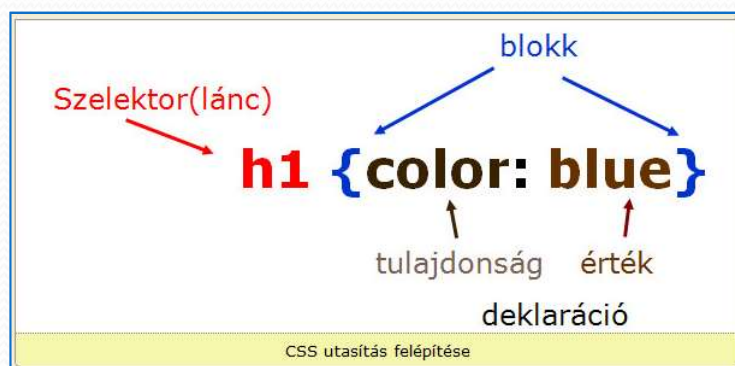


CSS3

- **A CSS utasítás**

A CSS-utasítások (szabályok) segítségével beállíthatjuk a weblapok HTML-elemeinek vizuális megjelenését. Egy CSS-utasítás két részből áll:

- A szelektor(lánc) tartalmazza a formázandó HTML-elem(ek)et.
- A deklaráció végzi el a szelektorban meghatározott elemek formázását.



- *A CSS-utasítás szintaxisa:*

- A szabály egy HTML - szelektorról vagy egy szelektorláncról kezdődik, amelynek stílusát szeretnénk beállítani.
- A szelektort egy kapcsos zárójelpár által határolt deklarációs blokk követ, amelyben tetszőleges számú CSS-deklaráció szerepel.
- Egy CSS-deklaráció két részből áll:
 - Egy tulajdonságból, amelynek értéket szeretnénk beállítani.
 - Egy értékből, amit a tulajdonságnak adunk értékül.
- Egy tulajdonság-érték párost kettőspont választ el.
- Minden CSS-deklarációt egy pontosvessző zár le.

CSS3

- *A CSS-utasítások csoportosítása:*

Ha több szelektorra is szeretnénk ugyanazt a CSS-szabályt alkalmazni, akkor szelektor-csoportot kell létrehoznunk. Azaz a deklarációs blokk előtt vesszővel elválasztva felsoroljuk a kívánt szelektorokat, így az összes szelektorra vonatkozik majd a stílusdeklaráció.

```
h1 {font-family: verdana;}  
h2 {font-family: verdana;}  
h3 {font-family: verdana;}
```



```
h1, h2, h3 {font-family: verdana;}
```

- *Több tulajdonság megadása ugyanazon szelektorra:*

```
h1 {font-family: helvetica;}  
h1 {font-size: 12pt;}  
h1 {font-style: normal;}
```



```
h1 {  
    font-family: helvetica;  
    font-size: 12pt;  
    font-style: normal;  
}
```

- *Kompakt (rövidített) megadási forma (shorthand property):*

Ha több szelektorra is szeretnénk ugyanazt a CSS-szabályt alkalmazni, akkor szelektor-csoportot kell létrehoznunk. Azaz a deklarációs blokk előtt vesszővel elválasztva felsoroljuk a kívánt szelektorokat, így az összes szelektorra vonatkozik majd a stílusdeklaráció.

```
h1 {font-weight: bold;}  
h1 {font-size: 12pt;}  
h1 {font-family: helvetica;}
```



```
h1 {font: bold 12pt helvetica;}
```


CSS3

• Stíluslap csatolási módok

- **Beágyazott, inline megadás :**

```
<p style="color: green;">Az egész bekezdés zöld</p>
```

Jellemzők: nehézkes módosítás, nem lehet megfelelően kihasználni a CSS azon lehetőségeit pl.: hogy különböző média típusokra (pl. képernyő, mobil eszköz, nyomtatási nézet, képernyőolvasó) más-más stílust rendelhessünk.

Szintaxis:

- `<tag style="paraméter:érték;paraméter:érték"></tag>`
- Az attribútum értékét, azaz a CSS-deklarációkat dupla idézőjelek között adjuk meg.
- Egy CSS deklaráció egy tulajdonság-érték párból áll, amit egy kettőspont választ el egymástól. Minden deklaráció végén pontosvessző áll.
- **A formázni kívánt elemnél használjuk a style attribútumot. A szabály az adott elemre, illetve azok leszármazottaira lesz érvényes.**

```
<style type="text/css">
  h1 {color:blue}
</style>
```

Jellemzők : A CSS-szabályainkat elhelyezhetjük a style elem által határolt blokkban is. Több lapból áll site esetén duplikációkat okozhat.

Szintaxis:

- helyezzük el a HTML-kódunk fej `<head>` részében a `<style type="text/css"></style>` elemeket.
- a style elemben levő `type="text/css"` attribútum elárulja a böngészőnek, hogy az elem a CSS szabványnak megfelelő szöveges stílusszabályokat tartalmaz.
- A CSS-szabályokat a style elemen belül helyezhetjük el. Ezek pontos szerkezetét később ismertetjük.
- A style elemet úgy is használhatjuk, hogy kiegészítjük a media paraméterrel is, ami azt fogja befolyásolni, hogy a stílusok milyen médiatípusokra vonatkozzanak

CSS3

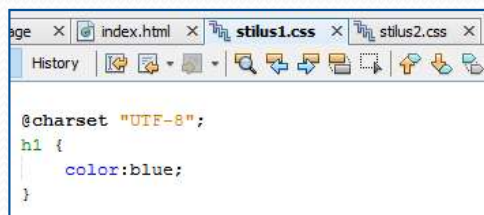
```
<!DOCTYPE html>
...5 lines
<html lang="hu">
  <head>
    <meta charset="utf-8">
    <title>Stíluslap csatolási módok</title>
    <!-- külső stíluslap belinkelése -->
    <link rel="stylesheet" type="text/css" href="css/stilus1.css" title="alap">
    <!-- lapon belüli definíció -->
    <style type="text/css">
      @import url("css/stilus2.css");
      @media print {
        body {
          font-family: 'MV Boli';
        }
      }
    </style>
  </head>
  <body>
    <h1>A címsor1 kék</h1>
    <h2>A címsor2 piros</h2>
    <!-- Az alábbi sorban egy beágyazott megadást látunk -->
    <p style="color: green;">Az egész bekezdés zöld</p>
  </body>
</html>
```

Képernyőn:

A címsor1 kék

A címsor2 piros

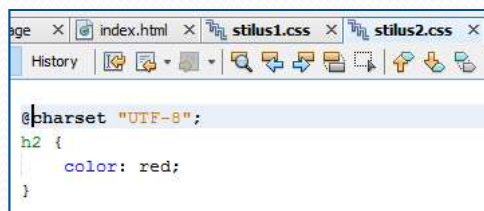
Az egész bekezdés zöld



index.html x stilus1.css x stilus2.css x

History

```
@charset "UTF-8";
h1 {
  color: blue;
}
```



index.html x stilus1.css x stilus2.css x

History

```
@charset "UTF-8";
h2 {
  color: red;
}
```

Nyomtatásban:



CSS3

- **Stíluslap csatolási módok**

- **Hivatkozás külső stíluslapra, a link elem segítségével:**

```
<style type="text/css">  
  <link rel="stylesheet" type="text/css" href="pelda.css">  
</style>
```

Jellemzők: A külső fájl, ami a CSS szabályokat tartalmazza egy egyszerű szöveges fájl, *css* kiterjesztéssel. Itt már valóban elválasztódik a tartalom a megjelenéstől. A gyakorlatban ez egy széleskörűen használt módszer stílus hozzáadására.

Szintaxis:

- Helyezzük el a `<link>` elemet a HTML lapunk fejrészába.
- a `rel="stylesheet"` attribútum értékével tudatjuk a böngészőnek, hogy az elem egy stíluslapot fog belinkelni
- a `href="stíluslap.css"` a CSS-szabályokat tartalmazó stíluslap relatív címét határozza meg, a HTML-fájllhoz képpe
- a `type="text/css"` megmondja a böngészőnek, hogy a belinkelt stíluslap a CSS szabványnak megfelelő
- itt is megadhatjuk a *media* paramétert, ami megadja, hogy a stílusok mely médiatípusokra vonatkozzanak.
- több `<link>` elemmel több stíluslapot is belinkelhetünk – ez a sebességet csökkentheti

CSS3

• Stíluslap csatolási módok

• Stíluslap beimportálása

Jellemzők: Az `@import` kulcsszóval külső stíluslapot importálhatunk be HTML-fájlunkba. Ez gyakorlatilag ugyanazt csinálja, mint a link tag, csak ez nem HTML-, hanem CSS-utasítás. Így egy külső stíluslapállomány is tartalmazhat más állományokra vonatkozó importálási utasítást.

Szintaxis:

- Helyezzük el a `<style type="text/css"> </style>` elemeket és attribútumait a HTML lapunk fejrészébe.
- Helyezzük el a style elemek közé az importálási szabályt `@import url(stiluslap.css);`, ami meghatározza a külső CSS-fájl relatív útvonalát a HTML-fájlhoz képest.
 - A `@import url(stiluslap.css);` megadás helyett használhatjuk a rövidebb megadást is: `@import "stiluslap.css";`
 - Fontos, hogy az importálási szabály végéről ne hagyjuk le a pontosvesszőt!
 - **A style elemekbe az importálási szabályon kívül más szabályok is beilleszthetők, azonban ilyenkor az import szabálynak meg kell előznie minden más CSS-szabályt!!!**
- Az `@import url('stiluslap.css');` (vagy rövidebben `@import 'stiluslap.css';`) szabály segítségével CSS fájlokat is beimportálhatunk más CSS-fájlokba. Ilyenkor fontos, hogy a CSS-fájl legelső sora tartalmazza az import szabályt.
- Az importálás történhet úgy is, hogy megadjuk, hogy milyen médiatípusra vonatkozik a beimportált állomány:
 - `@import url("nyomtat") print;` ← Nyomtatás médiára szánt stílusállomány importálása.
 - `@import url("proj.css") projection, tv;` ← Kivetítőre, TV-re szánt stíluslap importálása.
 - `@import url("keskeny.css") handheld and (max-width: 400px);` ← A kézi és a maximum 400px vízszintes méretű eszközökre készült stíluslap importálása.
- A fenti szabály alkalmazásával fontos lépést tehetünk majd a reszponzív arculat felé, ahol minden felbontásban az optimális stíluslapot tudjuk felkínálni.

CSS3

• A CSS-szabályok értékei, mértékegységek

A CSS-tulajdonságoknak adható értékek két nagy csoportba oszthatók: vannak előre definiált kulcsszavak, vannak a bizonyos szabályoknak eleget tevő kifejezések.

- **Hosszúságértékek** : A CSS-szabályok definícióiban a tulajdonságok egy részének hosszúságértékeket kell megadnunk értékül. Ezek két nagy csoportra oszthatók: abszolút és relatív hosszúságértékekre
 - Abszolút hosszúságértékek: rendszerint hosszúságmértékben vannak megadva, azaz minden értékhez hozzátartozik az a hosszmérték, amiben értelmezendő
 - in: inch (1 in = 96px = 2.54cm (96dpi))
 - cm: centiméter
 - mm: millimeter
 - pt: pont (1/72 inch)
 - pc: pica (12 pont)
- **Relatív hosszúságértékek** : A relatív hosszúságértékek önmagukban sosem fejeznek ki semmit, mivel *értékük más értékektől függ*. A segítségükkel megváltoztathatjuk egy tulajdonság aktuális vagy megöröklött értékét. fajtái:
 - **em**: Ekkor az em elé írt szám jelképezi azt a szorzót ahány szorosára kívánjuk változtatni az aktuális (vagy megöröklött) font magasságának értéket.
 - **ex**: Relatív érték az adott betűtípus kis x betűjének magasságához képest. (A böngészők általában az 1em = 2ex szabály szerint számolnak, pedig ez tipográfiailag helytelen: az x betű magassága rendszerint nem egyenlő a választott betűméret felével.)
 - **px (pixel)**: Mivel a képpixelek mérete nagyban függ a kijelző tulajdonságaitól, így a pixelben megadott méretek, távolságok nem garantálnak egységes megjelenést. (nyomtatás 1px = nyomtatott pont)
 - **%**: A százalékos értékek relatívak, azaz értékük más értékekből számolódik ki. Ez lehet alapértelmezett, vagy öröklött érték. Megadása egy számérték és egy azt követő százalék jel segítségével történik.
 - **ch**: Relatív érték az adott betűtípus kis 0 (nulla) betűjének magasságához képest.
 - **rem**: Relatív érték az gyöker elem betű magasságához képest.
 - **vw**: Relatív érték az a böngésző ablakméret (viewport) szélességének 1%-ához képest.
 - **vh**: Relatív érték az a böngésző ablakméret (viewport) magasságának 1%-ához képest.
 - **vmin**: Relatív érték az a böngésző ablakméret (viewport) kisebb méretének 1%-ához képest.
 - **vmax**: Relatív érték az a böngésző ablakméret (viewport) nagyobb méretének 1%-ához képest.

CSS3

• A CSS-szabályok értékei, mértékegységek

- **URL-k megadása:** URL-ek segítségével külső objektumokra hivatkozhatunk, ami lehet pl.: CSS-fájl, JavaScript-fájl, HTML-fájl, videó, kép ... Egy ilyen érték az **url** kulcsszóval kezdődik, amit zárójelek között, a hivatkozott objektum pontos relatív vagy abszolút címe követ.

```
body {  
    background-image: url('kepek/hatter.jpg');  
}  
li {  
    list-style: url('http://www.vala.hol/kep.png') disc  
}
```

```
<p style="color:BlueViolet"></p>  
<p style="color:#f69"></p>  
<p style="color:#deb887"></p>  
<p style="color:rgb(51,71,28);"></p>  
<p style="color:rgb(50%,70%,30%);"></p>  
<p style="color:hsl(300,50%,70%)"></p>
```

- **Színek megadása :** A szöveg színét a color, az elem háttérszínét a background-color paraméterrel adhatjuk meg. A színt meghatározó érték sokféle lehet, megadhatjuk kulcsszóval, de kódokkal is. fajtái:
 - használhatunk **előre definiált színeket**, ezeket az angol nevükkel adhatjuk meg
 - **A színek megadhatjuk hexadecimális RGB kódjukkal**
A számítógépes kijelzők a vörös, kék és zöld színek keverékeként állítják elő a színeket. Az RGB kód használatakor a színeket e komponensek mennyiségével adhatjuk meg. A hexadecimális RGB kódok # jellel kezdődnek, amit a vörös zöld és végül a kék komponensek aránya követ
 - **A színek megadhatjuk decimális RGB kódjukkal**
Ekkor az egyes komponensek értékei 0–255 közötti intervallumba esnek. A szintaxis a következő: a színt az rgb kulcsszó vezeti be, amit zárójelek között a három komponens értéke vesszővel elválasztva követ.
 - **A színek megadhatjuk százalékos RGB kódjukkal**
RGB kód megadása százalékos értékekkel is lehetséges. Ekkor az egyes komponensek értékei 0% és 100% közötti intervallumba esnek. A szintaxis a következő: a színt az rgb kulcsszó vezeti be, amit zárójelek között a három komponens százalékos értéke vesszővel elválasztva követ.
 - **A színek megadhatjuk HSL kódjukkal:** A színeket megadhatjuk azok HSL kódjukkal is, ahol a "H" a színárnyalatot (Hue), az "S" a színtelítettséget (Saturation), az "L" pedig a világosságot (Lightness) jelképezi. A színárnyalat egy 0–360 intervallumba tartozó egész szám. A telítettség és a világosság pedig egy 0% és 100% közötti százalékos érték

CSS3

- **A CSS-szabályok értékei, mértékegységek**

- **Átlátszóság:** A CSS3 egyik gyakran használt új funkciója az átlátszóság. A grafikai programokban is gyakran használunk átlátszó színeket, vagy akár rétegeket. Az átlátszóság mértékét *transparency*, az átlátszatlanságot *opacity* néven találhatjuk általában a programokban, illetve az úgynevezett alfa csatorna (rövidítve A) segítségével tudunk még átlátszóságot beállítani.

```
<p style="color:rgb(51,71,28,0.3);"></p>
<p style="color:rgb(50%,70%,30%,0.5);"></p>
<p style="color:hsl(300,50%,70%,0.5)"></p>
```

Érdemes gondoskodnunk arról, hogy az átlátszóságot nem ismerő böngészők is megjelenítsenek egy megfelelő (nem átlátszó) színt.

```
section {
    background: rgb(180,50,50); /* fallback*/
    background: rgb(180,50,50,0.3);
}
```

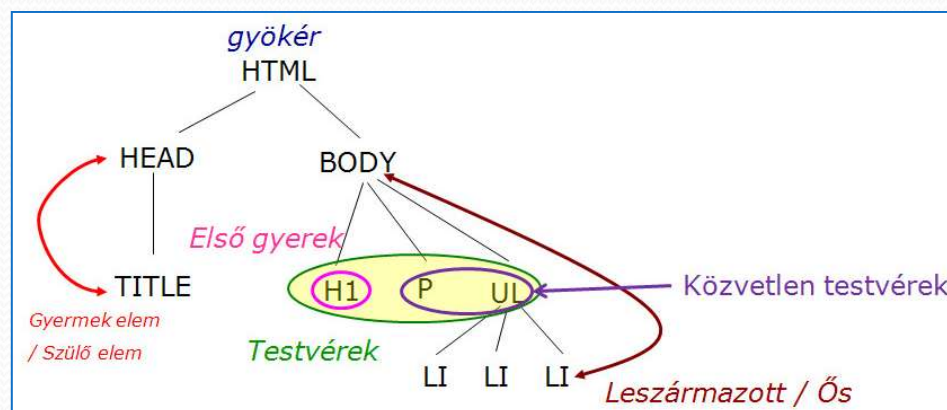
- **Szög megadása:** Több CSS3-as tulajdonságnál is szükség lehet egy szög megadására. Ezek megadási módja a következő:
 - *deg*: szöggel. Például: 120deg.
 - *rad*: radiánnal. Például: 1.5rad.
 - *grad*: gradienssel. Például: 100grad.

CSS3

• Öröklődés, kiértékelési sorrend, dokumentumfa

- **Dokumentumfa:** A HTML-fájljainkat felépítő tageket tartalmazó fa struktúra
 - *A fa gyökere a html tag, a levelei pedig beágyazott (inline), vagy olyan blokk elemek, amelyek nem tartalmazznak további tageket.*
 - *Szülőnek* nevezzük azokat az elemeket, amelyek tartalmazznak más tag-eket, amik a szülő elem gyermekei.
 - Azokat a gyermekeket, amelyek közös szülővel rendelkeznek, *testvéreknek* nevezzük.
 - Azok a testvérek, amelyek közvetlenül egymás után helyezkednek el a dokumentumban, *közvetlen testvéreknek* nevezzük.
 - Egy elem gyermekeinek a gyermekeit, annak *unokáinak* nevezzük. Fordítva pedig egy elem szülőjének a szülője, az *elem nagyszülője*.
 - A fában egy elem alatti részfa elemeit az elem *leszármazottainak*, a fa az elem felett elhelyezkedő elemeit, pedig annak *őseinek* nevezzük.
 - *Minden elemnek pontosan egy szülő eleme van, kivéve a gyökérelemet, amelynek nincs szülő eleme.*

```
<!DOCTYPE html>
<html lang="hu">
<head>
  <title>Dokumentumfa bemutatása</title>
</head>
<body>
  <h1>Címsor</h1>
  <p>Bekezdés</p>
  <ul>
    <li>Listaelem 1.</li>
    <li>Listaelem 2.</li>
    <li>Listaelem 3.</li>
  </ul>
</body>
</html>
```



CSS3

• Öröklődés, kiértékelési sorrend, dokumentumfa

• Öröklődés:

- A stíluslapok tervezésénél figyelembe kell vennünk, hogy a tulajdonságok egy része automatikusan öröklődik a szülő elemtől a gyerek (leszármazott) elemekre is.
- A teljes lapra vonatkozó, általános tulajdonságokat (pl. betűméret, betűcsalád) a *body* szelektorra érvényesen érdemes megadni, hiszen ez az elem minden más elemnek az őse!

```
body
{
  font-size: 12pt;
  font-family: Arial, Verdana, sans-serif
}
```

• Számított érték öröklése:

- Fontos tudnunk, hogy a gyermekelemek nem a szülőekben megadott relatív értékeket öröklik, hanem azok **számított értékét!**

```
body
{
  font-size: 12px;
  text-indent: 3em;
}
h1 { font-size: 25px }
```

A bekezdés első sorának behúzása = $3 * 12px = 36px$
Ez a 36px érték öröklődik tovább *behúzásként h1-re*

• Az inherit érték használata

- Ha azt szeretnénk, hogy egy elem a szülőtől örököljön olyan tulajdonság(oka)t, amely alapesetben nem öröklődik, akkor az inherit értéket a tulajdonságnak.

Inherit nélkül

Lorem ipsum dolor sit amet

Nam id nisl vitae neque posuere interdum

Inherit

Lorem ipsum dolor sit amet

Nam id nisl vitae neque posuere interdum

```
<!DOCTYPE html>
...5 lines
<html lang="hu">
<head>
  <title>CSS inherit tulajdonság bemutatása</title>
  <meta charset="utf-8">
  <style type="text/css">
    div {border:1px solid blue;padding:5px;}
    p {border:inherit;}
  </style>
</head>
<body>
  <div>
    <p>Lorem ipsum dolor sit amet</p>
    <p>Nam id nisl vitae neque posuere interdum</p>
  </div>
</body>
</html>
```

• Az initial érték használata

- Minden tulajdonságnak van egy kezdeti (initial) értéke, amely egy tulajdonság definíciós táblában van elhelyezve. Ha tehát vissza akarjuk állítani az alapértéket, de nem tudjuk, hogy az pontosan micsoda, használhatjuk az *initial* értéket.

```
div {min-width:initial;}
```

CSS3

• CSS3-as szelektorok

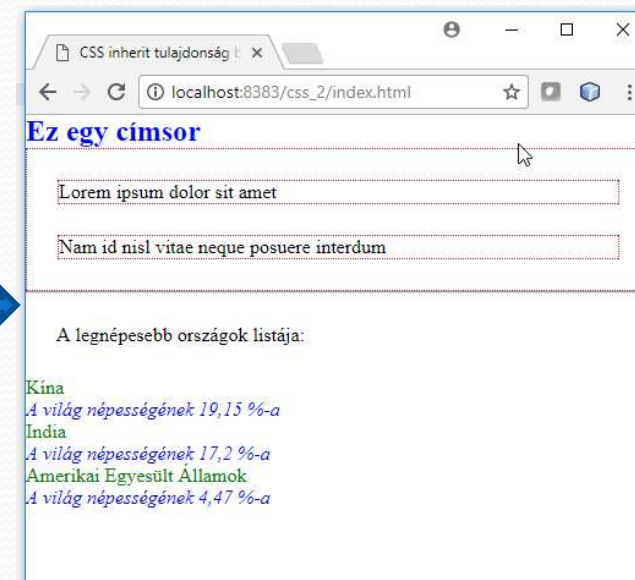
- **Univerzális szelektor (*)** – Az univerzális szelektor segítségével olyan CSS-szabályok írhatók, amelyek minden egyes HTML-elemre vonatkoznak. Ezt a * szelektort nagyon gyakran arra használjuk, hogy minden elem margóját és kitöltését lenullázzuk.
- **Típus szelektor (E)** – A típus szelektorral egy adott elemtípust határozhatunk meg. Például megcímezhetjük az összes bekezdést, címsort, listaelemet stb.
- **Szelektor leszármazott elemre (E F)**
 - Ha egy HTML-elem leszármazottaihoz szeretnénk stílust hozzárendelni, akkor egy speciális szelektorlistát kell alkalmazni, amelyben a szelektorokat szóközők választják el.
 - Ha a listában egy szelektor megelőz egy másikat, akkor az előbb szereplő szelektor a később szereplő szelektor őst jelent.
 - A szelektorlistában egymást követő szelektorok nem feltétlenül egymás közvetlen gyermekei illetve szülei.
 - Csak a szelektorlista utolsó eleme lesz stílussal ellátva.

```
<html lang="hu">
<head ...5 lines />
<body>
  <h2>Ez egy címsor </h2>
  <div>
    <p>Lorem ipsum dolor sit amet</p>
    <p>Nam id nisl vitae neque posuere interdum</p>
  </div>
  <hr>
  <p>A legnépesebb országok listája:</p>
  <ul>
    <li>Kína
      <ul>
        <li>A világ népességének 19,15 %-a</li>
      </ul>
    </li>
    <li>India
      <ul>
        <li>A világ népességének 17,2 %-a</li>
      </ul>
    </li>
    <li>Amerikai Egyesült Államok
      <ul>
        <li>A világ népességének 4,47 %-a</li>
      </ul>
    </li>
  </ul>
</body>
</html>
```

```
@charset "UTF-8";
/*univerzális szelektor*/
*{
  margin:0; /* margó lenullázása */
  padding:0; /* kitöltés lenullázása */
}

/* típus szelektorok*/
div{
  border: brown dotted thin;
}
p {
  margin:25px;
  border: inherit;
}
h2{
  color: blue;
}
li {
  color:green;
}

/*Szelektor leszármaztatott elemre*/
ul li li {
  color:blue;
  font-style:italic;
}
```

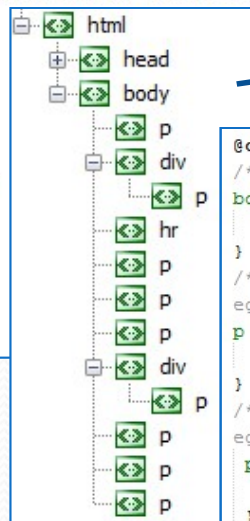


CSS3

• CSS3-as szelektorok

- **Szelektor gyermek elemre (E > F)** – Ha azt szeretnénk kifejezni, hogy az egyik szelektor a másik gyermeke, akkor a szelektorlistában a szelektorokat > jellel választjuk el. Csak a szelektorlista utolsó eleme lesz stílussal ellátva.
- **Szelektor közvetlen testvérre (E+F)** - Szelektorlistával lehetőségünk van közvetlen testvérségi viszony kifejezésére is. Ekkor a listában levő szelektorokat + jellel választjuk el. Csak a szelektorlista utolsó eleme lesz stílussal ellátva.
- **Szelektor általános (azaz tetszőleges) testvérre (E ~ F)** - Az általános testvérségi viszony kifejezésére is van lehetőség szelektorlistával. Ekkor a szelektorokat ~ jellel választjuk el. Csak a szelektorlista utolsó eleme lesz stílussal ellátva.

```
<html lang="hu">
  <head ...6 lines />
  <body>
    <p>Ez a bekezdés gyermeke a body tagnek,
      ezért a háttere világoskék színű.</p>
    <div>
      <p>Ez a bekezdés nem gyermeke a body tagnek.</p>
    </div>
    <hr>
    <p>Első bekezdés.</p>
    <p>Második bekezdés.</p>
    <p>Harmadik bekezdés.</p>
    <div>
      <p>Negyedik bekezdés.</p>
    </div>
    <p>Ötödik bekezdés.</p>
    <p>Hatodik bekezdés.</p>
    <p>Hetedik bekezdés.</p>
  </body>
</html>
```



```
@charset "UTF-8";
/*A gyermek elemek háttere világos kék lesz*/
body > p {
  background-color: lightblue;
}
/*Azoknak a bekezdéseknek, amelyeket megelőz
egy közvetlen bekezdés testvérük, piros lesz a betűszíne*/
p + p {
  color: red;
}
/*Azoknak a bekezdéseknek, amelyeket megelőz
egy bekezdés testvérük, dőlten lesznek szedve.*/
p ~ p {
  font-style: oblique;
}
```

Ez a bekezdés gyermeke a body tagnek, ezért a háttere világoskék színű.

Ez a bekezdés nem gyermeke a body tagnek.

Első bekezdés.

Második bekezdés.

Harmadik bekezdés.

Negyedik bekezdés.

Ötödik bekezdés.

Hatodik bekezdés.

Hetedik bekezdés.

CSS3

• CSS3-as szelektorok

- **Osztályok (class)** – ha nem a struktúra, hanem más szempont szerint szeretnénk az elemeket akkor alkalmazhatjuk az osztályokat.

A HTML-elemeinket a **class** attribútummal kiegészítve és annak értéket adva osztályba sorolhatjuk. Az így definiált osztályokra hivatkozhatunk szelektorral, ezáltal minden osztályhoz egységes stílus rendelhető.

Az osztályszerkezet többféle módon megadható:

- A szelektort az **elem.osztály** alakban, azaz az elemet annak osztályától egy ponttal elválasztva adhatjuk meg. Ekkor a szelektor az összes adott osztályba tartozó elemre illeszkedni fog.
- A szelektorból elhagyható az elem neve. Ekkor a szelektor alakja: **.osztály**, és az összes adott osztályba tartozó elemre illeszkedik, az elemtípustól függetlenül.
- Egy elem egyszerre több osztályba is besorolható, amivel a stílust tovább finomíthatjuk.
Szintaxis: **elem.osztály1.osztály2**

- **Egyedi azonosítók (ID)**

- A HTML-elemeket az id paraméterrel elláthatjuk egyedi azonosítókkal.
- A HTML5 szabvány azt mondja, hogy az azonosítónak legalább 1 karakter hosszúnak kell lennie és nem tartalmazhat szóközt.
- Az így definiált egyedi azonosítókra hivatkozhatunk szelektorokkal, amelyek szintaxisa a következő: **elem#id**
- Mivel minden id egyedi, ezért a szelektor a következő módon rövidíthető: **#id**

Megjegyzés:

- A HTML 4.01-es szabványban az egyedi azonosítókra az volt igaz, hogy betűvel kezdődhetnek, amelyet számok, betűk és a következő jelek követhetnek: - _ : .

CSS3

```
<!DOCTYPE html>
<!--
To change this license header, choose License Headers in Project Properties.
To change this template file, choose Tools | Templates
and open the template in the editor.
-->
<html lang="hu">
<head>
<title>CSS Szelektorok </title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link href="css/base.css" rel="stylesheet" type="text/css">
</head>
<body>
<h1 class="szegely">Címsor 1 (szegely osztály)</h1>
<h1 class="fontos">Címsor 1 (fontos osztály)</h1>
<h1 class="szegely fontos">Címsor 1 (szegely és fontos osztály)</h1>

<hr>



</body>
</html>
```

```
@charset "UTF-8";
h1.szegely {
    border:2px dotted blue;
}
.hfontos {
    color:red;
}
h1.szegely.hfontos {
    border:1px solid red;
    color:black;
}
img#logo {
    border:4px double black;
}
```

Címsor 1 (szegely osztály)

Címsor 1 (fontos osztály)

Címsor 1 (szegely és fontos osztály)

