíluslapban olyanokat definiál, hogy például a h1 címsor vastagon és nagyobb betűmérettel jelenjen meg; a linkek kékek és aláhúzottak legyenek, stb. Érhető módon ezt a stíluslapot akármelyik más felül fogja írni, hiszen ez az alapértelmezett megjelenést írja le a böngésző által.

Ezt a Firefox 13.0.1-es változatában például a Beállításoknál lehet megtekinteni, illetve módosítani:



A beállítások különböző böngészők esetében más és más lehet. Jó lenne ezt a megjelenítést nem bízni a böngészők alapértelmezett értékeire. Ezért nekünk kell meghatározni az oldalaink esetében, hogy milyen módon szeretnénk a tartalmakat megjeleníteni. Már említettük, hogy ezt stílusok használatával tehetjük meg.

A formázásokat többféle módon is hozzárendelhetjük a tartalomhoz.

1. Egyrészt használhatunk szövegközi stílusokat,
2. Másrészt használhatunk belső stíluslapokat
3. Harmadrészt csatolhatunk a dokumentumhoz külső stíluslapot is.

|  |  |
| --- | --- |
| ! | Ha nem készítetted el az előző gyakorlatot, akkor töltsd le 3.zip állományt. Itt megtalálható az előző feladatban elkészített forrásfájl. A továbbiakban ezzel dolgozunk tovább. (foxterrier.html) |

###### 1.Szövegközi stílusok

A szövegközi stílus esetében a formázási utasításokat közvetlenül a HTML oldalunk formázandó cimkéjének attribútumaként kell megadnunk.

Például ha meg akarjuk határozni a <h2> Méretei</h2> elem színét, akkor ezt a sort az alábbiak szerint kell módosítani:

<h2 style="color:green;">Méretei </h2>

A style attribútumot bármelyik címkén belül használhatjuk.

Az egyik probléma ezzel, hogy ha mi azt akarjuk, hogy minden második szintű címsorunk zöld legyen, akkor az összes ilyen cimkéhez be kell gépelni (vagy másolni) ezt az utasítást. De mi van akkor, ha meggondoljuk magunkat, és mégsem zöldet szeretnénk, hanem pirosat? Vagy ha újabb másodikszintű címsort illesztek be? Biztos emlékezni fogok arra, hogy milyen stílusjegyeket kell itt beállítani? Ugye érezhető, hogy nem ez az igazi megoldás.

|  |  |
| --- | --- |
| ! | A szövegközi stílusokat akkor érdemes használni, ha egy stílus csak egyszer, vagy nagyon kevésszer fordul elő a weboldalon, és ezért nem akarjuk külön kiírni a külső stíluslapban. |

Ha mondjuk alkalmaztuk a fenti egyszerű stíluslapot az oldalunkra, és minden címsor piros lett, de mi egyetlen címsornál azt szeretnénk, ha mégis zöld igazított lenne, akkor ezt szövegközi stílust használatával adjuk meg a kívánt címsornál (a szövegközi stílus felülbírálja az összes többi stílust, erről később szó lesz)

###### 2. Belső stíluslapok

Olyan stílusok megadásakor érdemes használni ezt a módszert, aminél tudjuk, hogy a weboldalunk csak egyetlen lapjához fogjuk felhasználni. A stílusdefiníciót A <head>...</head> részbe kell beírni a stíluslapot a <style>...</style> címkék közé.

<head>

<style type="text/css">

h2 {

    color: red;

}

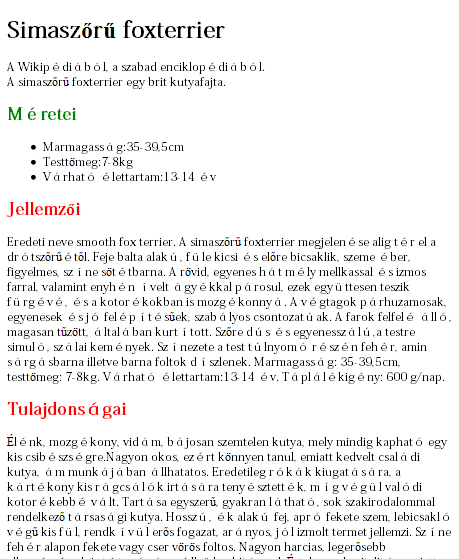
</style>

</head>

Ezzel egyszerre minden h2-es szintű címsorunk szövegszínét Pirosra változtattuk. Mivel az előző példában elhelyeztünk egy szövegközi stílust a „Méretei” cím esetében, ezért ez a szöveg zöld lett.

Figyeljük meg az utasítás szintaktikáját! A stiluslapon meg kell adni azt a cimkét, amit formázni szeretnénk. a cimkéhez tartozó formázási utasításokat { } jelek közé kell írni. Minden utasítást új sorban kell kezdeni. Az utasítás felépítése: Utasítás:érték. A sorok végén kötelező a pontosvessző (;) használata.

Most így néz ki az oldalunk:



###### 3. Külső stíluslapok

Most még ugyan csak egyetlen oldalból áll a weblapunk, de egy tisztességes weblap több 100 oldalból is állhat. Ezek mindegyike ugyanúgy kell, hogy kinézzen, tehát minden egyes oldalhoz hozzá kellene rendelni ugyanazokat a formázási utasításokat. Ezért létrehozunk egy új állományt, egy stíluslapot, mely tartalmazni fogja az összes formázási utasítást. Az állomány kiterjesztése .CSS lesz.

|  |  |
| --- | --- |
| ! | Hozzunk most létre egy új állományt a Notepad++ programmal. Mentsük el a munkamappánkba foxterrier.html mellé stilus.css néven. Állítsuk be az UTF-8-as kódolást, és mentsünk újra a fájlt. |

Az elkészült stíluslap akárhány HTML oldalhoz használható egyidejűleg. Ehhez azonban össze kell kötni a HTML fájlt és a stíluslapot. Természetesen egy cimkével, mégpedig a HTML kód fejrészében elhelyezett <link> címke segítségével. Nyissuk meg a foxterrier.html állományunkat és a fejrészt egészítsük ki az alábbi elemmel. Figyeljük meg, hogy ez páratlan címke, ügyeljünk a helyes lezárásra! Majd töröljük a korábban beírt belső stílusdefiníciót. (Ami a style között van)

<head>

.

.

.

<link href="stilus.css" rel="stylesheet" type="text/css" >

</head>

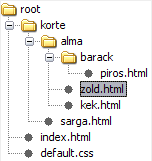
A rel és a type attribútum értéke stíluslap beágyazása esetén mindig a fenti érték.

A href utasításban elhelyezett érték a css fájl elérési útvonala. Mutathat akár az interneten elhelyezett állományra is, ilyenkor a fájl webcímét kell megadni. Leggyakoribb azonban, hogy a stíluslap ugyanabban a munkakönyvtár struktúrában van, mint a formázandó HTML állomány. Ilyenkor a stíluslap relatív elérési útvonalát kell a href attribútum értékeként megadni. Ha például a gyökérkönyvtárban létrehoztunk volna egy új mappát „stiluslapok” néven, és ebbe mentettük volna a stíluslapunkat, akkor href="stilulapok/stilus.css" – t kellett volna írnunk. Mivel most a stíluslap ugynabban könyvtárban található, mint a foxterrier.html, ezért elég csak a stíluslap nevét beírni az elérési útvonalba.

###### Mi is az a relatív elérési útvonal?

Kétféle hivatkozás létezik: relatív és abszolút. Abszolút elérési útnak nevezzük, ha a teljes internetcímet pl: <http://www.webhelycime.hu/> megadjuk, vagy a számítógépünk gyökérkönyvtárárólkezdve a teljes mapparendszert felsoroljuk addig a mappáig, ahol a fájlunk található. Ilyenkor a mappák neveit /-jellel kell elválasztani.

Relatív hivatkozás olyan elérési út, ahol a kiindulási pont annak a fájlnak a helye, ahol éppen állunk. Tekintsük az alábbi könyvtárszerkezetet egy webszerveren belül:



A root a webszervernek az a könyvtára, amibe feltölthetjük az elkészített weboldalainkat, képeinket, stíluslapjainkat stb. Ebbe kell feltölteni az index.html nevű weboldalt, amit a böngésző automatikusan keresni fog, ha beírja valaki a domain nevünket a böngészőbe, de erről majd később.

Relatív elérési útvonalak:

1. A kek.html fájl relatív elérési útvonala: „kek.html”
2. A piros.html fájl relatív elérési útvonala: „barack/piros.html”
3. A sarga.html fájl relatív elérési útvonala: „../sarga.html”

Használhatunk kiszolgálón belüli abszolút hivatkozást is:

1. sarga.html: /korte/sarga.html
2. zold.html: /korte/alma/zold.html
3. piros.html: /korte/alma/barack/piros.html

A / jelenti a root mappát, onnan kiindulva tudjuk meghatározni az elérési útvonalat.

Saját webhelyünkön belül mindig így hivatkozzunk a saját fájlainkra, mert ha később átvisszük más domain név alá az egész webhelyet, akkor nem kell átírni az abszolút hivatkozásokat. Tehát:

|  |  |
| --- | --- |
| ! | A href attribútm értéke a html állományhoz viszonyítva a css stíluslap relatív elérési útját tartalmazza. |

Most a stilus.css állományba írjuk be:

@charset "utf-8";

h2 {

color: red;

}

**Mentsük mindkét állományunkat!!!** Ha most megnézzük a munkánkat a böngészőben, semmiféle változást nem fogunk látni, ha jól dolgoztunk, mégis lényegesen több lehetőséget rejt magában a rendszerünk, mint elsőre gondolnánk!

### A rangsor

A CSS a rangsorolt stíluslapokat jelenti magyarul. Az elnevezés abból származik, hogy egyetlen HTML oldalhoz egyszerre több külső és belső stíluslapot is hozzárendelhetünk, és közben még szövegközi stílusokat is alkalmazhatunk. Emlékeztetőül:

Tehát egy oldalhoz egyszerre több külső stíluslapot is csatolhatunk, és közben még használhatunk belső és szövegközi stílusokat is ugyanabban az oldalban. Ez akkor lesz érdekes, ha a különböző stíluslapok ugyanarra a címkére írnak elő ugyanolyan tulajdonságokat. Ekkor a böngészőnek el kell tudnia dönteni, hogy melyik stílus végezze a formázást. Ezzel kapcsolatban általános szabályként elmondható: **Mindig a formázandó HTML kódhoz legközelebbi stílus előírás végzi a formázást.**

Tegyük fel, hogy egy oldalhoz 3 külső stíluslapot csatolunk, és használunk belső és szövegközi stílusokat is. Ha ezek a különböző stíluslapok ugyanarra a címkére írnak elő formázást, akkor végül az adott címkéhez legközelebbi stílus fog érvényre jutni. Ez azt jelenti, hogy a böngésző elkezd a HTML kódban fentről lefelé haladni, és mindig alkalmazza a talált stílusokat a megfelelő címkékre. Először olyanná fogja formázni az oldalt, ahogyan a forrásban legelsőnek megadott külső stíluslapban találja. Utána folytatja a formázást a forrásban másodikként megadott stíluslappal, ez a stíluslap át fogja írni azoknak a címkéknek a tulajdonságait, amiket az előtte lévő első stíluslap is megadott. A böngésző ugyanígy jár el a harmadikként megadott külső stíluslap, és a belső stíluslap esetén is. Végül, egy címkére megadott szövegközi stílus felül fogja írni az összes előtte lévő stílust.

A formázásokra azonban még két másik stíluslap is hatással van: a böngésző alapértelmezett stíluslapja, és a felhasználó által beállított stílusok.

Az alábbi ábra az imént elmondottakat mutatja be. Az ábra olyan sorrendben jelzi a stílusokat, amilyen sorrendben a kódban előfordulnak, és amilyen közel állnak a formázandó címékhez. A lentebb, azaz a formázandó címkéhez közelebb álló stílus felülbírálja a hátrébb állót:

Megjegyzés: A felhasználó módosíthatja a böngésző beállításait, készíthet akár saját stíluslapot is, de ezzel mi most itt nem foglalkozunk. Az ábra azt az esetet szemlélteti, amikor a böngésző alapértelmezett beállításait használjuk.

|  |  |
| --- | --- |
| ! | További információkat a dokumentumfáról, és a formázási hierarchiáról a 2. számú mellékletben olvashatunk. |

### Formázás külső stíluslapokkal

Vizsgáljuk most meg az új stilus.css állományba írt kódunkat.

A stíluslapfájl legelső sorába a karakterkódolásra vonatkozó információt írtunk. Itt is az UTF‑8 kódolást ajánlom, mint a HTML esetében.

@charset "utf-8";

Ezután az egész stíluslap nem más, mint kijelölők (selector) és meghatározásblokkok (declaration) felsorolása. A meghatározás két részből áll: tulajdonságból (property) és értékből (value).

h2 {

color: red;

}

Ebben az egyszerű stíluslapban a h2 címke a kijelölő (selector), a color: red; sor pedig a meghatározás, ezen belül a color a tulajdonság (property), a red pedig ennek a tulajdonságnak az értéke (value). Általánosan:

@charset "utf-8";

kijelölő {

    tulajdonság1: érték1;

    tulajdonság2: érték2;

    tulajdonság3: érték3;

    ...}

A tulajdonság megnevezése után kettőspontot ( : ) kell írnunk, majd következik az érték, ami után pontosvesszőt ( ; ) kell tennünk. Ha egy tulajdonsághoz több érték tartozik, akkor az értékeket csak szóközzel választjuk el egymástól.

A kijelölő egy címkét jelöl ki a HTML dokumentumban, és erre a kijelölt címkére vonatkozik a formázása.

###### Csoportosítás

Egy meghatározásblokkot egynél több címkére is alkalmazhatjuk, ha vesszővel elválasztva felsoroljuk a formázandó címkéket.

h1, h2 {

    background: blue;

}

###### CSS kommentek

HTML megjegyzést vagy kommentet a <!-- és a --> jelek közé írhatunk. CSS-ben pedig a /\* és a \*/ jelek közé. Egy megjegyzés több sort is átfoghat. A megjegyzéseket nem veszi figyelembe a böngésző, semmilyen hatással nincsenek a kinézetre, két dolgot szolgálnak:

* Hosszabb stíluslapokhoz érdemes megjegyzéseket írni a megfelelő helyre, hogy ha később elővesszük, akkor tudjuk, mit miért írtunk. Ez főleg akkor fontos, ha nem csak magunknak készítjük a stíluslapot, hanem másnak is meg kell értenie.
* Teszteléskor hasznosnak fogjuk találni a megjegyzést. Ideiglenesen megjegyzések közé tehetünk kódrészleteket, amikre éppen nincs szükség.

@charset "utf-8";

/\*Ez egy megjegyzés.\*/

h1, h2, h3 {

    color: red; /\*Ez egy megjegyzés.\*/

}/\*Ez

egy

többsoros megjegyzés.\*/

###### Színek definiálása a weboldalakon

A weboldal megjelenítésére képes eszközök (pl. monitor, nyomtató) a színeket három alapszínből, a piros-zöld-kék kombinációból keveri ki. Mintha lenne három festékes vödrünk, és ezeket kellene összeöntenünk egy bizonyos arányban ahhoz, hogy a kívánt színt kapjuk meg. Ezeknek az alapszíneknek az angol nevei rendre red, green, blue, ezért az ilyen színeket az angol neveikből rövidítve RGB-színeknek nevezzük.

Azt, hogy melyik színből mennyit akarunk felhasználni, háromféleképpen adhatjuk meg:

1. Hexadecimális, azaz 16-os számrendszerbeli számokkal, pl. #1E0EE2 -ez egy sötétkék szín, #F0F2AC -ez pedig egy világossárga. Az érték előtt kettőskeresztet használunk, majd az első kettő szám a piros, a középső kettő a zöld, az utolsó kettő pedig a kék mennyiségét jelzi 16-os számrendszerben. Szerencsére ezeket a számokkal nem kell vesződnünk, mert a kódszerkesztő programunk fel fog ajánlani egy színpalettát, és a kiválasztott szín kódját be fogja szúrni helyettünk.
2. A második módszer az RGB értékek százalékos formában történő megadása. Ekkor azt mondjuk meg, hogy a három alapszínből százalékosan milyen arányban forduljon elő a színünkben. Pl. color: rgb(30%,20%,50%);
3. Hasonló módszer a harmdik, csak most nem százalékosan, hanem egy 0 és 255 közti tízes számrendszerbeli számmal adjuk meg az RGB színek arányát. Pl. color: rgb(34,55,201);
4. Használhatunk előre definiált angol színneveket is. Ezeket a W3C határozta meg a szabványban, ezek nevei alább láthatóak: maroon; red; orange; yellow; olive; purple; fuchsia; white; lime; green; navy; blue; aqua; teal; black; silver; gray

Mivel weboldalunkat nagyon sokféle operációs rendszerrel, monitorral, grafikus kártyával stb. nézhetik meg a látogatók, ezért nem lehetünk 100%-ig biztosak benne, hogy az általunk kikevert színárnyalatot minden monitor képes lesz megjeleníteni. Van viszont 216 darab színárnyalat, amelyeket még az olyan monitorok is megjelenítek, amelyek csak 256-féle színt képesek. Ezeket **webtűrő színeknek** nevezzük (browser-safe colors).

A továbbiakban a hexadecimális írásmódot fogom használni, mert ez a legrövidebb írásmód.

Az alábbi példák mindegyike a fehér színt állítja elő a h1-es címsorra: (Persze valós példáknál csak egyszer adjunk meg a color értékét!)

h1 {

    color: #FFFFFF;

    color: rgb(100%,100%,100%);

    color: rgb(255,255,255);

    color: white;

}

##### Szövegek formázásának lehetőségei

Bármely szöveget tartalmazó cimkéhez rendelhetünk formázási utasításokat. Amennyiben a teljes oldalunk összes szövegére szeretnénk egy szövegformázást alkalmazni, úgy azokat a formázási utasításokat a body cimkéhez kell rendelnünk.

Az alábbi táblázat összefoglalja a szövegekre alkalmazható formázási lehetőségeket:

|  |  |  |
| --- | --- | --- |
| Tulajdonság | Leírás | Lehetséges értékek |
| color | Szöveg színének megadása. rgb kóddal  hexadecimális kóddal  szín nevével | color-rgb  color-hex  color-name |
| text-align | Szövegigazítás: bal,  jobb,  közép,  sorkizárt. | left  right  center  justify |
| text-decoration | Aláhúzást,  felülhúzást,  áthúzást, vagy  villogást állít be a szövegre. | none  underline  overline  line-through  blink |
| text-indent | Bekezdés első sorának beljebb kezdése. A length értékét pixelben, vagy képpontban kell megadni, vagy százalékosan | length  % |
| line-height | Sorközök megadására szolgál. | normal  number  length  % |
| letter-spacing | A karakterek közti szóköz nagyságát szabályozza. | normal  length |
| word-spacing | A szavak közti szóköz nagyságát szabályozza. | normal  length |
| text-transform | Szöveget  kisbetűssé,  nagybetűssé, vagy  minden szót nagy kezdőbetűssé változtat. | none  capitalize  uppercase  lowercase |
| font-family | A használandó betűcsaládról ad prioritásos listát: Ha az első megadott font nem található, akkor próbálkozik a másodikkal és így tovább. | Betűcsalád neve, pl: Arial, "Times New Roman", Verdana;  Vagy: általános típus:  serif  sans-serif  cursive  fantasy  monospace |
| font-style | Dőlt betűs szöveg megjelenítését teszi lehetővé. | normal  italic  oblique |
| font-size | Szöveg méretét lehet megadni. | pt  px  smaller  larger  %  em |
| A font-size tulajdonság értékéhez csak ezeket az értékeket ajánlott megadni: %, em, smaller, larger. Ugyanis ezek az értékek a szülő elem betűméretéhez képest relatívan adják meg a betűméretet. Ha abszolút értékeket használunk, akkor az Internet Explorer 6 felhasználók nehezen fogják tudni átméretezni a szöveget, mert az IE 6 csak kis skálán engedi változtatni a betűméretet. Továbbá az abszolút betűméret eltérhet a látogató által megszokottól. Abszolút méretet csak nyomtatási stíluslaphoz használjunk, ezek a következők: xx-small, x-small, small, medium, large, x-large, xx-large, pt, px.  A 2em és a 200% egyaránt azt jelenti, hogy az elem fontmérete legyen kétszerese a szülő elem fontméretének. | | |
| font-weight | Szöveg vastagságát adja meg. | normal  bold |
| font-variant | Kiskapitális szöveg megadása. | normal  small-caps |
| A kiskapitális szöveg olyan szöveg, ami először csupa nagybetűs lesz, aztán az eredetileg is nagybetűk egy kicsivel nagyobb méretűek leszenek, mint a többi nagybetű. | | |
| font | Rövidítő tulajdonság az összes eddigi tulajdonság beállításához, a jobb oldali sorrendben. | font-style  font-variant  font-weight  font-size/line-height  font-family |

Nézzük most a mi konkrét oldalunk formázását. Először állítsuk be a weboldalunkon használni kívánt betűcsaládokat. Képernyőn való olvasáshoz érdemes talp nélküli betűtípust választani. Ezek ugyanis tiszta, könnyen olvasható betűképet adnak. A betűtípust mindig illesszük az oldal hangulatához. Mi itt most Verdana betűtípust választunk. Mivel egyáltalán nem bizonyos, hogy ez a betűtípus minden számítógépen telepítve van, ezért második típusként adjuk meg az Arial típust. Ha véletlenül ilyen sem lenne a gépen, akkor utasítsuk arra a böngészőt, hogy a meglévő telepített fontkészletből válaszzon valamilyen talp nélküli típust.

body {

font-family: Verdana, Arial, sans-serif;

}

Állítsuk még be az alapértelmezett betűszínt és betűméretet is.

body {

font-family: Verdana, Arial, sans-serif;

color:#4c1f22; /\*sötétbarna szín\*/

font-size:12pt;

}

A címeket alakítsuk nagybetűssé, legyenek vastagon szedve

h1, h2 { /\*Egyszerre adjuk meg a két stílus azonos jegyeit\*/

text-transform:uppercase;

font-weight: bold;

}

Az első szintű címsorban a betűk közti távolság legyen 5pt-os, a betűméret legye 16 pt.

h1 {

letter-spacing: 5pt;

font-size: 16pt;

}

A 2. szintű címsorok betűméretét állítsuk 14pt-ra.

h2 {

font-size: 14pt;

}

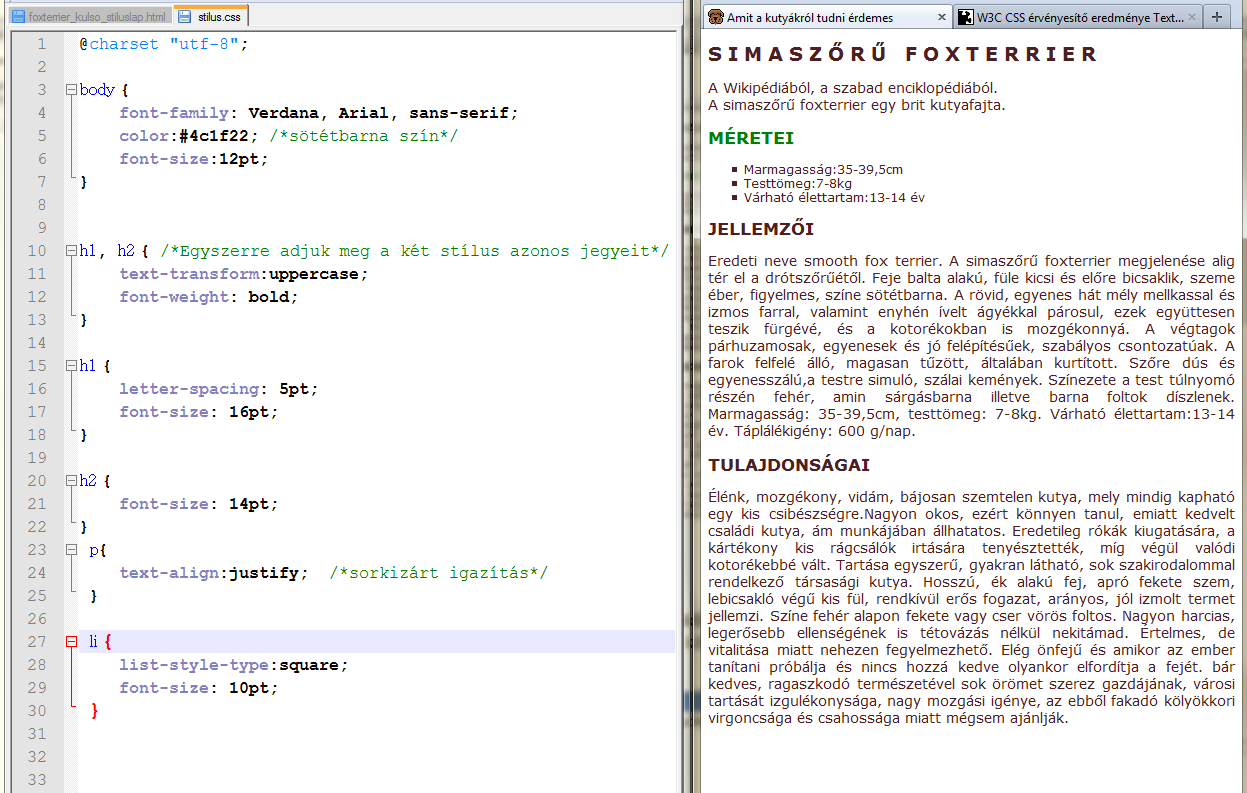
A bekezdések legyenek sorkizárt igaztásuak.

p{

text-align:justify; /\*sorkizárt igazítás\*/

}

A bal oldalon azt látjuk, hogy néz ki a stíluslapunk, a jobb oldalon pedig azt, hogy jelenik meg a HTML oldalunk a böngészőben. Figyeljük meg, hogy a CSS szerkesztése során a cimkéket, tulajdonságokat és értékeket más-más színnel jelöli a Notepad++. Ez segíthet abban, hogy az esetleges elírásokat felimerjük.



A „ Méretei” szöveg azért zöld, mert a HTML kódban bennehagytuk a sorközi formázást ennél az elemnél.

### Listák formázása stíluslappal

A listák formázásakor természetesen használhatjuk az eddig megismert karakter és bekezdés formázási utasításokat. Ez a táblázat most csak kifejezetten a listákra vonatkozó tulajdonságokat mutatjuk be.

|  |  |  |
| --- | --- | --- |
| Tulajdonság | Leírás | Lehetséges értékek |
| list-style-type | A listaelemek előtti kis jel megadására szolgál.: pötty  kör  négyzet  nincs  szám (számozott lista)  a,b,c  A,B,C  i,ii,iii, iv  I, II, III, IV | disc  circle  square  none  decimal  lower-alpha  upper-alpha  lower-roman  upper-roman |
| list-style-image | Kép beállítása a listaelemek jelzésére. | none  url |
| list-style-position | A listaelmek előtti kis jel vagy kép helyzetének megadására szolgál. | outside  inside |
| list-style | Rövidítő tulajdonság. | list-style-type  list-style-position  list-style-image |
| A list-style rövidítő tulajdonság használatával az összes eddigi tulajdonságot kiválthatjuk. | | |

A mi példánkban a felsorolásunk jelének válasszuk a négyzetet. A felsorolás betűméretét állítsuk be 10 pt-re.

li {

list-style-type:square;

font-size: 10pt;

}

A listák formázásával kapcsolatos további formázásokat majd a menük készítésénél részletesebben is tárgyaljuk majd.

### Összegzés

Ebben a gyakorlatban megismerkedtünk a stíluslapok használatával, a karakter és bekezdésformázási lehetőségekkel.

A CSS állományokat, akárcsak a HTML kódokat validálni lehet az alábbi weboldalon: <http://jigsaw.w3.org/css-validator/>, Töltsük fel ide az elmentett CSS állományunkat, hasonlóan, mint ahogy az a HTML oldalunk validálásakor már tanultuk. Javítsuk az esetleges hibákat, majd a hibátlan kód esetén kapott kódot másoljuk a HTML oldalunk body tagjei közé!

A megoldások a 3.zip fájl „mo” mappájában megtalálhatóak.

### Feladat

Az előző gyakorlat feladatában elkészített recept.html állományt formázzuk meg az alábbi mintának megfelelően. A betűcsalád legyen Calibri. A betűméret 1.1em. A címek mérete legyen 1.3em, illetve 1.2em. a listaelemek legyenek dölten szedve. A Definíciós lista definíciói (Porcukor, Önkelő liszt) legyenek vastagon szedve.

Validáltassuk az oldalt! A CSS-t is!

