WEBTERVEZÉS – GYAKORLATI JEGYZET

5. gyakorlat

A HTML és CSS további lehetőségei

Készítették:

Cservenák Bence

Farkas Anikó

Savanya Sándor

FIGYELEM!

A jegyzet folyamatosan készül, így előfordulhatnak benne apróbb hibák, hiányosságok, elírások. Ha valaki esetleg ilyet találna, kérem, írjon a `Cservenak.Bence@stud.u-szeged.hu` címre, hogy mihamarabb javíthassuk.

1. A HTML további használati lehetőségei

A. Multimédia elemek

A HTML5-ös szabvány támogatja multimédia (audió, videó) beágyazását a weboldalunkra.

- `<audio>…</audio>`: hangállomány beszúrása
  - `controls` attribútum: vezérlőgombok megjelenítése
  - `<source/>`: több alternatív fájl megadása (páratlan tag)
    - a böngésző az első általa felismerhetőt fogja beágyazni
    - `src`: a fájl elérési útvonala
    - `type`: a fájl MIME-típusa
- `<video>…</video>`: videó beszúrása
  - `controls` attribútum és `<source/>` tag ugyanúgy, mint hangállomány esetén
  - `autoplay` attribútum: automatikus lejátszás
  - `width`: videó szélessége, `height`: videó magassága

```
<h2>A kedvenc zeném</h2>
<audio controls>
   <source src="powerwolf.mp3" type="audio/mpeg"/>
   <source src="greenday.wav" type="audio/wav"/>
   Ez akkor jelenik meg, ha a böngésző nem támogatja
   hangállomány beágyazását
</audio>
```

A kedvenc zeném


```
<h2>A kedvenc videóm</h2>
<video controls width="360" height="280">
   <source src="allstar.mp4" type="video/mp4"/>
   <source src="rickroll.ogg" type="video/ogg"/>
   Ez akkor jelenik meg, ha a böngésző nem támogatja
   videóállomány beágyazását
</video>
```

A kedvenc videóm

B. Egyéb beágyazások

Az `<object>`, illetve `<embed>` tagekkel

C. A canvas tag

- `<canvas>…</canvas>`: JavaScript alapú rajzolásra használjuk
  - grafika, animációk, böngészős játékok
  - `width`: vászon szélessége
  - `height`: vászon magassága

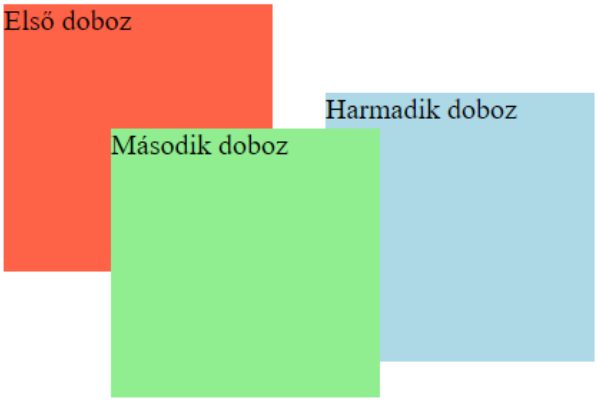
2. A CSS további használati lehetőségei

A. A `z-index` tulajdonság

- Értékéül megadható egy egész szám (akár negatív szám is)
- Az egymást takaró elemek közül az kerül előtérbe, amelyiknek nagyobb a `z-index` értéke

```
<div class="first">Első doboz</div>
<div class="second">Második doboz</div>
<div class="third">Harmadik doboz</div>

<!-- CSS kód a head-ben -->
<style>
   div { width: 150px; height: 150px; }
   .first {
      background-color: tomato;
      position: absolute;
      top: 50px; left: 20px;
      z-index: 1;
   }
   .second {
      background-color: lightgreen;
      position: absolute;
      top: 120px; left: 80px;
      z-index: 200;
   }
   .third {
      background-color: lightblue;
      position: absolute;
      top: 100px; left: 200px;
      z-index: -1;
   }
</style>
```



B. Lekerekített sarkok

- **`border-radius`:** az elem sarkainak sugara
  - paraméterezése: [bal felső sarok] [jobb felső sarok] [jobb alsó sarok] [bal alsó sarok]
  - nem megadott paraméterérték esetén az értéket szemben lévő saroktól örökli

```
<div class="rounded">Ez egy lekerekített sarkú téglalap</div>

<!-- CSS kód a head-ben -->
<style>
  .rounded { border: 2px solid red; border-radius: 0 10px 0 25px; height: 50px;
             width: 300px; padding: 10px; }
</style>
```

Ez egy lekerekített sarkú téglalap

C. Árnyék

- **text-shadow**: szövegárnyék
  - paraméterezés: [vízszintes árnyék] [függőleges árnyék] [mérték] [szín]
    - az utolsó két paraméter elhagyható
- **box-shadow**: dobozszerű elem körüli árnyék
  - paraméterezés: [vízszintes árnyék] [függőleges árnyék] [mérték] [kiterjedés] [szín]
    - az utolsó három paraméter elhagyható

```
<p>Ez egy árnyékkal rendelkező szöveg</p>
<div>Ez egy árnyékkal rendelkező doboz</div>

<!-- CSS kód a head-ben -->
<style>
  p { text-shadow: 2px 0 lightgray; }
  div { border: 1px solid black; box-shadow: 0 3px lightblue, 0 2px 25px yellow;  }
</style>
```

Ez egy árnyékkal rendelkező szöveg

Ez egy árnyékkal rendelkező doboz

D. Pseudo-elemek

A CSS pszeudo-elemeket HTML objektumok specifikus részeinek formázására használjuk.

- `::before`: generált tartalom közvetlenül az elem előtt
- `::after`: generált tartalom közvetlenül az elem után
- `::selection`: a kijelölt szövegrész
- `::first-letter`: a szöveg első karaktere
- `::first-line`: a szöveg első sora

```
<p>Első bekezdés</p>
<p>Második bekezdés</p>
<p>Harmadik bekezdés</p>

<!-- CSS kód a head-ben -->
<style>
  p::selection { background-color: lightgreen; }
  p::first-letter { font-size: 120%; color: red; }
  p::after { content: " -- bekezdés vége"; }
</style>
```

Első bekezdés -- bekezdés vége

Második bekezdés -- bekezdés vége

Harmadik bekezdés -- bekezdés vége

Példa: A kijelölés, első betű és az elem utáni tartalom megadása pseudo-elemek segítségével

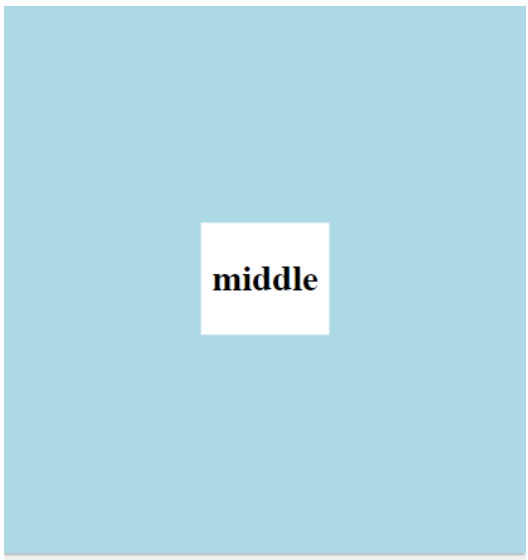
E. Transzformálás

- **transform**: HTML elem transzformálása
  - A transzformálás régebbi böngészőkben nem érhető el
  - `translate(x, y)`: eltolja az elemet a jelenlegi helyéhez képest (`x`: jobbra, `y`: le)
  - `rotate(xdeg)`: elforgatja az elemet `x` fokkal (`x` negatív is lehet)
  - `scale(w, h)`: nyújtás (`w`: szélesség, `h`: magasság)
    - `scaleX(n)`: eredeti szélesség megnyújtása `n`-szeresére
    - `scaleY(n)`: eredeti magasság megnyújtása `n`-szeresére
  - `skew(xdeg, ydeg)`: elferdítés (`x`: vízszintes, `y`: függőleges)
    - `skewX(ndeg)`: vízszintes elferdítés `n` fokkal
    - `skewY(ndeg)`: függőleges elferdítés `n` fokkal
  - `matrix(scaleX(), skewY(), skewX(), scaleY(), translateX(), translateY())`: a fentiek összevonása
  - `rotateX(ndeg)`: elforgatja az elemet az x-tengely körül `n` fokkal
  - `rotateY(ndeg)`: elforgatja az elemet az y-tengely körül `n` fokkal
  - `rotateZ(ndeg)`: elforgatja az elemet a z-tengely körül `n` fokkal

```
<div><h1>middle</h1></div>

<!-- CSS kód a head-ben -->
<style>
  body { background-color: lightblue; }

  div {
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
    margin: 0 auto;
    text-align: center;
    background-color: white;
    padding: 10px;
  }
</style>
```



Példa: HTML objektum középre igazítása, vízszintesen és függőlegesen

```
<div>ez eldőlt…</div>

<!-- CSS kód a head-ben -->
<style>
  div {
    transform: translate(50px, 20px)rotate(50deg);
    width: 100px;
    height: 100px;
    background-color: lightgreen;
  }
</style>
```

Példa: HTML objektum eltolása, és 50 fokkal való elforgatása

F. A `transition` tulajdonság (áttűnések)

- Időben dinamikusan változtathatjuk meg elemeink stílusát azok állapotváltásai között
- Az áttűnések a régebbi böngészőkben nem érhetők el
- Megadjuk a CSS tulajdonságot, amit az áttűnés megváltoztat, valamint az áttűnéshez szükséges időt (másodpercben)
- `transition-timing-function`: az átmenet lefolyásának típusa
  - `linear`: egyenletes állapotváltás
  - `ease`: fokozatosan lassuló állapotváltás
  - `ease-in`: gyorsuló állapotváltás
  - `ease-out`: lassuló állapotváltás
  - `ease-in-out`: gyorsuló, majd lassuló állapotváltás
- Vesszővel elválasztva több átmenet is megadható

```
<div></div>

<!-- CSS kód a head-ben -->
<style>
  div {
    width: 100px;
    height: 100px;
    background: tomato;
    transition: width 2s, height 2s;
    transition-timing-function: ease-in;
  }

  div:hover { width: 300px; height: 300px; }
</style>
```

Példa: Egy téglalap alakú doboz méreteinek megnövelése áttűnésekkel

G. Animációk

- A CSS lehetőséget biztosít animációk készítésére JavaScript vagy Flash használata nélkül
- Az animációk régebbi böngészőkben nem érhetők el
- `@keyframes animacio_neve { … }`: az animáció kódja
  - `from { … }`: az animáció kezdetén fennálló stílus
  - `to { … }`: az animáció befejezése után fennálló stílus
  - megadhatók százalékértékek is (az animáció hány százalékban hajtódott végre)
- Az animálandó elem esetén megadható CSS tulajdonságok:
  - `animation-name`: az animáció neve (ami a `@keyframes` után áll)
  - `animation-duration`: az animáció hossza
  - `animation-delay`: animáció kezdete előtti késleltetés
  - `animation-iteration-count`: hányszor játszódjon le az animáció
    - megadható szám vagy `infinite` (végtelen)
  - `animation-timing-function`: az animáció lefolyásának típusa
    - `linear`, `ease`, `ease-in`, `ease-out`, `ease-in-out` - mint a `transition`-nél
  - `animation-direction`: az animáció iránya
    - `normal`: az animáció előrefelé lesz lejátszva (alap)
    - `reverse`: az animáció visszafelé lesz lejátszva
    - `alternate`: az animáció először előrefelé, majd visszafelé lesz lejátszva
  - ezek összevonva is megadhatók: `animation: name duration timing-function delay iteration-count direction;`

```
@keyframes else {
   0% { background-color: yellow; }
   25% { background-color: orange; }
   50% { background-color: tomato; }
   75% { background-color: lightgray; }
   100% { background-color: lightgreen; }
}

body {
   animation-name: elso;
   animation-duration: 10s;
   animation-iteration-count: infinite;
   animation-direction: alternate;
}
```

Példa: Váltakozó háttérszín animáció

```
<div class="float-in">CSS</div>

<!-- CSS kód a head-ben -->
<style>
   @keyframes masodik {
      from { left: -50%; }
      to { left: 50%; }
   }
   .float-in {
      width: 100px; height: 100px;
      background-color: lightgreen;
      position: relative;
      text-align: center;
      animation: masodik 3s forwards;
   }
</style>
```

Példa: Beúszás effektus animáció

H. Media query-k, nyomtatási stíluslap

- Lényegük, hogy a megadott CSS formázást csak a meghatározott médiatípusok és feltételek esetén végzi el a böngésző → pl. különböző eszközökre és felbontásokra való optimalizálás
- Szintaxis: `@media [not|only] mediatype and (mediafeature) { CSS formázás }`
  - `not` (opcionális): az egész media query jelentését negálja
  - `only` (opcionális): régi, CSS3 media query-ket nem ismerő böngészőknél nem formáz
  - `mediatype` lehetséges értékei:
    - `screen`: számítógépek, táblagépek, okostelefonok, stb. képernyője
    - `print`: nyomtató (**nyomtatási stíluslap**)
      - `@page` direktívával megadhatjuk a nyomtatási margókat is
    - `speech`: felolvasóprogram
    - `all`: minden médiatípus eszköz (alapértelmezett)
  - `mediafeautre` helyén megadjuk a feltételt, ami mellett elvégezzük a formázást

```
<img src="giraffe.jpg" alt="Zsiráf"/>
<p>A zsiráf (Giraffa camelopardalis) Afrikában
élő párosujjú patás emlős…</p>

<!-- CSS kód a head-ben -->
<style>
   img {
      height: 140px;
      float: left;
      margin-right: 10px;
   }

   @media only screen and (max-width: 768px) {
      img {
         float: none;
         display: block;
         margin: auto;
      }
   }
</style>
```

Példa: Egy media query, ami a 768px-nél nem szélesebb képernyők esetén megszűnteti a szöveg kép körüli körbe-futtatását, és középre igazítja a képet

```
@media print {
   p {
      font-size: 12pt;
   }

   img {
      display: none;
   }

   h1, h2, h3, h4, h5, h6 {
      page-break-after: avoid;
   }
}

@page :left {
   margin: 0.5cm;
}
```

Példa: Nyomtatási stíluslap készítése

I. Egyéb hasznos CSS ismeretek

- Google Web Fonts betűtípusok használata
- Lenyíló menü készítése
- Hasábolás
- Elemek számozása