Web-technológia

HTML, I. rész

Dr. Hatwagner F. Miklós

Széchenyi István Egyetem, Győr

https://github.com/wajzy/GKxB_INTM049.git

2020. szeptember 21.

SZÉCHENYI EGYETEM
UNIVERSITY OF GYŐR

INFORMATIKA TANSZÉK
DEPARTMENT OF COMPUTER SCIENCE

- Cél: a nyers szöveg egyes részeit strukturálni, jelentésbeli többletet hozzáadni (pl. fejezetcím, bekezdés)
- Történeti előzmény: nyomdai előkészítés, kéziratok szerkesztése, gépi szedőrendszerek
- Példák jelölőnyelvekre: roff, LaTeX, SGML

RUNOFF

nyers szövegből és parancsokból (.XX) álló fájlok → tördelt megjelenítés buta terminálokon (OS: Compatible Time Sharing System, CTSS, 1963)

runoff

a RUNOFF bővített képességű portja *IBM Selectric* terminálokhoz (OS: Multiplexed Information and Computing Service, multics, ≈'60-as évek vége)

roff

a runoff továbbfejlesztése a Bell Telephone Labs-nál (1973) a PDP-11 géphez kapcsolt *Graphic Systems CAT* (grafikus szedőegység) miatt. A roff család:

troff

typesetter roff a CAT-hez

nroff

terminálokhoz és nyomtatókhoz

roff

korlátozott képességű runoff utód, nem fejlesztették tovább

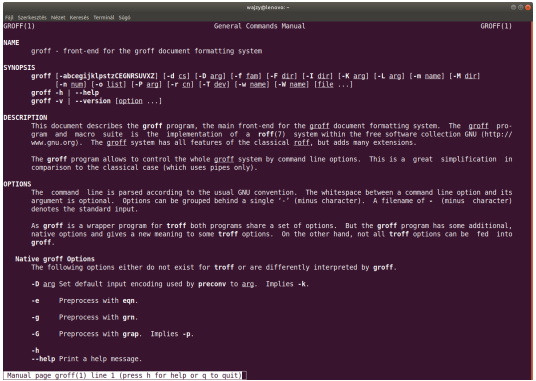
groff

GNU implementáció, máig fejlesztik → man oldalak

/usr/share/man/man1/groff.1.gz/groff.1

```
.TH GROFF 1 "10 February 2018" "Groff Version 1.22.3"
.SH NAME
groff \- front-end for the groff document formatting system


.\" ----------------------------------
.SH SYNOPSIS
.\" ----------------------------------
.
.SY groff
.OP \-abcegijklpstzCEGNRSUVXZ
.OP \-d cs
.OP \-D arg
.OP \-f fam
.OP \-F dir
.OP \-I dir
```



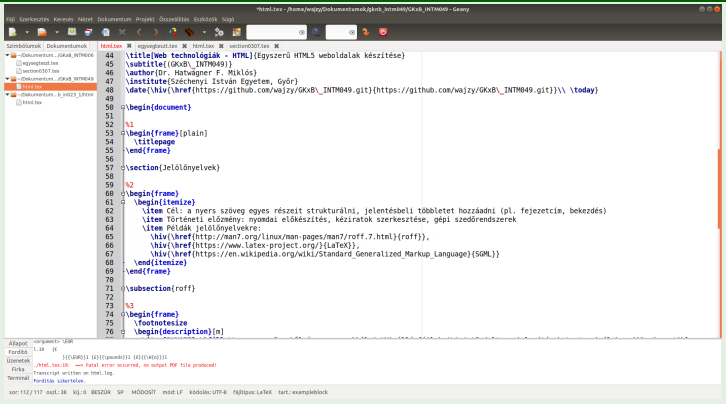
TEX

Betűszedő rendszer, fejlesztője Donald E. Knuth, 1978 (Elégedetlenség könyvének szedésével.)

LATEX

TEX-en alapuló szövegformázó rendszer, Leslie Lamport, 1983

html1.tex



- SGML (Standard Generalized Markup Language), ISO 8879:1986
- Szabványos jelölőnyelv dokumentumok szerkezetének leírására, beleértve a címkék definiálását is
- Gépfüggetlen metanyelv
- Előzménye: GML (1969)
  - C. Goldfarb (IBM), E. Mosher, R. Lorie
  - dokumentumtípusonként egyedi kódolási séma definiálható
  - előre definiált elemek egymásba ágyazhatóak
  - először az IBM nyomdarendszere használta
- Tulajdonságai
  - Deklaratív: struktúrát és attribútumokat rögzít, nem a feldolgozás módját ($\rightarrow$ időtállóság)
  - Gépi feldolgozás lehetősége

- Legfontosabb építőelemek
  - Elemek ([element] nyitó és záró cimkék [tag] által határolva)
  - A nyitó tagben attribútumok (kulcs-érték párok) adhatók meg
  - Elemek egymásba ágyazhatóak
  - Elemek, attribútumok alkalmazási szabályai → Document Type Definition (DTD)
- Néhány korai, jelentős alkalmazás
  - Electronic Manuscript Project of the Association of American Publishers (AAP, tudományos dokumentumok)
  - Computer-aided Acquisition and Logistic Support (CALS, katonai dokumentumok kezelése)
  - LinuxDoc (Linux csomagok)

SGML példa

```
<!DOCTYPE PEOPLE SYSTEM
  "people.dtd">
<PEOPLE DATE="15 6 2000">
 <NAME TITLE="Mr">
  <FIRST>Wally</FIRST>
  <LAST>Wallpaper</LAST>
 </NAME>
 <NAME>
  <LAST>Jackson</LAST>
 </NAME>
 <NAME TITLE="Dr">
  <FIRST>Susan</FIRST>
  <MIDDLE>Ramsay</MIDDLE>
  <LAST>Sukie</LAST>
 </NAME>
</PEOPLE>
```

people.dtd

```
<!ELEMENT people - - (name+)>
<!ATTLIST people date NUMBERS #REQUIRED>

<!ELEMENT name - - (first?, middle?, last)>
<!ATTLIST name title CDATA #IMPLIED>

<!ELEMENT first - - (#PCDATA)>
<!ELEMENT middle - - (#PCDATA)>
<!ELEMENT last - - (#PCDATA)>
```

Forrás: OmniMark dokumentáció

- ENQUIRE: a CERN dokumentumtároló, -megosztó szoftvere. (Tim Berners-Lee, 1980)
- HTML első említése: T.B.L., 1991 (18 elem, melyek a CERN SGMLguid-on, a kutatóintézet SGML alkalmazásán alapultak)
- A HTML egy SGML alkalmazás: 1993-2014
- HTML 4.01: Strict/Transitional/Frameset DTD, 1999
- Aktuális változat: HTML5
- Néhány újdonság: videó- és hanglejátszás, vektorgrafika, többszálúsítás, helyi adattárolás, bittérképes grafika, stb.
- „Élő szabvány", meghatározó szervezetek: W3C (ajánlások), WHATWG (innovatív technológiák)
- 2019-től a WHATWG tartja karban a HTML szabványát.
- XHTML: XML előírásoknak megfelelő HTML; a HTML5 „feleslegessé" tette

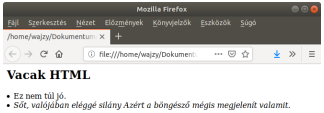
Általános tulajdonságok

- Egyszerű szövegfájl (jellemzően UTF-8 kódolással)
- Dokumentum *strukturájának* jelölésére, pl.
  - fejlécek
  - listák
  - bekezdések
  - hiperhivatkozások
- Megjelenítést befolyásolja
  - böngésző alapértelmezése
  - felhasználó globális beállításai a böngészőben
  - stíluslapok (CSS)
- Megjelenítés leválasztása
  - helyes megjelenítés többféle böngészőben
  - könnyebben karbantartható oldalak
  - nem vizuális böngészők támogatása

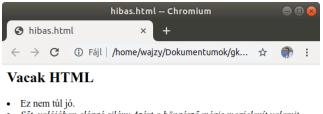
- Struktúra kialakítása az SGML-hez hasonlóan: egymásba ágyazható elemek, címkék, attribútumok
- Beágyazási szabályok, használható attribútumok → „szabvány” (ajánlás)
- Helytelenül formázott dokumentumok
  - Nincsenek hibaüzenetek
  - A böngésző a tőle telhető legjobb eredményt nyújtja
  - Kompatibilitási okokból az elavult megoldásokat is kénytelen támogatni
  - Ellenőrzés különböző böngészőkben vs. szintaxis validálás

hibas.html

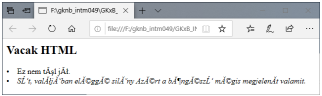
```
<h2>Vacak HTML</h1>
<li><a>Ez nem túl</b> jó.
<li><i>Sőt, valójában eléggé <g>silány</g></em>
</ul>
Azért a böngésző mégis megjelenít <a hiv=jajj>valamit.
```



Mozilla Firefox 69.0



Chromium 76.0.3809.100



Microsoft Edge 44.17763.1.0

Válasszunk egy szövegszerkesztőt (pl. Geany, VS Code, NotePad++, ...), gépeljük be és mentsük ki az alábbi fájlt `elso.html` néven, UTF-8 kódolással!

elso.html

```
<!DOCTYPE html>
<html lang="hu-HU">
  <head>
    <!-- Ez itt egy megjegyzés -->
    <title>Első weboldalam!</title>
    <meta charset="utf-8" />
  </head>
  <body>
    <h1>Minden kezdet nehéz</h1>
    <p>Még ezt sem volt egyszerű összehozni, pedig olyan rövidke...</p>
  </body>
</html>
```

Dokumentum típusának meghatározása

HTML5 Nincs DTD!

```
<!DOCTYPE html>
```

4.01, Szigorú

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
```

4.01, Átmeneti

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

4.01, Keretek

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
```

Elemek (element)

- Általában nyitó és záró cimkék (tag) között, pl. `<body>...</body>`, `<p>...</p>`
- Néha a böngésző kitalálja, hol kellene lennie az elem (pl. `<p>`, `<li>`) záró címkéjének, így az elhagyható, de nem javasolt (XML elemző számára szabálytalanná teszi a fájlt)
- Léteznek üres elemek is; itt nincs mit közbezárni címkékkel, pl. `<meta />`, vízszintes vonal `<hr />` vagy `<hr>`
- Rögzített szabályok szerint egymásba ágyazhatók
- Kis- és nagybetűkre érzéketlen, de ajánlott a kisbetűs írásmód
- A szöveg tördelése független a forrásszöveg tördelésétől (pl. az egymás mellé gépelt fehér karaktereket egynek tekinti)
- Címkék mindig < és > jelek között
- Jelentéssel bíró karakterek bevitele entitásokkal (pl. < → < vagy > → >)

Elemek

`<html>` gyökérelem, pontosan egynek kell lennie (dokumentum nyelve attribútummal, ld.)

`<head>` metaadatok

- `<title>` dokumentum címe (böngészőablak vagy -fül felirata)
- `<meta>` általános metaadat

`<body>` megjelenítendő tartalom

- `<h1>` „Címsor1”
- `<p>` Bekezdés (paragraph)

Megjegyzések

`<!--` és `-->` között

Attribútumok

- Mindig a nyitó címkében (pl. `<html lang="hu-HU">`, RFC1766 szerint)
- Kulcs-érték párok, = jellel elválasztva
- Ajánlott a kulcsot kisbetűvel írni
- Az értéket ajánlott idézni, lehetőleg "-vel (de az ' is megfelel; szóközt tartalmazó értéknél pedig kötelező)
- Egy címkében lehet több attribútum is
- Vagy egy sem (minimalizált szintaxis); itt az attribútum léte hordoz információt (pl. `<p hidden>`). XML feldolgozók megkövetelik az értéket, pl. `<p hidden="hidden">`

Globális attribútumok: minden HTML elemben használhatóak. Például:

`lang`
: Az elem nyelve, pl. `lang="en"` vagy `lang="en-US"`

`dir`
: Írásirány, balról jobbra (`dir="ltr"`) vagy jobbról balra (`dir="rtl"`).

`title`
: Extra információ az elemről, jellemzően felbukkanó „buborékban” (tooltip).

`id`
: Az elem egyedi azonosítója. Főleg oldalon belüli hivatkozásoknál (`<a> elem`), formázásoknál (→ CSS) és JavaScript programok készítésekor használják. Értéke nem kezdődhet számjegy karakterrel. Pl. `id="elsoBekezdes"`.

`class`
: Egy elem valamely csoporthoz tartozását jelöli. Hasonlít a szövegszerkesztők stílusaihoz, mert minden, azonos attribútum értékű elem azonos módon formázható CSS-sel. pl. `class="btn-primary"`.

`style`
: Egy elem egyedi formázását teszi lehetővé CSS-sel. Használata általában kerülendő, de JavaScript programokban, webalkalmazásoknál szokás használni. Pl. `style="color: red;"` pirosra állítja az írásszínt.

`hidden`
: Elrejti az elemet (`hidden="hidden"`).

accesskey

Gyorsbillentyűt definiál egy elem aktiválásához/beviteli fókuszba kerüléséhez. Jellemzően hivatkozásoknál (<a>) és űrlapok vezérlőinél (<input>) használják, pl. `accesskey="b"`. Valamilyen váltó billentyű(k) együttes megnyomását igényli, pl. `Alt` vagy `Alt+Shift`.

tabindex

Főleg beviteli mezők vagy hivatkozások `Tab` billentyűvel történő bejárási sorrendjének magadásához/módosításához használják, az 1 érték jelöli az első vezérlőt. Pl. `tabindex="3"`.

data-*

Alkalmazásspecifikus adatokat rögzít az elemhez, pl. `data-toggle="modal"`. Gyakran JavaScript programok számára így ad meg paramétereket a Backend.

`contenteditable`
: Egy elem szerkeszthetőségét állítja. Értékét alapvetően a szülőtől örökli. Pl. `<p contenteditable="true">Szerkeszthető bekezdés!</p>` esetén egy felhasználó által szerkeszthető bekezdést kapunk.

`spellcheck`
: Helyesírás-ellenőrzés bekapcsolása űrlapok szövegbeviteli mezőin és szerkeszthető elemek tartalmára. Pl. `<p contenteditable="true" spellcheck="true">Ez egy szerkeszthető, helyesírás-ellenőrzött bekezdés!</p>`

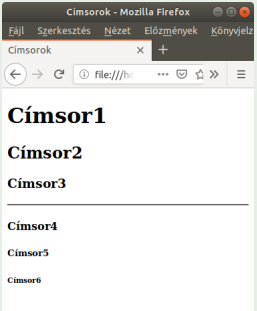
`draggable, dropzone`
: Húzd és ejtsd támogatás megvalósításához, de JavaScript kód is szükséges a megvalósításhoz.

`translate`
: Szövegek lefordítását lehetne vele tiltani (`translate="no"`), de a böngészők gyengén támogatják.

- `<h1>`, `<h2>`, ..., `<h6>`: legmagasabbtól legalacsonyabb szintig
- Például: `<h1>Első fejezet, amelyben bemutatnak bennünket Micimackónak és a méheknek, mellékesen a könyv is elkezdődik</h1>`
- Általában nagyobb betűméretek és a címsor elé és/vagy mögé tett térközök jellemzik
- Keresőmotorok is használhatják a dokumentum struktúrájának feltérképezésére
- Tematikus részek elválasztására gyakran elválasztó vonalat (`<hr />`) használnak

Készítsen weboldalt, ami egymás alá írja a *Címsor1*, *Címsor2*, . . . , *Címsor6* szövegeket, a nekik megfelelő HTML elemekkel, a 3. és a 4. szövegsor között pedig húz egy vízszintes vonalat!

cimsorok.html



- Bekezdések `<p>` elemmel jelölhetők
- Például: `<p title="Bek. 1">Első bekezdés.</p><p>Második bekezdés.</p>`
- Alapértelmezetten térközt hagy a böngésző a bekezdések között
- Sortörés új bekezdés (és térköz) nélkül: `<br />`
- Előformázott szöveg (pl. karakteres módú program kimenete), egyenszélességű (monospace) karakterekkel, forrás fehér karaktereinek megőrzésével: `<pre>`

„Tradicionális” tördelési módok

Blokkszintű (block)
: Új sorban kezdődik, és a teljes elérhető szélességet elhasználja, pl. `<p>`, `<pre>`, `<h1>`-`<h6>`, `<hr />`

Soron belüli (inline)
: Aktuális sorban kezdődik, csak olyan széles helyet foglal, ahol éppen elfér, pl. `<br />`

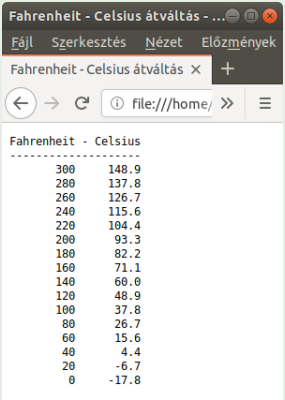
Formázza meg a verset a mellékelt ábra szerint, azaz

- a költő neve legyen első szintű címsor,
- a mű címe második szintű,
- a szakaszok (arab számokkal jelölve) harmadik szintűek.
- A bekezdéseket és a bekezdésen belüli sortöreseket állítsa be!
- A bekezdések fölé mozgatva az egeret lássuk a bekezdés sorszámát, pl. 1/1, 1/2, . . . , 2/3

attila.html

Formázza meg a klasszikus Fahrenheit-Celsius átváltó program kimenetét a mellékelt ábra szerint, azaz őrizze meg a program karakteres kimenetének formázását a <pre> elemmel!
A <pre> elem megőrzi a HTML forrásban lévő szóközöket, sortöréseket, és monospace betűtípust használ.

fahrcels.html



A soron belüli formázó elemek közül előnyben részesítjük azok használatát, melyek *szemantikai többletet* adnak a szöveg tartalmának, a pusztán *formázási célú* elemekkel szemben → erre ott a CSS!

Ettől függetlenül, néhány, csak formázási célú elem használata továbbra is szabványos a HTML5-ben.

`<small>`
: Kisbetűs szöveg. (A `<big>` elem a HTML5-ben már nem támogatott.)

`<i>` (italics)
: Döntötten szedett szöveg, jelentéstöbblet nélkül.

`<em>` (emphasized)
: Hangsúlyos, fontos szövegrész, melyet a böngésző alapértelmezetten általában dőlt betűkkel jelenít meg.

`<b>` (bold)
: Félkövéren szedett szöveg, jelentéstöbblet nélkül.

`<strong>` (strong importance)
: Kiemelten hangsúlyos, fontos szövegrész, melyet a böngésző alapértelmezetten általában félkövér betűkkel jelenít meg.

`<sup>` (superscript)
: Felső index.

`<sub>` (subscript)
: Alsó index.

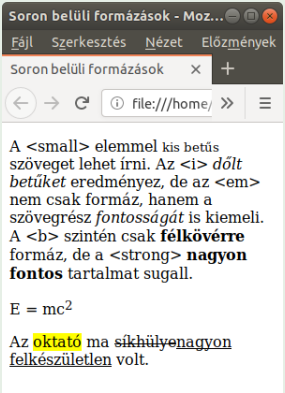
`<ins>` (inserted)
: Utólag beszúrt szöveg (ált. aláhúzással jelölve).

`<del>` (deleted)
: Kitörölt szöveg (ált. áthúzással jelölve).

`<mark>`
: Kijelölt szöveg (ált. sárga háttérrel kiemelve).

Próbálja meg előállítani azt a HTML fájlt, ami a jobb oldalon látható módon jelenik meg a böngészőben! Kiinduláshoz felhasználhatja a dokumentum nyers szövegét. Ne feledje, hogy az olyan, önmagukban is jelentéssel bíró karakterek megjelenítése, mint pl. a < karakter, HTML entitásokkal lehetséges.

soronbelul.html



Szövegrészek idézése. Forrás jelölése: `cite` attribútummal.

`<q>` (quote)

Rövid szövegrészlet idézése, ált. automatikusan körbeveszi a böngésző idézőjelekkel. Soron belüli elem.

`<blockquote>`

Hosszú szövegrészek, bekezdések idézése. Jellemzően behúzással formázva. Blokkszintű elem.

Rövidítések. Kifejtés megadható: `title` globális attribútummal.

`<abbr>`

Rövidítések és betűszavak jelöléséhez. Soron belüli elem.

`<acronym>`

Betűszavak jelöléséhez használták, ELAVULT.

Szöveg írásirányának jelölése: `dir` globális attribútummal, alapértelmezést a nyelvvel együtt a `html` elemben kell állítani. Értékek:

`ltr` (left to right)
: balról jobbra

`rtl` (right to left)
: jobbról balra

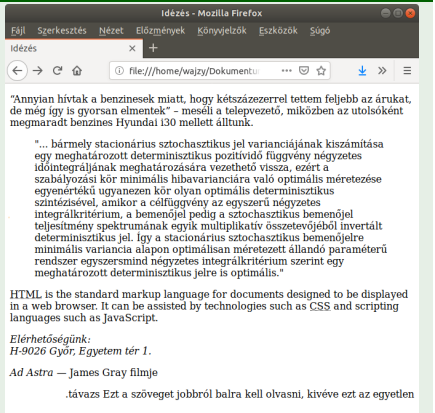
Írásirány helyi módosítása: `<bdo>` (Bi-Directional Override) elemmel, `dir` attribútummal

Mű címének jelölése: `<cite>` soron belüli elemmel.

Postacím megadása: `<address>` blokkszintű elemmel.

Próbálja meg előállítani azt a HTML fájlt, ami a jobb oldalon látható módon jelenik meg a böngészőben! Kiinduláshoz felhasználhatja a dokumentum nyers szövegét. A gondolatjelek többféle szélességben elérhetők: `–`, `—`.

idezes.html



Képeket az `<img>` üres, soron belüli elemmel adunk meg. Attribútumok:

`alt` (alternate text)
: Kép leírása (gyengénlátók, szöveges böngészők, stb. számára), kötelező.

`src` (source)
: Teljes/relatív elérési útvonal, URL

`width`
: Szélesség képpontokban

`height`
: Magasság képpontokban

Bár a méretet megadó attribútumok nem elavultak, *ajánlott* a méretet CSS-sel megadni (CSS szabályok felülbírálják az attribútumok tartalmát). Az oldalak felesleges újratördelése elkerülhető a méretek megadásával.

Széles körben támogatott formátumok: jpeg, gif, png.

Képek (<img> elemek) beágyazhatók a <figure> blokk szintű elembe.
Célszerű beágyazni a kép feliratát is <figcaption> elemben.

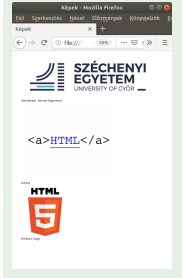
Például:

```
<figure>
  <img alt="kep" src="kep.jpeg" width="320" height="240" />
  <figcaption>Az egyetem logoja</figcaption>
</figure>
```

Készítse el az alábbi oldalt! Képek adatai:

1. Forrás: SZElogo.png, méret: 7088x2363 (kicsinyítse 10%-ra!)
2. Forrás: html.gif, méret: 500x400
3. Forrás: https://www.w3.org/html/logo/downloads/HTML5_Logo_256.png, méret: 256x256

kep.html



Néha a képfájlokat nem tudjuk/akarjuk feltölteni egy szerverre → beágyazzuk őket a weboldalba Data URL-ek segítségével:

Általános felépítés

```
data:[<mediatype>][;base64],<data>
```

dataurl.html

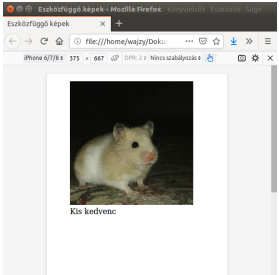
```
<img src="...ZnPg0K" alt="IT logo" width="400" />
```

- Nem csak képek forrása adható meg ilyen módon, de ez a leginkább jellemző.
- Sok böngészőnél az URL hossza nem haladhatja meg a 64kB-ot.
- Base64 kódolást lehet végezni Linuxon/Max OS-en a base64 programmal, de vannak online eszközök is.

Kis kijelzőre nincs értelme túl nagy képet letölteni, majd összezsugorítani → eszközfüggő képek letöltése *CSS3 media query* segítségével

kepmeret.html, képfájlok

```
8     <figure>
9       <picture>
10        <source media="(min-width: 1800px)" srcset="dog.jpeg" />
11        <source media="(min-width: 512px)" srcset="cat.jpeg" />
12        <img alt="Egér" src="mouse.jpeg" />
13      </picture>
14      <figcaption>Kis kedvenc</figcaption>
15    </figure>
```



1. A böngésző az első, feltételeknek megfelelő képet fogja használni.
2. Az utolsó beágyazott elem egy <img> legyen! (Kompatibilitás miatt, és alapértelmezett képet is definiál.)

Hiperhivatkozások készíthetők <a> (anchor) soron belüli elemmel. A nyitó és záró címkék közötti szöveg/kép az érzékeny terület. Attribútumok:

`href` (hypertext reference)
: Abszolút/relatív útvonal/URL, az ugrás célja. Könyvtár megadás esetén célszerű / jellel zárni.

`target`
: Hol nyíljon meg a betöltött tartalom? Értéke lehet:

- `_blank` Új ablakban/fülön
- `_self` Ugyanott, ahol a link is található (alapértelmezés)
- `_parent` Szülő keretben. (A keretek ELAVULTAK.)
- `_top` A teljes ablakban, a keretből „kitörve”. (A keretek ELAVULTAK.)
- *keretnév* Adott nevű keretben. (A keretek ELAVULTAK.)

`title`
: Tooltip szöveg

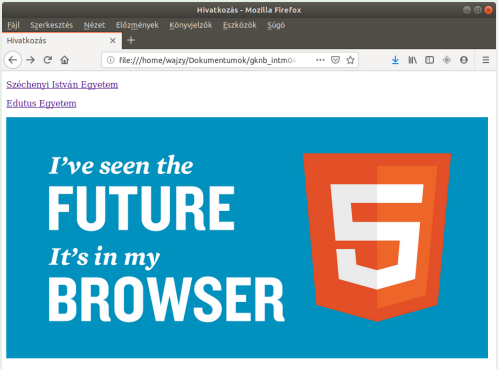
Például: `<a href="https://www.google.com/">Ugrás a kereső oldalára</a>`

Az ábrának megfelelően készítsen három hivatkozást egymás alatti bekezdésekben!

- A Széchenyi Egyetem oldalát az aktuális oldal helyére kell betölteni,
- az Edutusét új ablakba/fülre!
- Végül a HTML5sticker.png képet használva lehessen eljutni a W3C oldalára!

Készítsen feliratokat (SZE, Edutus, HTML5), melyek megjelennek az egeret a link fölé mozgatva!

hivatkozas.html

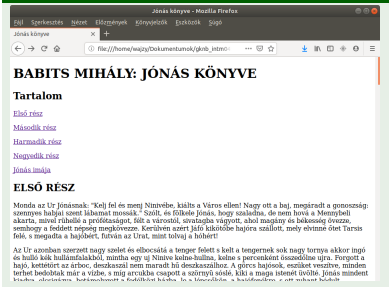


Oldalon belülre mutató hivatkozások is készíthetők, főleg hosszú dokumentumokhoz.

1. Egyedi azonosító készítése az ugrás céljához `id` (globális) attribútummal, pl.: `<h1 id="egy">Első fejezet</h1>`
2. Hivatkozás (`<a>` elem) készítése, a `href` attribútum értékét #-tel kell kezdeni, majd az azonosítóval folytatni, pl. `<a href="#egy">Ugrás az 1. fejezetre</a>`

A jonas.txt fájlból kiindulva hozza létre az ábrán látható oldalt! Az első sor első szintű címsor, a tartalomjegyzék felirat és a vers egyes részei második szintűek. A vers bekezdései bekezdésként jelöltek. A tartalomjegyzék soraira kattintva lehessen elugrani a megfelelő versszakig!

jonas.html



Kép egyes részei kijelölhetők, és más-más oldalakra hivatkozhatnak.

1. Megjeleníteni a képet `<img>` elemmel, elhelyezni benne a `usemap` attribútumot, melynek értéke #-tel kezdődik, és a *térkép* nevével folytatódik.
2. Létrehozni valahol egy `<map>` elemet, melynek `name` attribútuma tartalmazza a térkép nevét.
3. Ebbe beágyazni `<area>` üres elemeket, melynek `shape` attribútuma egy alakzatot, `coords` attribútuma pedig ennek koordinátáit definiálja:
   - `rect` téglalap, két átellenes sarok X, Y koordinátájával
   - `circle` kör, középpont X, Y koordinátája, sugár
   - `poly` poligon, egymást követő pontok x, Y koordinátái; az utolsót összeköti az elsővel
   - `default` a teljes kép külön nem jelölt része
4. Az `<area>` `href` attribútuma definiálja a célt.

A cyclist.jpg felhasználásával hozza létre a terkepek.html weboldalt! A képen kijelölendő területek alakja, koordinátái és célja az alábbi:

Mit jelöl ki?	Alakzat	Koordináták	Cél
Első kereket	Kör	217, 813, 145	https://ebike.hu/termekek/kerek/felni/
Kosarat	Téglalap	102, 420, 256, 643	https://ebike.hu/termekek/kiegeszitok/csomagtarto/elore/
Gyerekülést	Poligon	895, 346, 859, 409, 841, 480, 774, 507, 744, 579, 771, 690, 742, 732, 832, 724, 813, 606, 873, 589, 873, 466, 915, 358	https://ebike.hu/termekek/kiegeszitok/gyermekules/hatra-vazra/
Maradék részt			https://ebike.hu/

Szerver oldali térképek

A kattintás helyét küldi a szervernek URL-be kódolva.

1. Hozzunk létre hivatkozást a webhelyre! (<a> elem)
2. Ágyazzuk bele a képet (<img>) az ismap attribútummal!

Például: `<a href="weblap.html"><img alt="Kép" src="kep.jpeg" ismap="ismap"></a>`

Feladat: alakítsa át az előző feladatot szerver oldali térképet használóra! (Megoldás)

Táblázat célja: adatok táblázatos formában történő megjelenítése.

Ne használjuk az oldal elrendezésének megadására (→ CSS)!

Kötelező elemek:

`<table>`
: Maga a táblázat.

`<tr>` (table row)
: A táblázat egy sora, a `<table>` elembe kell beágyazni.

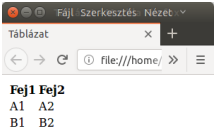
`<td>` (table data)
: A sor egy cellája, a `<tr>` elembe kell beágyazni, vagy helyette használható a

`<th>` (table header)
: fejléc cella. (Félkövér, középre zárt.)

A táblázatnak és a celláknak alapértelmezés szerint nem látszanak a szegélyei. A cellák szélessége a tartalmuktól függ.

tabla1.html

```
8   <table>
9     <tr><th>Fej1</th><th>Fej2</th></tr>
10    <tr><td>A1</td><td>A2</td></tr>
11    <tr><td>B1</td><td>B2</td></tr>
12  </table>
```



Szomszédos cellák (`<td>`, `<th>`) összevonásakor az összevont cellák száma

vízszintes összevonásnál
: a bal szélső cella `colspan` attribútumában,

függőleges összevonásnál
: a legfelső cella `rowspan` attribútumában

van megadva. Az összevont cella tartalmát a bal szélső/legfelső cella elem tartalma adja meg. A többi cella HTML elemét nem is adjuk meg!

tabla21.html, tabla2.css

```
9   <table>
10    <tr><th rowspan="2">State of Health</th>
11        <th colspan="2">Fasting Value</th>
12        <th>After Eating</th></tr>
13    <tr><th>Minimum</th><th>Maximum</th>
14        <th>2 Hours After Eating</th></tr>
15    <tr><td>Healthy</td><td>70</td>
16        <td>100</td><td>Less than 140</td></tr>
17    <tr><td>Pre-Diabetes</td><td>101</td>
18        <td>126</td><td>140 to 200</td></tr>
19    <tr><td>Diabetes</td><td>More than 126</td>
20        <td>N/A</td><td>More than 200</td></tr>
21  </table>
```

State of Health	Fasting Value		After Eating
	Minimum	Maximum	2 Hours After Eating
Healthy	70	100	Less than 140
Pre-Diabetes	101	126	140 to 200
Diabetes	More than 126	N/A	More than 200

Készítse el az ábra alapján a `tabla22.html` fájlt! A cellák szegélyeinek megrajzolása érdekében

1. ágyazza be a következő sort a `<head>` elembe: `<link rel="stylesheet" type="text/css" href="tabla2.css" />`
2. Mentse ugyanabba a mappába a `tabla2.css` fájlt!

tabla22.html, tabla2.css

Country List		
Country Name or Area Name	**ISO ALPHA 2 Code**	**ISO ALPHA 3 Code**
Afghanistan	AF	AFG
Aland Islands	AX	ALA
Albania	AL	ALB
Algeria	DZ	DZA
American Samoa	AS	ASM
Andorra	AD	AND
Angola	AO	AGO

Opcionális elemek, a `<table>`-be ágyazva:

`<caption>`
: Táblázat címét adja meg (felül, középre igazítva)

`<colgroup>`
: Oszlopcsoport kialakítása, hogy az oszlopok azonos módon formázhatóak legyenek

`<col>`
: `<colgroup>`-ba ágyazandó, az oszlop formázásának megadásához. Azonosan formázandó szomszédos oszlopoknak elég egy ilyen elem, a `span` attribútum adja meg az oszlopok számát.

További opcionális elemek:

`<thead>`
: A fejléc sorainak (`<tr>`) egységbe zárására.

`<tbody>`
: A törzs részt zárja egybe.

`<tfoot>`
: A lábléc sorait zárja egybe.

Ezek a részek egységesen formázhatók, hosszú táblázatoknál a fej/láb minden oldalon újra kinyomtatható, esetleg a törzs görgethető.
Sorrend fontos: `<caption>` → `<colgroup>` → `<thead>` → `<tbody>` → `<tfoot>`

tabla31.html

```
<table>
  <caption>Calories</caption>
  <colgroup>
    <col class="health" />
    <col span="2" class="fasting" />
    <col class="after" />
  </colgroup>
  <thead>
    <tr><th rowspan="2">State of Health</th>
        <th colspan="2">Fasting Value</th>
        <th>After Eating</th></tr>
    <tr><th>Minimum</th><th>Maximum</th>
        <th>2 Hours After Eating</th></tr>
  </thead>
  <tbody>
    <tr><td>Healthy</td><td>70</td>
        <td>100</td><td>Less than 140</td></tr>
    <tr><td>Pre-Diabetes</td><td>101</td>
        <td>126</td><td>140 to 200</td></tr>
    <tr><td>Diabetes</td><td>More than 126</td>
        <td>N/A</td><td>More than 200</td></tr>
  </tbody>
</table>
```

Calories

State of Health	Fasting Value		After Eating
	Minimum	Maximum	2 Hours After Eating
Healthy	70	100	Less than 140
Pre-Diabetes	101	126	140 to 200
Diabetes	More than 126	N/A	More than 200

tabla31.css

```
table { border-collapse: collapse; }
thead { background-color: navy;
        color: white; }
thead th { border-color: yellow; }
td, th { border: 1px solid black; }
.health {
  background-color: lightgray; }
.fasting {
  background-color: lightcoral; }
.after {
  background-color: lightgreen; }
```

Készítse el a `tabla22.html` átalakításával az ábra alapján a `tabla32.html` fájlt! A megfelelő formázás érdekében

1. cserélje le a korábbi `<link>` elemet a következőre:
   `<link rel="stylesheet" type="text/css" href="tabla32.css" />`
2. Mentse ugyanabba a mappába a `tabla32.css` fájlt!
3. Az első oszlop `<col>` elemének `class` attribútuma legyen `country`, az utolsó kettőé `code` értékű!

tabla32.html, tabla32.css

Country Codes

Country List		
Country Name or Area Name	**ISO ALPHA 2 Code**	**ISO ALPHA 3 Code**
Afghanistan	AF	AFG
Aland Islands	AX	ALA
Albania	AL	ALB
Algeria	DZ	DZA
American Samoa	AS	ASM
Andorra	AD	AND
Angola	AO	AGO

Táblázat beágyazható egy másik táblázat (fejléc) cellájába.
Előnye: a cellák mérete, oszlopok száma táblázatonként eltérő lehet.

a1	a2	
b1	b2	
a.1	a.2	a.3
b.1	b.2	b.3
c.1	c.2	c.3

tabla4.html, tabla4.css

```
<table>
  <tr><td>
    <table>
      <tr><td>a1</td><td>a2</td></tr>
      <tr><td>b1</td><td>b2</td></tr>
    </table>
  </td></tr>
  <tr><td>
    <table>
      <tr><td>a.1</td><td>a.2</td><td>a.3</td></tr>
      <tr><td>b.1</td><td>b.2</td><td>b.3</td></tr>
      <tr><td>c.1</td><td>c.2</td><td>c.3</td></tr>
    </table>
  </td></tr>
</table>
```

Háromféle felsorolás (lista) létezik a HTML-ben:

1. Számozatlan
2. Számozott
3. Definíciós

Rendkívül rugalmasan formázható CSS-ből, pl. weboldalak menürendszere is kialakítható

- Számozatlan felsorolás: `<ul>` (unordered list) elemmel
- Ennek elemei: beágyazott `<li>` (list item) elemekkel

bevasarlas.html

```
8   <p>Mit kell venni a boltban?</p>
9   <ul>
10    <li>Sajt</li>
11    <li>Sör</li>
12    <li>Saláta</li>
13  </ul>
```

Mit kell venni a boltban?

- Sajt
- Sör
- Saláta

Feladat: készítsen a mellékelt ábrának megfelelően egy számozatlan felsorolást tartalmazó weblapot! (Forrás: Bűvös Szakács)

sor.html

Witbier sör hozzávalói, 21l végtérfogathoz

- 2 kg Bohemian Pilsner maláta (Weyermann 3 EBC)
- 0,1 kg Müncheni maláta (8L)
- 0,3 kg Acidulated (savas) maláta
- 2 kg búzapehely
- 0,5 kg zabpehely

- Számozott felsorolás: `<ol>` (ordered list) elemmel, attribútumai:

  `type` felsorolásjel típusa

Att. érték	Felsorolásjel
1	Arab számok (alapértelmezés)
A	Latin nagybetűk
a	Latin kisbetűk
I	Nagybetűs római számok
i	Kisbetűs római számok

  `start`
  : Az első elem sorszáma

  `reversed`
  : Csökkenő sorrendet ír elő

- Ennek elemei: beágyazott `<li>` (list item) elemekkel

futurama.html (Futurama)

```
8    <ol type="1" start="1">
9      <li>HOME</li>
10     <li>SWEET</li>
11     <li>GOTO 1</li>
12   </ol>
```

1. HOME
2. SWEET
3. GOTO 1

Kiindulva a jaegermeister.txt fájlból, hozza létre az ábrán látható HTML fájlt!

jaegermeister.html

Die Toten Hosen: Zehn kleine Jägermeister (részletek)

10. kleine Jägermeister rauchten einen Joint
    Den einen hat es umgehauen, da waren's nur noch neun
9. kleine Jägermeister wollten gerne erben
    Damit es was zu erben gab, musste einer sterben
8. kleine Jägermeister fuhren gerne schnell
    Sieben fuhren nach Düsseldorf und einer fuhr nach Köln
7. kleine Jägermeister waren beim Rendezvous
    Bei einem kam ganz unverhofft der Ehemann hinzu
6. kleine Jägermeister wollten Steuern sparen
    Einer wurde eingelocht, fünf durften nachbezahlen
5. kleine Jägermeister wurden kontrolliert
    Ein Polizist nahm's zu genau, da waren sie noch zu viert
4. kleine Jägermeister bei der Bundeswehr
    Sie tranken um die Wette, den Besten gibt's nicht mehr
3. kleine Jägermeister gingen ins Lokal
    Dort gab's zwei Steaks mit Bohnen und eins mit Rinderwahn
2. kleine Jägermeister baten um Asyl
    Einer wurde angenommen, der andere war zu viel

Többszintű felsorolások

- <li> elem belsejébe újabb felsorolás ágyazható
- A számozás újrakezdődik az első szinten → CSS

1. Tea
    1. Fekete
    2. Zöld
2. Kávé
    - Bécsi pörkölés
    - Francia pörkölés

tobbszintu.html

```
<ol>
  <li>Tea
    <ol>
      <li>Fekete</li>
      <li>Zöld</li>
    </ol>
  </li>
  <li>Kávé
    <ul>
      <li>Bécsi pörkölés</li>
      <li>Francia pörkölés</li>
    </ul>
  </li>
</ol>
```

- Definíciós lista létrehozása <dl> (description list) elemmel
- A kifejezés megadása <dl>-be ágyazott <dt> (term) elemmel
- Magyarázat az ezt követő <dd> (description) elemben

froccs.html

```
8   <dl>
9     <dt>Kisfröccs</dt>
10    <dd>1 dl bor + 1 dl szódavíz</dd>
11    <dt>Nagyfröccs</dt>
12    <dd>2 dl bor + 1 dl szódavíz</dd>
13    <dt>Hosszúlépés</dt>
14    <dd>1 dl bor + 2 dl szódavíz</dd>
15  </dl>
```

Kisfröccs
    1 dl bor + 1 dl szódavíz
Nagyfröccs
    2 dl bor + 1 dl szódavíz
Hosszúlépés
    1 dl bor + 2 dl szódavíz

Hozza létre az ábrán látható HTML fájlt!

betuszavak.html

HTML
    Hypertext Markup Language
CSS
    Cascading Style Sheets
SVG
    Scalable Vector Graphics

Egy HTML oldal megjelenítése egy másik oldalban: `<iframe>` elemmel

Attribútumok:

`src` (source)
: A keretbe betöltendő dokumentum URL-je

`width`
: Keret szélessége képpontban

`height`
: A keret magassága képpontban

További attribútumok:

`name`
: Ez azonosítja a keretet, amibe pl. új tartalom tölthető egy <a> elemmel, ha annak `target` attribútuma a `name` értékét tartalmazza

`srcdoc`
: Megjelenítendő dokumentum HTML kódja (magasabb prioritású, mint `src`, ha támogatott)

`sandbox`
: Megjelenítési környezet korlátozásainak feloldása (`allow-forms`, `allow-pointer-lock`, `allow-popups`, `allow-same-origin`, `allow-scripts`, `allow-top-navigation`) részletek

Megjegyzések

- Az `srcdoc`-ot csak az újabb böngészők támogatják
- A webszerverek az X-Frame-Options HTTP válasz fejléccel kérhetik az ezt támogató böngészőktől, hogy ne engedjék a lapot `<iframe>`-be tölteni.
- `src+sandbox` biztonságos korszerű böngészőkben, de nem biztonságos a sandbox-ot nem támogatókban!
- `srcdoc+sandbox` biztonságos korszerű böngészőkben, és nem működik (=biztonságos) az elavultakban

iframe.html

```
8     <p><a href="https://uni.sze.hu/kezdolap
        ↪ " target="belso">
9       Széchenyi István Egyetem</a></p>
10    <p><a href="https://www.edutus.hu/"
        ↪ target="belso">
11      Edutus Egyetem</a></p>
12    <iframe name="belso" width="400" height
        ↪ ="300"
13      srcdoc="<p>Válasszon egyetemet!</p>"
14      src="http://xenia.sze.hu/index.html">
          ↪ </iframe>
```

Széchenyi István Egyetem

Edutus Egyetem

Válasszon egyetemet!

A <head> tartalmazza a HTML oldal metaadatait. HTML5-től elhagyható, de javasolt használni. Beágyazható elemek:

<title>
: A dokumentum címe, kötelező.

<style>
: CSS stílusok, formázás megadása; ált. jobb külön fájlba helyezni, ld. később

<base>
: A relatív URL-ek a `href` értéke alapján lesznek értelmezve. A `target` más elemek `target` attribútumának alapértelmezett értékét adja meg.

`<link>`

Külső erőforrás és a dokumentum kapcsolatát adja meg. Jellemző alkalmazásai:

- Stíluslap meghatározása: `href`-ben a CSS fájl URL-je, `rel` (relationship) *stylesheet*, a `type` *text/css* értékű.
- Ikon (favicon = favorite icon) beállítás: `href`-ben az ikon URL-je, `rel` *icon*, a `type` pl. *image/svg+xml* értékű. (További részletek.)

`<meta>`

HTTP fejlécek kulcs (`http-equiv` attribútum) - érték (`content` attribútum) párok formájában történő megadására. Jellemző kulcsok:

- *content-type*, a MIME típus és karakterkódolás megadására: `content="text/html; charset=UTF-8"` → HTML5-től: csak a `charset="UTF-8"` attribútummal
- *refresh*, automatikus újratöltés, pl. percenként: `content="60"`

`<meta>`

Metaadatok kulcs (`name` attribútum) - érték (`content` attribútum) párok formájában történő megadására. Jellemző kulcsok:

- *description*, weboldal általános leírása
- *keywords*, kulcsszavak keresőmotoroknak az oldal tartalmához kapcsolódóan
- *author*, szerző
- *viewport*, nézetablak beállítás, `content="width=device-width, initial-scale=1.0"`. Probléma: mobil eszközök nagy felbontásúak, de kis méretűek, számítógép-kijelzőre optimalizált oldalak gyenge felhasználói élménnyel használhatók. *width=device-width*: a nézetablak szélessége alkalmazkodik az eszköz szélességéhez. *initial-scale=1.0* nagyítás kezdeti értéke. Részletek

`<script>`
: JavaScript programok megadására; előnyösebb a <body> végébe tenni (DOM felépül, az oldalbetöltést a JS kód nem lassítja).

`<noscript>`
: JavaScript támogatás hiányában a közrezárt szöveget megjeleníti. HTML5-től a body-ba is kerülhet.

HTML5-től a `<html>`, `<head>` és <body> elemek elhagyhatók, de ezt nem ajánljuk.

fejresz.html (fejresz/fejresz.css, fejresz/fejresz.js, fejresz/fejresz2.html)

```
3   <head>
4     <title>Fejrész</title>
5     <base href="fejresz/" target="_blank" />
6     <link rel="stylesheet" type="text/css" href="fejresz.css" />
7     <link rel="icon" type="image/png" href="html5_64.png" />
8     <meta charset="utf-8" />
9     <meta http-equiv="refresh" content="60" />
10    <meta name="description" content="Fejrész elemeinek bemutatása" />
11    <meta name="keywords"
12      content="HTML, head, title, base, link, meta, script, noscript" />
13    <meta name="author" content="Hatwagner F. Miklós" />
14    <meta name="viewport"
15      content="width=device-width, initial-scale=1.0" />
16  </head>
```

fejresz.html

```
17  <body>
18    <p>Kattintson <a href="fejresz2.html">ide</a>!</p>
19    <script src="fejresz.js"></script>
20    <noscript>Az Ön böngészője nem támogatja a JavaScript
21      futtatását.</noscript>
22  </body>
```

Kattintson ide!

Kattintson ide!

A macska.txt fájlból kiindulva készítse el a macska.html oldalt az ábrának megfelelően!

1. Az oldal címe legyen *Macska*!
2. Az oldalt formázza meg a macska.css stíluslap segítségével!
3. Jelenítse meg a macska.png fájlt ikonként (favicon)!
4. A dokumentum kódolása UTF-8 szerint történjen!
5. Készítsen *ismertetőt*, adjon meg *kulcsszavakat* a keresőmotorok számára! Adja meg a saját nevét *szerzőként*!
6. A nézetablak szélességét igazítsa a megjelenítő szélességéhez, a nagyítás legyen 1x-es!
7. Szúrja be a macska képét (macska.jpeg)!

A HTML szemantikus elemei az oldal funkcionális részeinek jelölésére:

`<main>`

A dokumentum legfőbb tartalmát jelöli, ami nem ismétlődik más oldalakon, azaz nem tartalmazza pl. a menüsort, oldal logot, szerzői jogi információt. Csak egyszer fordulhat elő a dokumentumban! Nem lehet az `<article>`, `<aside>`, `<footer>`, `<header>`, `<nav>` leszármazottja. Célja: akadálymentesítés, Safari olvasó funkciója is ezt emeli ki.

`<nav>`

Az oldal legfontosabb navigációs hivatkozásainak gyűjteménye, pl. menü, tartalomjegyzék. A menüt gyakran CSS formázott `<ul>` elemekkel valósítják meg. Több `<nav>` is lehet egy oldalon, pl. külön az oldalon belüli, és azon kívülre mutató hivatkozásoknak.

`<article>`

Az oldal többi részétől függetlenül, önmagában is értelmes tartalmi rész, pl. fórum- vagy blogbejegyzés, hír. Jellemzően címsorral kezdődik. Egy oldalon belül többször is előfordulhat, akár egymásba is ágyazhatók (pl. blogbejegyzés és az arra adott reakciók). A megjelenés dátumát gyakran tartalmazza beágyazott `<time>` elemmel (a `datetime` attribútummal gépek is felismerik).

`<section>`

Dokumentum egy része, fejezete, de akár fejléce, lábléce is lehet. Ajánlott címsorral ellátni. Nem lehet az `<address>` leszármazottja.

`<aside>`
: A tartalomhoz lazán kapcsolódó kiegészítés, megjegyzés. Akadálymentesítési okokból használják a `role` attribútumot.

`<header>`
: Általában a dokumentum bevezetőjét, navigációs hivatkozásokat tárol. Gyakran tartalmaz címsor (`<h1>`-`<h6>`) elemeket, logot, szerzőt. Többször is előfordulhat a dokumentumban, pl. több `<article>` elejében.

`<footer>`
: Egy dokumentum vagy fejezet lábléce. Jellemzően a szerző nevét, szerzői jogi információt, kapcsolatfelvétel módját (ld. `<address>`), oldaltérképet, impresszumot, stb. tartalmaz. Többször is előfordulhat egy dokumentumban (pl. minden `<article>` végén).

reszek.html

```
7   <body>
8     <header>
9       <h1>Web technológiák</h1>
10      <p>A korszerű webalkalmazások a HTML, CSS és JS technológiákon
11       alapulnak.</p>
12      <p>Megjelenés dátuma:<time datetime="2019-11-06␣16:00">2019.
13       november 6.</time></p>
14    </header>
15    <nav>
16      <a href="#html">HTML</a> |
17      <a href="#css">CSS</a> |
18      <a href="https://en.wikipedia.org/wiki/JavaScript">JS</a>
19    </nav>
```

reszek.html

```
20      <main>
21        <article id="html">
22          <header>
23            <h2>HTML</h2>
24            <p>A web leírónyelve</p>
25          </header>
26          <p>A HTML (angolul: HyperText Markup Language=hiperszöveges

34          SGML alapú XML leíró nyelven alapul.</p>
35          <section>
36            <h3>Használata</h3>
37            <p>HTML általában szöveges állományokban található meg olyan
```

reszek.html

```
47          stb.), valamint egyéb eszközök, például mobiltelefon.</p>
48        <aside role="note">
49          <p>A legnépszerűbb webböngészők a <a
50          href="https://www.google.com/chrome/">Google Chrome</a>, a
51          <a href="https://www.mozilla.org/hu/firefox/new/">Mozilla
52          Firefox</a> és a <a
53          href="https://www.microsoft.com/hu-hu/windows/microsoft-
            ↪ edge">
54          Microsoft Edge</a>.</p>
55        </aside>
56      </section>
```

reszek.html

```
    </main>
    <footer>
      <p>Készítette: <a
      href="mailto:miklos.hatwagner@sze.hu?subject=Web%20technológia">
      Dr. Hatwágner F. Miklós</a>. Minden jog fenntartva.</p>
      <address>
        Széchenyi István Egyetem<br />
        Informatika Tanszék<br />
        Győr, Egyetem tér 1. B602<br />
        H-9026<br />
      </address>
    </footer>
  </body>
```

Tartalom megjelenítése / elrejtése

`<details>`
: Interaktív oldalrész, amit a felhasználó elrejthet/megjeleníthet (alapértelmezetten rejtett; megjeleníthető az `open` attribútummal). Gyakorlatilag bármi beleágyazható.

`<summary>`
: A blokk kattintható fejléce, mindig látszik.

reszletek.html

Most vetítjük:

▾ Az aszfalt királyai

Ford v. Ferrari / Le Mans `66
amerikai életrajzi dráma, akciófilm, 152 perc, 2019

Az Oscar-díjas Matt Damon és Christian Bale a főszereplői az igaz történet alapján készült filmnek, melyben a látnoki erővel bíró autótervező, Carroll Shelby (Damon) és a vakmerő, brit születésű autóversenyző, Ken Miles (Bale) együtt küzdenek meg a vállalati beavatkozással, a fizika törvényeivel és saját démonjaikkal, hogy megépíthessenek egy forradalmi versenyautót a Ford Motor Companynek, amivel legyőzhetik Enzo Ferrari autóit az 1966-os Le Mans-i 24 órás versenyen.

▸ Terminátor: Sötét végzet

reszletek.html

```
9     <details open="open">
10      <summary>Az aszfalt királyai</summary>
11      <p>Ford v. Ferrari / Le Mans '66<br />
12       amerikai életrajzi dráma, akciófilm, 152 perc, 2019</p>
13      <p>Az Oscar-díjas Matt Damon és Christian Bale a főszereplői az
14       igaz történet alapján készült filmnek, melyben a látnoki erővel

20       1966-os Le Mans-i 24 órás versenyen.</p>
21    </details>
```

A c64.txt fájl felhasználásával készítse el a `c64.html` fájlt!

- Az újság neve és a rovatok kerüljenek a dokumentum fejlécébe!
- Készítsen navigációs sávot az *index*, *c64.com* és *Wikipédia* elemekből!
- Az oldal *fő* része tartalmazza a teljes cikket!
- A cikknek is legyen fejléce, ami a *cikk címéből*, *szerzőjéből* és a *megjelenés idejéből* áll!
- Az *„Egy maroknyi dollárért”* és a *„Specifikáció”* legyen a cikk két fejezetének címe!
- A *„Microprocesszor”* és *„Video hardver...”* kattintásra jelenjen meg/tűnjön el!
- Az *„Önök írták”* legyen az „oldalsáv” fejléce!

c64.html

index.hu

Tech, hardver

index | C64.com | Wikipedia

Idén 25 éves a Commodore 64

Szerző: Stöckert Gábor

Megjelent: 2007.12.29. 16:18

1982-ben a szocialista blokktól nyugatra a játékteren akkori csodától - Pac Man, Galaga és társai - lenyűgözött gyerekek ZX Spectrumot, Apple II-t vagy Atari 2600-at kértek a jézuskától. Az igazán előrelátók azonban Commodore 64-ért nyúztak a szülőiket: nemcsak azért, mert az akkor pár hónapja piacon levő gép által produkált kép és hang sokkal alkalmasabb volt játékra, mint a konkurens masináké, hanem azért is, mert a későbbi csak C64 és C=64 rövidfilmekkel illetett kütyű jóval olcsóbb volt a rivális számítógépeknél, ezért a felmenők ellenkezését is könnyebb volt legyűrni. Az Index tisztelegő a legszakszerűbb otthoni computer előtt.

Természetesen ős is karácsonya kaptam a C64-emet, bár csak őt vagy hat évvel a pénz debitálása után. Fater napokig számolgatta, hogy mikként lesz a Martaháférstrassén a családi vagyonból hűtőláda és tíj cipők mellett még számítógép is, és hogyan ösztatja meg, hogy az anyósát is vinni kelljen a fejenként kirbbeti schillinglimit miatt. Végül é lemondott a cipőről, és ezzel mindjárt perbbémat megoldotta: a kirakatán örökre "beszelünk magyarul" felirattal kérkedő osztrák műszaki boltból egy Commodore 64 utazott velünk haszág a negyedik ülésen.

Egy maroknyi dollárért

Akkor a platformot már újra gyártották: 1985-ben ugyanis a Commodore cég többször leállította a modell gyártását, mert azt gondolta, hogy az amerikai piac már nem nagyon veszi fel több C64-et. Európában azonban még csak ekkor futott be igazán a gép, megtermelve a videojáték-fejlesztés iparszerűvé válásához szükséges kritikus tömeget. Végül 1994-ben gördült le az utolsó C64 a futószalagról: hé egy évtized alatt 30 millió adtak el belőle, amivel maig tartja a legszakszerűbb számítógép Guinness-rekordját.

A fényes karriert egy jó üzleti érzékkel megáldott, agresszív cégvezető, Jack Tramiel indította el. Tramiel 1954-ben alapította a Commodore-t, és eleinte írógépek javításával és irodabútorokkal foglalkozott. Erdeklődött az elektronika ujdonságai iránt, ezért később a cég kínálatába bekerültek a digitális órák, a szövegszerkesztő gépek, és az első üzleti számítógépek. A hetvenes évek második felében Tramiel már saját computer közösítésén gondolkozott. 1976-ban megvásárolta a MOS Technologies nevű cégvgártót. 1980-ban pedig bemutatta a VIC-20-at, a Commodore 64 elődjét. A gép sikeres volt – pár év alatt egymillió darabot vittek belőle –, de legnagyobb haszna abban rejlett, hogy Tramiel felismerte a gépre készült játékok korlátait.

Specifikáció

- Microprocesszor
  - MOS 6510 vagy MOS 8500 (a 6510/8500 valójában egy módosított 6502-es egy integrált 6 bites I/O porttal, a 8500-as pedig a 6510-es CMOS technológiával készült változata)
  - Órajel: 0,985 MHz (PAL) / 1,023 MHz (NTSC)
- Video hardver: MOS VIC-II, MOS 6567/8567 (NTSC), MOS 6569/8569 (PAL)
  - Szöveges mód: 40×25, 16 szín
  - Grafikus módok: 160×200 (egy karakternyi helyen maximum 4 színt tartalmazhat), 320×200 (monokróm)
  - 8 hardveres sprite (mozgatható grafikus elem, játékprogramokhoz)

Önök írták

"Neki köszönhetem, hogy ma ott tartok ahol. A sok meggyilkolt joystick után rájöttem, hogy én is akarok ilyen programokat írni. Hát írtam. Ma is. Csak ma már más platformon, más célból :). Köszönöm C64! Isten áldassék!"

"A táp bordája megrohadt, annyit ment hétvégenként. :) Háromcsukólós (tárcsás optokapus házi készítésű robotkart is hajtottunk vele egyszer. Mintán a pályaadatokat beadtuk, indításkor a monitorvezérlést is kikapcsolta a program, hogy bírja. Kiszőtt, de menti! Csupa vicces emlék."

Számítógépes programok írásához kapcsolódó elemek:

`<kbd>` (keyboard)
: Felhasználói adatbevitelre, pl. gépelésre, szóbeli parancsra van szükség.

`<samp>` (sample)
: Egy program lehetséges kimenetét mutatja.

`<code>`
: Programkód jelölésére. Nem őrzi meg a tördelést → `<pre>`

`<var>` (variable)
: Matematikai kifejezés, vagy program változóinak jelölésére.

Amennyiben a derékszögű háromszög befogóit *a* és *b* jelöli, a *c* átfogó hossza Pitagorasz tétele szerint a következő JavaScript függvénnyel határozható meg:

```
function atfogo(a, b) {
    return Math.sqrt(a*a + b*b);
}
```

Működését kipróbálhatja a konzol ablakban, miután előhívta a Webfejlesztő eszköztárat az `F12` megnyomásával. Ha a függvényt meghívja az `atfogo(3,4)` paranccsal, megjelenik az eredmény: `5`.

kod.html

```
8     <p>Amennyiben a derékszögű háromszög befogóit <var>a</var>
9     és <var>b</var> jelöli, a <var>c</var> átfogó hossza Pitagorasz
10    tétele szerint a következő JavaScript függvénnyel határozható meg: </p>
11    <pre><code>
12  function atfogo(a, b) {
13      return Math.sqrt(a*a + b*b);
14  }
15    </code></pre>
16    <p>Működését kipróbálhatja a konzol ablakban, miután
17    előhívta a Webfejlesztő eszköztárat az <kbd>F12</kbd>
18    megnyomásával. Ha a függvényt meghívja az
19    <code>atfogo(3,4)</code> paranccsal, megjelenik az eredmény:
20    <samp>5</samp>.</p>
```

Induljon ki a vizsga.txt fájlból, és készítse el ebből azt a HTML fájlt, ami a mellékelt ábrának megfelelően jelöli

- a program változóinak nevét,
- a függvény kódját,
- a lista (lehetséges válaszok) elemeit!

vizsga.html

Mi lesz az *uj* karakterlánc tartalma az alábbi C/C++ függvény hívását követően, ha *regi* tartalma "Mari foz, Mari sut, Mari mosogat.", *mit* értéke "Mari", *mire* értéke pedig "Julcsa"?

```
void nevek(const char* regi, char* uj,
           const char* mit, const char* mire) {
  const char *eleje = regi;
  char *vege;
  int mitHossz = strlen(mit);
  while((vege = strstr(eleje, mit)) != NULL) {
    strncat(uj, eleje, vege-eleje);
    strcat(uj, mire);
    eleje = vege + mitHossz;
  }
  strcat(uj, eleje);
}
```

Jelölje meg a helyes választ!

a. "Mari foz, Mari sut, Mari mosogat."
b. "Marisa foz, Marisa sut, Marisa mosogat."
c. "Julcsa foz, Julcsa sut, Julcsa mosogat."
d. "Julcsa foz, Julcsa sut, Julcsa"
e. Semmi, a program lefagy.
f. Semmi, a program nem fordítható.

Nem lehet minden karaktert közvetlenül begépelni, mert

- Böngésző félreértelmezheti, pl. < (cimke eleje), " (attribútum érték jelölése)
- ezért nem teljesíti a szabvány és a validator elvárásait sem
- Nincs megfelelő billentyű

Megoldás: HTML entitások, megadhatóak

- névvel (könnyen megjegyezhető): `&mnemonik;`, pl. `&`
- karakterkészletbeli kódjával (jobb támogatás):
  - Tizes számrendszerben: `&#kod;`, pl. `&`
  - Tizenhatos számrendszerben: `&#xhexakod;`, pl. `&`

Referencia

Kar.	Leírás	Mnemonik	Kód
	Nem törhető szóköz (Non-Breaking SPace)	nbsp	160
<	Kisebb, mint (Less Than)	lt	60
>	Nagyobb, mint (Greater Than)	gt	62
&	Kereskedelmi és (AMPersand)	amp	38
"	Idézőjel (QUOTe)	quot	34
€	Euro	euro	8364
©	Szerzői jogvédelem (COPYright)	copy	169
∀	Minden (FOR ALL)	forall	8704
∈	Eleme	isin	8712
Α	Alfa	Alpha	913
→	Jobbra (Rightwards ARRow)	rarr	8594

entitas.html

```
8     <h1>HTML oldal szerkezete</h1>
9     <pre>
10 &lt;!DOCTYPE html&gt;
11 &lt;html lang=&quot;hu-HU&quot;&gt;
12   &lt;head&gt;
13     &lt;title&gt;Az oldal címe&lt;/title&gt;
14     &lt;!--esetleges további fejléc-információk--&gt;
15   &lt;/head&gt;
16   &lt;body&gt;
17     &lt;p&gt;első bekezdés&lt;/p&gt;
18     &lt;p&gt;második bekezdés&lt;/p&gt;
19   &lt;/body&gt;
20 &lt;/html&gt;
21 </pre>
```

HTML oldal szerkezete

```
<!DOCTYPE html>
<html lang="hu-HU">
  <head>
    <title>Az oldal címe</title>
    <!--esetleges további fejléc-információk-->
  </head>
  <body>
    <p>első bekezdés</p>
    <p>második bekezdés</p>
  </body>
</html>
```

Web-technológia

HTML, II. rész

Dr. Hatwagner F. Miklós

Széchenyi István Egyetem, Győr

https://github.com/wajzy/GKxB_INTM049.git

2020. szeptember 30.

SZÉCHENYI EGYETEM
UNIVERSITY OF GYŐR

INFORMATIKA TANSZÉK
DEPARTMENT OF COMPUTER SCIENCE

Űrlapok

Vastag kliens alkalmazásokhoz hasonlóan adatok gyűjthetőek űrlapokkal, amiket

- feldolgoztathatunk a szerver oldalon (most ez lesz), vagy
- JavaScript programmal a kliensen.

Űrlap vezérlők a `<form>` elembe ágyazva használhatók. Legfontosabb attribútumai:

`action`
: A szerver oldali feldolgozó script URL-je

`target`
: Hova töltse a szerver válaszát (`_blank`, `_self` (alapért.), `_parent`, ...)

`novalidate`
: Ne végezzen input ellenőrzést a böngésző

`autocomplete`
: Korábban megadottak alapján kiegészíti az adatokat (`on` (alapért.), `off`)

`method`

Adattovábbítási módszer:

`get`

Alapértelmezett. URL lekérdező karakterláncában továbbítja az adatokat.

- Az URL betehető a könyvjelzők közé, és
- a böngészőben is könnyen szerkeszthető

`post`

HTTP tranzakcióként továbbít.

- Nincsenek méretkorlátok (URL hossza korlátozott, $\approx$ 2000 karakter)
- Fájlok csak így továbbíthatók
- A böngészőben nem látszik, de ettől még nyílt szövegként továbbítják a hálózaton!

`enctype` (type of encoding)

Meghatározza a küldött adatok kódolását. Kötelező megadni, ha a `method` attr. értéke `post`. Lehetséges értékek:

`application/x-www-form-urlencoded`
: Alapértelmezett. Minden szóközt és speciális karaktert helyettesít.

`multipart/form-data`
: Fájlok küldésekor kell használni.

`text/plain`
: Csak a szóközöket helyettesíti.

A legtöbb vezérlő az `<input>` elemmel hozható létre, pl. egy számbeviteli mező néhány attribútuma, és hatása:

`type`
: `number`, számbeviteli mező létrehozása

`name`
: A szerver oldalon ez lesz az adat *kulcsa*

`min`
: A legkisebb bevihető érték

`max`
: A legnagyobb bevihető érték

`step`
: Ennyit változtatnak a léptetőgombok/kurzurvezérlő nyilak az értéken, ennyi többszöröseit lehet megadni, illetve ha értéke `any`, akkor bármilyen racionális szám megadható

`required`
: A mező kitöltése kötelező

Űrlapbeküldő gomb: `<input>` elem

- `type="submit"`
- `value` attr. adja meg a gomb feliratát

Cimke létrehozása vezérlőhöz: `<label>` elemmel. A cimkére kattintva a vezérlő is aktiválódik (pl. rádiógombnál kiválasztás). Kapcsolat cimke és vezérlő között:

- vezérlő a cimke belsejébe ágyazva
- a cimke `for` attribútumában megadható a vezérlő `id`-je

Vezérlők logikai csoportosítása: `<fieldset>` elemmel
Csoport feliratának megadása: `<legend>` elemben

Adja meg egy háromszög oldalhosszait!

A: 2

B: 2

C: 2

Számolás

urlap1.html (urlap1.php)

```
8     <form action="http://xenia.sze.hu/~wajzy/html/urlap1.php">
9       <fieldset>
10        <legend>Adja meg egy háromszög oldalhosszait!</legend>
11        <div><label>A: <input name="a" type="number" min="0"
12          step="any" required="required" /></label></div>
13        <div><label for="b0ldal">B: </label>
14          <input id="b0ldal" name="b" type="number" min="0"
15            step="any" required="required" /></div>
16        <div><label>C: <input name="c" type="number" min="0"
17          step="any" required="required" /></label></div>
18        <div><input type="submit" value="Számolás" /></div>
19      </fieldset>
20    </form>
```

A legtöbb vezérlő az `<input>` elemmel és annak `type` attribútumával állítható elő. Az összes vezérlővel használható főbb attribútumai:

`autocomplete`
: Automatikus kiegészítés a korábban begépelt tartalom alapján. (`on`, `off`)

`autofocus`
: Oldalbetöltés után automatikusan fókuszba kerül

`disabled`
: Letiltott vezérlő

`form`
: Ha a vezérlő nincs `<form>` elembe ágyazva, akkor annak `id`-je alapján logikailag összekapcsolható vele.

`id`
: A vezérlőt egyedileg azonosítja a kliens oldalon (pl. JavaScript programokban)

`name`
: A vezérlőt azonosítja a szerver oldalon (kulcs)

`readonly`
: Csak olvashatóvá teszi a vezérlőt

`required`
: A mező kitöltése kötelező

`value`
: A vezérlő értéke

Egysoros szövegbeviteli mező: `type="text"` (alapértelmezés)

`list`
: Egy `<datalist>` elem (ajánlatok legördülő listája) `id`-jét tartalmazza

`maxlength`
: A begépelhető karakterek száma

`pattern`
: Reguláris kifejezés az érték ellenőrzéséhez. Hiba esetén a `title` értéke magyarázza az elvárt formátumot.

`placeholder`
: Helyőrző, ami az első karakter begépeléséig segít a felhasználónak, pl. „A rendszámot kötőjel nélkül, nagybetűkkel adja meg!”

`size`
: A vezérlő szélessége átlagos karakterszélességben mérve (CSS jobb)

A `<datalist>` elembe `<option>` elemek ágyazhatók, melyek gépelést segítő legördülő lista elemeiként jelennek meg.
A (`<select>` elemben is használható) `<option>` attribútumai:

attribútum nélkül
: Az elem tartalma jelenik meg a legördülő listában, és ezt továbbítják a szervernek is

`label`
: A listában egy rövidebb szöveg jelenhet meg, mint az elem értéke

`selected`
: Kiválasztottá teszi az elemet az oldal betöltésekor (csak `<select>`-nél)

`value`
: Ha megadják, ennek tartalmát küldi a szervernek (vagy másolja a szövegmezőbe), nem a megjelenő szöveget

Reguláris kifejezések dióhéjban

Általános forma: /minta/módosítók

Input ellenőrzésre csak a minta szolgál; ha a vezérlő értéke nem illeszkedik az általános mintára, az űrlapot nem lehet beküldeni.

- Az általános karakterek önmagukat reprezentálják.
- A metakarakterek önmagukban is halmazt vagy vezérlőjelet testesítenek meg, pl. \d a számjegyek, \s a fehér karakterek, \n az újsor, stb.
- Halmaz elemei közül elég egynek illeszkednie, pl. [abc] elfogadja az abc első három betűje közül bármelyiket.
- Halmaz intervallummal is megadható, pl. [a-z] a kisbetűk halmaza, de a két módszer kombinálható is, pl. [a-zA-Z] a betűk halmaza.
- Az [^abc] minden jelre illeszkedik, ami nem a, b, és c.
- Alternatívák is létrehozhatók (a|b) formában.

Mennyiségek szabályozása

- `n+` legalább egy előfordulást vár el
- `n*` 0 vagy több előfordulást követel
- `n?` 0 vagy 1 előfordulás
- `n{X}` pontosan X előfordulás
- `n{X,Y}` legalább X, legfeljebb Y előfordulás
- `n{X,}` legalább X előfordulás

Pl. egy rendszámot ellenőriz a `[A-Za-z]{3}-\d{3}` minta. További minták.

További részletek

Űrlapok

Név: Pl. Dr. Prof. Remek Elek
Neptun kód:
Kedvenc böngészőprogram:
Beküldés

Név: Kiss Miska
Neptun kód: aBc123
Kedvenc böngészőprogram: Google Chrome
Beküldés

Név: Kiss Miska
Neptun kód: aBc12!
Kedvenc böngészőprogram: Google Chrome
Beküldés

urlap2.html (urlap2.php)

```
9       <div><label>Név: <input name="nev" type="text" size="32"
10        placeholder="Pl. Dr. Prof. Remek Elek" required="required" />
11        </label></div>
12      <div><label>Neptun kód: <input name="neptun"
13        pattern="[a-zA-Z0-9]{6}" maxlength="6"
14        title="a kód pontosan 6 alfanumerikus jelből áll" />
15        </label></div>
```

urlap2.html (urlap2.php)

```
16        <div><label>Kedvenc böngészőprogram: <input name="bongeszo"
17          list="programok" /></label></div>
18        <datalist id="programok">
19          <option value="ie" label="MS␣Internet␣Explorer">
20            Microsoft Internet Explorer</option>
21          <option value="edge">Microsoft Edge</option>
22          <option selected="selected">Google Chrome</option>
23          <option>Mozilla Firefox</option>
24          <option>Apple Safari</option>
25        </datalist>
```

Jelszóbeviteli mező: `type="password"`
Nagyon hasonló a `text` típushoz, de a karaktereket „elrejti” (viszont nem titkosítja!)

Színválasztó: `type="color"`
A `value` értéke `#RRGGBB` formátumú.

Dátumválasztó: `type="date"`
A `value` értéke *éééé-hh-nn* formátumú. Használhatóak még a `min`, `max` attribútumok.

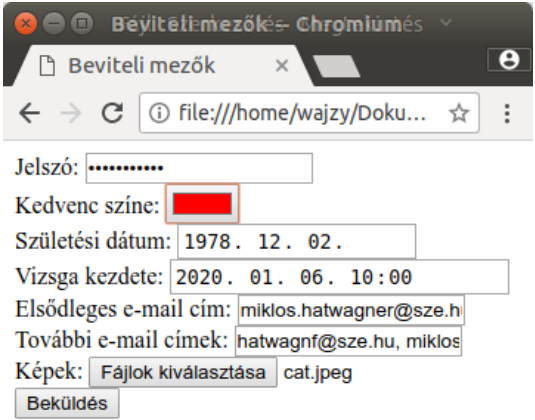
Dátum és időválasztó (időzóna nélkül): `type="datetime-local"`
A `value` értéke *éééé-hh-nnTóó:pp* formátumú.

E-mail cím megadása: `type="email"`
Automatikusan ellenőrzi formailag. Több cím is megadható vesszőkkel elválasztva, ha megadják a `multiple` attribútumot.

Rejtett mező: `type="hidden"`
Egyáltalán nem jelenik meg, de egy (pl. JavaScript kóddal előállított) kulcs (`name`) - érték (`value`) pár felküldhető a szerverre.

Fájlok feltöltése: `type="file"`
Több fájl feltölthető a `multiple` attribútum megadásával. Korlátozható a tallózható fájlok köre, pl. `accept=".jpg"` csak JPEG fájlokat enged, egy MIME-típus, mint pl. a `accept="video/*"` mindenféle videót, stb. Korlátozható a fájlok mérete is:
`<input type="hidden" name="MAX_FILE_SIZE" value="`*méret*`" />`
A kódolás típusa legyen `enctype="multipart/form-data"`, a HTTP metódus `method="POST"`!

Űrlapok


Beviteli mezők - Chromium és
Beviteli mezők
file:///home/wajzy/Doku...
Jelszó:
Kedvenc színe:
Születési dátum: 1978. 12. 02.
Vizsga kezdete: 2020. 01. 06. 10:00
Elsődleges e-mail cím: miklos.hatwagner@sze.h
További e-mail címek: hatwagnf@sze.hu, miklos
Képek: Fájlok kiválasztása cat.jpeg
Beküldés

urlap3.html (urlap3.php)

```
8   <form action="http://xenia.sze.hu/~wajzy/html/urlap3.php"
9     enctype="multipart/form-data" method="post">
10    <div><label>Jelszó: <input name="jelszo" type="password"
11      maxlength="16" required="required" /></label></div>
12    <div><label>Kedvenc színe: <input name="szin" type="color"
13      value="#FF0000" /></label></div>
14    <div><label>Születési dátum: <input name="datum" type="date"
15      min="1900-01-01" max="2020-12-31" /></label></div>
16    <div><label>Vizsga kezdete: <input name="datumido"
17      type="datetime-local" /></label></div>
```

urlap3.html (urlap3.php)

```
18        <div><label>Elsődleges e-mail cím: <input name="e-mail"
19          type="email" /></label></div>
20        <div><label>További e-mail címek: <input name="e-mailek"
21          type="email" multiple="multiple" /></label></div>
22        <input name="rejtett" value="titok" type="hidden" />
23        <input type="hidden" name="MAX_FILE_SIZE" value="1000000" />
24        <div><label>Képek: <input name="kepek[]" type="file"
25          multiple="multiple" accept="image/*" /></label></div>
26        <div><input type="submit" value="Beküldés" /></div>
27      </form>
```

Készítse el az alábbi űrlapot! (urlap4.html)

Az adatokat küldje a `http://xenia.sze.hu/~wajzy/html/urlap3.php` címre!

Adja meg kedvenc filmjének adatait!

Cím: Toy Story 4.

Megtekintés tervezett időpontja: 2019-11-24T20:00

Játékidő (perc): 100

Borító: Tallózás... Nincs kijelölve fájl.

Küldés

A szerver a következő kulcs-érték párokat kapta a HTTP tranzakció során:

```
Array
(
    [cim] => Toy Story 4.
    [vetites] => 2019-11-24T20:00
    [jatekido] => 100
    [MAX_FILE_SIZE] => 1000000
)
```

Év és hónap kiválasztás: `type="month"`
A `value` értéke *éééé-hh* formátumú. Használhatóak még a `min`, `max` attribútumok.

Év és azon belül egy hét kiválasztása: `type="week"`
A `value` értéke *éééé-Whh* formátumú.

Idő megadása: `type="time"`
A `value` értéke *óó:pp* formátumú.

Nyomógomb: `type="button"`
Önmagában semmire nem jó, JavaScript kódok futtatására használják.

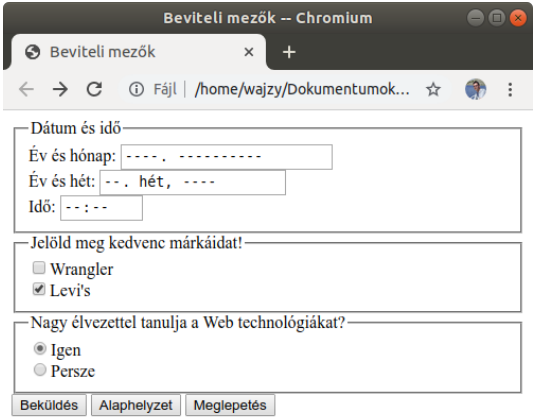
Űrlap alaphelyzetbe állítása: `type="reset"`
Az űrlap betöltése utáni állapotot állítja vissza; ha véletlenül kattintanak rá, minden addigi munka elvész!

Jelölőnégyzet: `type="checkbox"`
Egymástól független választási lehetőségekhez.

Rádiógomb: `type="radio"`
Egy rádiógomb-csoporton belül csak egy elem lehet kiválasztva. A felhasználó nem tudja az összes elemet ki nem választottá tenni. Csoportképző módszer: azonos `name` attribútum értékek.

Mindkét típusra igaz, hogy egy elemet kiválasztani a `checked` attribútummal lehet. A `value` attribútumot rádiógomboknál „kötelező” megadni, mert különben csak az `on` értéket kapja meg a szerver → a gomb azonosíthatatlan. A ki nem választott elemeknek még a kulcsa sem látszik a szerveren.

Űrlapok


Beviteli mezők -- Chromium
Beviteli mezők
Fájl
/home/wajzy/Dokumentumok...
Dátum és idő
Év és hónap: ----. ----------
Év és hét: --. hét, ----
Idő: --:--
Jelöld meg kedvenc márkáidat!
Wrangler
Levi's
Nagy élvezettel tanulja a Web technológiákat?
Igen
Persze
Beküldés
Alaphelyzet
Meglepetés

urlap5.html

```
11        <div><label>Év és hónap: <input name="evho" type="month" />
12          </label></div>
13        <div><label>Év és hét: <input name="evhet" type="week" />
14          </label></div>
15        <div><label>Idő: <input name="ido" type="time" />
16          </label></div>

19        <legend>Jelöld meg kedvenc márkáidat!</legend>
20        <div><label><input name="wrangler" type="checkbox"
21          value="wrangler" />Wrangler</label></div>
22        <div><label><input name="levis" type="checkbox"
23          checked="checked" />Levi's</label></div>
```

urlap5.html

```
27        <div><label><input name="web" type="radio"
28          value="igen" />Igen</label></div>
29        <div><label><input name="web" type="radio"
30          checked="checked" />Persze</label></div>

34        <input type="reset" value="Alaphelyzet" />
35        <input type="button" id="meglepetes" value="Meglepetés" />
```

Beküldés gomb képpel helyettesítve: `type="image"`
`src` adja meg a képet, `width` a szélességet, `height` a magasságot, `alt` a helyettesítő szöveget. Beküldi a kattintás helyét is (`x`, `y` kulcsok).

A `submit` és `image` vezérlők néhány közös attribútuma:

- `formaction`: a `<form>` `action` attribútumának értékét írja felül; pl. két gomb két külön helyre küldheti az adatokat.
- `formenctype`, `formmethod`, `formnovalidate`, `formtarget` hasonlóan, a `<form>` attribútum értékeinek felülírására szolgál.

Csúszka: `type="range"`
Alapértelmezetten [0, 100] közti érték kiválasztására, ahol a pontos érték nem számít.
Adatbevitel szabályozható a `min`, `max`, `step` attribútumokkal.

Keresőmező: `type="search"`
Adatok keresésére egy oldalon.

Telefonszám: `type="tel"`
A bevitt szám ellenőrizhető a `pattern` segítségével.

URL: `type="url"`
A böngésző formai ellenőrzést végezhet vagy speciális gombokat tehet a virtuális billentyűzetre (pl. *.com*, *www.*, ...)

Űrlapok

Csúszka:

Keresés:

Telefon: Pl. +36 20 123-4567

URL: https://github.com/wajzy/GKxB_IN

A szerver a következő kulcs-érték párokat kapta a lekérdező karakterláncban:

```
Array
(
    [csuszka] => 127
    [kereso] =>
    [telefon] =>
    [weblap] => https://github.com/wajzy/GKxB_INTM049
    [x] => 64
    [y] => 65
)
```

urlap6.html (urlap6-submit.png, urlap2.php)

```
9       <div><label>Csúszka: <input name="csuszka" type="range"
10        min="0" max="255" step="1" value="127" /></label></div>
11      <div><label>Keresés: <input name="kereso" type="search" />
12        </label></div>
13      <div><label>Telefon: <input name="telefon" type="tel"
14        placeholder="Pl.␣+36␣20␣123-4567" pattern="[+]?[\d\s/-]*" />
15        </label></div>
16      <div><label>URL: <input name="weblap" type="url"
17        value="https://github.com/wajzy/GKxB_INTM049" />
18        </label></div>
19      <div><input type="image" src="urlap6-submit.png"
20        width="125" height="121" alt="Beküldés"
21        formaction="http://xenia.sze.hu/~wajzy/html/urlap2.php" />
22        </div>
```

Vezérlők, amiket *nem* az `input` elemmel hozunk létre:

`<select>`

Legördülő lista, melynek bejegyzéseit `option` elemekkel adhatjuk meg. Attribútumai:

`size`

Ennyi elem fog látszani egyszerre egymás alatt (görgethető doboz)

`multiple`

Több elem is kiválasztható egyszerre (Ctrl, Shift) (PHP: a `name` értéke `[]`-re végződjön!)

`<optgroup>`

Sok lehetőség esetén csoportok képzésére: `<select>` → `<optgroup>` → `<option>`

`label`

Csoport (ki nem választható) felirata

`<textarea>`

Hosszabb szövegek megadására. Még a címkék közötti fehér karaktereket is belefoglalja a vezérlő tartalmába! Attribútumok:

rows

Sorok száma

cols

Oszlopok száma

`<button>`

Hasonló, mint `<input type="button">`, de HTML/CSS formázható tartalommal.

type

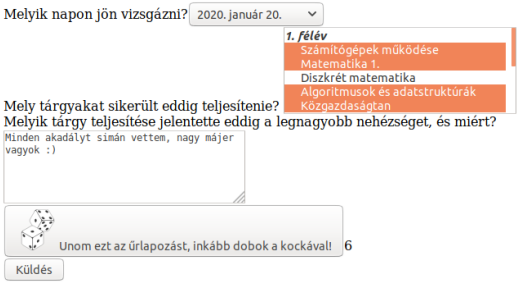
A gomb típusa, lehet `button`, `submit` és `reset`.

`<output>`

JavaScript számítások eredményének jelölésére. Nincs látványos formázása.

`for`

Azon `<input>` elemek `id`-i szóközzel elválasztva, melyekből kiindulva az eredmény megszületett.




```
Array
(
    [vizsga] => 2020. január 20.
    [elotanulmanyok] => Array
        (
            [0] => GKNB_INTM012
            [1] => GKNB_MSTM001
            [2] => GKNB_MSTM016
            [3] => KGNB_NETM042
        )

    [sz_nehez] => Minden akadályt simán vettem, nagy májer vagyok :)
)
```

urlap7.html (dice.svg, urlap2.php)

```
9       <div><label>Melyik napon jön vizsgázni?<select name="vizsga">
10        <option>2019. december 16.</option>
11        <option>2020. január 6.</option>
12        <option selected="selected">2020. január 20.</option>
13        <option>2020. január 27.</option>
14      </select></label></div>
```

urlap7.html

```
15        <div><label>Mely tárgyakat sikerült eddig teljesítenie?
16          <select name="elotanulmanyok[]" multiple="multiple" size="6">
17            <optgroup label="1.␣félév">
18              <option value="GKNB_INTM012">Számítógépek működése</option>

22              <option value="KGNB_NETM042">Közgazdaságtan</option>
23            </optgroup>
24            <optgroup label="2.␣félév">
25              <option value="GKNB_INTM001">Rendszer és irányítás</option>

29              <option value="GKNB_MSTM008">Matematika 2.</option>
30            </optgroup>
31        </select></label></div>
```

urlap7.html

```
32      <div><label for="k_nehez">Melyik tárgy teljesítése jelentette
33        eddig a legnagyobb nehézséget, és miért?</label><br />
34        <textarea id="k_nehez" name="sz_nehez" rows="5" cols="40">
            ↪ Minden akadályt simán vettem, nagy májer vagyok :)</
            ↪ textarea>
35      </div>
36      <div><button type="button" id="kocka">
37        <img src="dice.svg" alt="Dobókocka" width="50" height="50" />
38        Unom ezt az űrlapozást, inkább dobok a kockával!
39        </button><output name="sz_dobas" id="k_dobas" for="kocka">
40      </output></div>
```

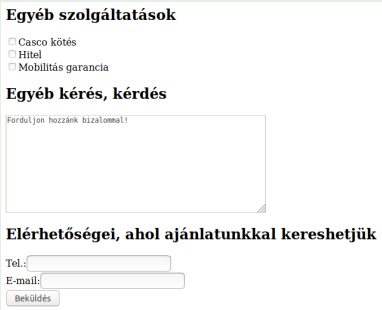
Űrlapok

A mellékelt ábrának megfelelően hozzon létre egy weboldalt! Az adatokat küldje a `http://xenia.sze.hu/~wajzy/html/urlap2.php` címre!

- Az *Extrák* közül többet is lehessen választani!
- A választható *színek* legyenek: *fehér*, *szürke*, *fekete*, *vörös*, *kék*, *sárga*. Alapértelmezetten legyen kiválasztva a *vörös*!
- A *fényezés* típusa alapértelmezetten legyen *normál*!

urlap8.html

Az oldal teteje



Az oldal alja

A HTML5-ben beépülő modulok (pl. Flash) és akár programozás nélkül lehet videót lejátszani a `<video>` elemmel (de a böngészők codec támogatása hiányos). Használható formátumok:

- MP4 (video/mp4, legjobb böngésző támogatás)
- WebM (video/webm)
- Ogg (video/ogg)

Ha biztosra akarunk menni: publikálás több formátumban is.
Formátumok közötti konvertálás: pl. MiroVideoConverter
A lejátszás programozható a Media API-val

Ha a <video> elem nem támogatott egy böngészőben, a címkék közötti szöveg jelenik meg. Opcionális attribútumok:

`autoplay`
: A lejátszás azonnal indul; nem ajánlott, zavarhatja a felhasználót

`controls`
: Vezérlő gombokat jelenít meg

`width`, `height`
: A lejátszó ablak szélessége, magassága képpontokban; ajánlott megadni

`loop`
: Végtelenített lejátszás

`muted`
: Némítás

`poster`
: Egy kép, amit a betöltés alatt / lejátszás megkezdéséig lát a felhasználó. Érték: URL

`preload`
: Adatfolyam betöltési módja. Érték: `auto` | `metadata` | `none`.

`src`
: Videó forrása. Nem ajánlott a használata, mert csak egyetlen forrás nevezhető meg, amit valószínűleg nem támogat minden böngésző. Érték: URL.

video1.html

```
8   <video src="http://clips.vorwaerts-gmbh.de/big_buck_bunny.mp4"
9     controls="controls" width="640" height="360">
10    <p>Az Ön böngészője nem támogatja a video lejátszást, de a
11    fájlt megpróbálhatja letölteni <a
12    href="http://clips.vorwaerts-gmbh.de/big_buck_bunny.mp4">innen</a
        ↪ >.</p>
13  </video>
```

Több adatforrás is megadható beágyazott `<source>` elemekkel, melyek közül a böngésző az első támogatott formátumhoz tartozót fogja választani. Attribútumok:

`src`
: Adatforrás. Érték: URL

`type`
: A forrás MIME típusa.

A `<video>` elemmel történő kísérletezéshez egy érdekes eszköz.

video2.html

```
8     <video controls="controls" width="640" height="360">
9       <source src="http://clips.vorwaerts-gmbh.de/big_buck_bunny.mp4"
10      type="video/mp4" />
11      <source src="http://clips.vorwaerts-gmbh.de/big_buck_bunny.ogv"
12      type="video/ogg" />
13      <source src="http://clips.vorwaerts-gmbh.de/big_buck_bunny.webm"
14      type="video/webm" />
15      <p>Az Ön böngészője nem támogatja a video lejátszást, de a
16      fájlt megpróbálhatja letölteni
17      <a href="http://clips.vorwaerts-gmbh.de/big_buck_bunny.mp4">innen</a>,
18      <a href="http://clips.vorwaerts-gmbh.de/big_buck_bunny.ogv">innen</a>
19      vagy
20      <a
21      href="http://clips.vorwaerts-gmbh.de/big_buck_bunny.webm">innen</a>.</p>
22    </video>
```

A videók feliratozhatóak is a `<track>` elemmel. Felirat formátum: VTT. Online szerkesztő, átalakító. Attribútumok:

`default`
: Kijelölhető több feliratsáv közül az alapértelmezett.

`kind`
: Feliratsáv típusa: `captions` | `chapters` | `descriptions` | `metadata` | `subtitles` (ez az alapértelmezés).

`label`
: Feliratsáv címkéje, pl. a felirat nyelve.

`src`
: A felirat forrása, kötelező. Érték: URL

`srclang`
: Felirat nyelvének ISO 639-1 kódja, pl. hu.

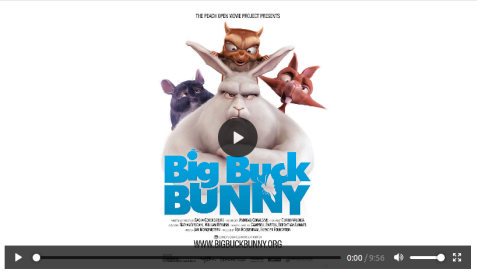
video3.html (subtitle.vtt)

```
8   <video controls="controls" width="640" height="360">
9     <source src="http://clips.vorwaerts-gmbh.de/big_buck_bunny.mp4"
10    type="video/mp4" />
11    <source src="http://clips.vorwaerts-gmbh.de/big_buck_bunny.ogv"
12    type="video/ogg" />
13    <source src="http://clips.vorwaerts-gmbh.de/big_buck_bunny.webm"
14    type="video/webm" />
15    <track src="subtitle.vtt" srclang="hu" label="Magyar"
16    default="default" />
17    <p>Az Ön böngészője nem támogatja a video lejátszást, de a
18    fájlt megpróbálhatja letölteni
19    <a href="http://clips.vorwaerts-gmbh.de/big_buck_bunny.mp4">innen</a>,
20    <a href="http://clips.vorwaerts-gmbh.de/big_buck_bunny.ogv">innen</a>
21    vagy <a
22    href="http://clips.vorwaerts-gmbh.de/big_buck_bunny.webm">innen</a>.</p>
23  </video>
```

Feladat: készítse el a Big Buck Bunny webes lejátszóját!

- A videó felbontása 800x450 képpont.
- Poszter fotó: `https://upload.wikimedia.org/wikipedia/commons/thumb/c/c5/Big_buck_bunny_poster_big.jpg/800px-Big_buck_bunny_poster_big.jpg`
- Két adatforrás is van, ezeket kell felajánlani: `https://download.blender.org/peach/bigbuckbunny_movies/big_buck_bunny_480p_stereo.ogg` és ugyanezen az útvonalon `big_buck_bunny_480p_h264.mov` néven.
- Feliratsáv nem áll rendelkezésre.
- Biztosítsa a letöltés lehetőségét, ha a böngésző nem támogatja a lejátszást!

video4.html



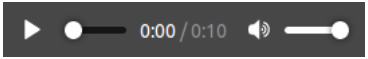
Az <audio> elemmel lehet hangokat, zenét lejátszani. Szabványos formátumok:

- MP3 (audio/mpeg, legjobb böngésző támogatás)
- Wav (audio/wav)
- Ogg (audio/ogg)

Attribútumai a méretektől eltekintve azonosak a <video>-éival.
Itt is célszerű az adatforrást beágyazott <source> elemekkel megadni, és a lejátszást nem támogató böngészőknél hibaüzenetet megjeleníteni.

audio1.html (dog.ogg, dog.mp3)

```
8    <audio controls="controls">
9      <source src="dog.mp3" type="audio/mpeg" />
10     <source src="dog.ogg" type="audio/ogg" />
11     <p>Az Ön böngészője nem támogatja a hang lejátszást, de a
12     fájlt megpróbálhatja letölteni
13     <a href="dog.mp3">innen</a>, vagy
14     <a href="dog.ogg">innen</a>.</p>
15   </audio>
```



Blokkszintű és soron belüli elemek

- A blokkszintű (*block-level*) elemek mindig új sorban kezdődnek, és vízszintesen kitöltik a rendelkezésre álló helyet. Sokkal találkoztunk már:

  `<p>`, `<h1>`-`<h6>`, `<nav>`, `<main>`, `<article>`, `<section>`, `<header>`, `<footer>`, `<aside>`, `<address>`, `<blockquote>`, `<figure>`, `<figcaption>`, `<form>`, `<fieldset>`, `<ul>`, `<ol>`, `<li>`, `<dl>`, `<dt>`, `<dd>`, `<hr>`, `<pre>`, `<table>`, `<video>`.

- Néhánnyal ez az oktatóanyag nem foglalkozik, pl.

  `<canvas>`, `<noscript>`.

- Egy, a későbbiekben fontos, más elemek csoportosítására (konténer) és formázására (→ CSS) használt, szemantikai töltettel nem rendelkező elem: `<div>`.

- A soron belüli (*inline*) elemek nem kezdődnek új sorban, és csak annyi helyet foglalnak el, amennyit a tartalom mérete indokol. Sokkal találkoztunk már:

  `<a>`, `<abbr>`, `<em>`, `<strong>`, `<b>`, `<i>`, `<big>`, `<small>`, `<sup>`, `<sub>`, `<br>`, `<label>`, `<input>`, `<output>`, `<button>`, `<select>`, `<textarea>`, `<bdo>`, `<cite>`, `<q>`, `<code>`, `<var>`, `<samp>`, `<kbd>`, `<img>`, `<map>`, `<time>`.

- Néhány, főleg elavult vagy más területekre átvezető elemmel ez az oktatóanyag nem foglalkozik, pl.

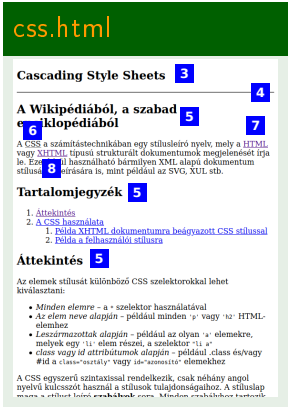
  `<acronym>`, `<object>`, `<tt>`, `<script>`.

- Egy, a későbbiekben fontos, soron belüli részek (adatok, pl. szöveg és más soron belüli elemek) megjelölésére és formázására (→ CSS) használt, szemantikai töltettel nem rendelkező elem: `<span>`.

A blokkszintű és soron belüli elemek megkülönböztetése a HTML 4.x szabványok sajátossága volt. Ezeket leváltották a HTML5 tartalmi kategóriái, de bizonyos párhuzamosságok továbbra is megfigyelhetőek (pl. *inline* → *phrasing content*), és a régi fogalmak a korszerű oldalak készítésénél is többnyire egyértelműen és kifejező módon használhatóak.

Összefoglaló feladatok

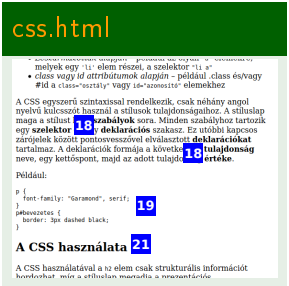
Készítse el a CSS wiki oldala által ihletett weblapot!



1. Induljon ki a css.txt fájlból!
2. Készítsen ebből magyar nyelvű HTML5 dokumentumot, UTF-8 karakterkódolással!
3. Az oldal első szintű címsora és a böngészőfülön megjelenő szöveg egyaránt legyen „Cascading Style Sheets”!
4. Ezt válassza el egy vízszintes vonallal...
5. ...a második szintű címsorként jelölt sortól („A Wikipédiából, . . . ”)! A „Tartalomjegyzék” és az „Áttekintés” szintén 2. szintű címsorként legyen jelölve!
6. Az első bekezdés elején álló „CSS”-t jelölje meg rövidítésként! Ha valaki fölötte tartja az egeret, jelenjen meg a „Cascading Style Sheets, magyarul: lépcsőzetes stíluslapok” szöveg egy buborékban!
7. Jelölje meg hivatkozásként a „HTML” szót, ami a megfelelő Wikipédia oldalra mutat!
8. Járjon el ugyanígy az „XHTML”-lel is, de azt új fülön nyissa meg!

Összefoglaló feladatok

9. A dokumentum tartalma alkosson egy cikket!
10. Minden, ami a tartalomjegyzék felett van, legyen fejlécként jelölve!
11. Külön szakaszként jelölje a tartalomjegyzéket, és az azt követő fő fejezeteket („Áttekintés”, „A CSS használata”)!
12. A tartalomjegyzék pontjait egymásba ágyazott, számozott listák alkossák!
13. Ezek egyúttal legyenek hivatkozások a dokumentumon belüli fejezetekre!
14. Az „Áttekintés” fejezetben hozzon létre nem számozott felsorolást!
15. A kötőjel előtti részt jelölje meg hangsúlyos szövegrészként!
16. A kötőjeleket cserélje közepes hosszúságú gondolatjelekre!
17. A felsorolás pontjaiban szereplő *-ot, HTML elemneveket, stb. jelölje meg programkódként!



18 A felsorolást követő szövegben a fontos fogalmakat (szabályok, szelektor, deklarációs, ...) jelölje meg félkövérrel szedendő szövegként!

19 A példaként szereplő CSS szöveget jelölje meg előformázott szövegrésznek!

20 A CSS kód töredékeit ágyazza be olyan, előredefiniált jelentéssel nem bíró soron belüli elembe, melynek `class` attribútuma utal a kód azon részének szemantikai töltetére! A `p` elemneveknél az attribútum értéke legyen `element`, a `bevezetes`-nél `id`, a `font-family` és `border` tulajdonságnál `property`, a kettőspont utáni részeknél `keyword`, kivéve a `"Garamond"`-ot, amit jelöljön `string`-nek, a 3-at, amit `quantity`-nak, a `px`-et pedig `unit`-nak!

21 Jelölje meg 2. szintű címsorként „A CSS használata” sort!

Összefoglaló feladatok

22 Jelölje meg a bekezdéseket!

23 A fejezet első bekezdésében szereplő `h2`-t jelölje meg programkódként!

24 A következő, minta kód legyen előformázott szövegnek jelölve! Figyeljen rá, hogy minden karakter megjelenjen, és ne kezdje értelmezni azokat a böngészőprogram!

25 Jelölje meg a „Példa XHTML dokumentumra...” és „Példa a felhasználói...” sorokat 3. szintű címsornak!

26 Jelölje meg a fájlnevet dőlt betűs szövegként!

27 A CSS szövegrészt jelölje előformázott szövegként!

Készítse el a szamla.txt szövegének felhasználásával az alábbi számlát!

- A dokumentum legyen UTF-8 kódolású, a böngészőfülön álljon *Számla* felirat!
- Kapcsolja össze a HTML fájlt a szamla.css stíluslappal!
- Helyezzen el a HTML fejrészében olyan leírást, amely a „Számla, táblázatok gyakorlásához” szöveget tartalmazza!
- A számlát egy külső táblázatból, és az annak fejrészébe, törzsébe, lábrészébe beágyazott táblázatokból lehet összeállítani. Ügyeljen az általános és fejléc cellák közötti különbségekre is!

szamla.html

Számla

Szállító		Vevő			
Vicik&Vacak Bt. 1234 Kukutyin, Fő u. 3. Bankszámla: 12345678-12345678 Adóig. szám: 12345678-1-12		John Connor 5678 Salakszentmotoros, Mellék u. 72.			
Megrendelés száma, kelte	Teljesítés kelte	Számla kelte	esedékessége	sorszáma	
	2013.09.27	2013.09.27	2013.09.27	12345/2013	
Fizetés módja: készpénz					
Késedelmi kamatunk a mindenkori jegybanki alapkamat kétszerese.					

Megnevezés, VTSZ, SZJ	Me.	Mennyiség	Egys. ár.	Érték	%	ÁFA	Bruttó
Tricorder	db	2	25.000	50.000	27	13.500	63.500
Fluxuskondenzátor	db	1	1.000.000	1.000.000	27	270.000	1.270.000
HAL 9000	db	1	10.000.000	10.000.000	27	2.700.000	12.700.000

AFA alapja: 11.050.000	értéke: 2.983.500	Fizetendő: 14.033.500
azaz tizennégymillió-harmincháromezer-ötszáz		
	Átadó	Átvevő

Készítse el az űrlapot!

- A dokumentum legyen UTF-8 kódolású, a böngészőfülön álljon *Űrlap* felirat, szövegét jelölje meg magyarként!
- Az űrlap tartalmát küldje `POST` módszerrel a `http://xenia.sze.hu/~wajzy/feldolgozo.php` címre!
- A beviteli mezők előtt álló magyarázó feliratok legyenek a vezérlővel összekapcsolt címkék!
- A „Név” és „Életkor” első betűit jelenítse meg félkövér betűkkel!
- Ezek a kiemelt betűk legyenek a vezérlő fókuszba kerülésének gyorsbillentyűi! (Többnyire `Alt+Shift+`*gyorsbillentyű*)
- Névként legfeljebb 64 karaktert lehessen bevinni, és jelenítsen meg helykitöltő szöveget arra az időre, amíg a felhasználó nem ad meg adatot!
- Az életkornál csak 0 és 120 közé eső számokat lehessen megadni, és kötelező kitölteni.

Összefoglaló feladatok

- Nyelvvizsgákból lehessen egyszerre többet is kijelölni! Angol és német nyelvből (ezek legyenek a csoportok címkéi) is lehet alap-, közép- és felsőfokú vizsgát megadni. Az angol középfok már az oldal betöltésekor legyen alapértelmezetten kiválasztva!
- Vegye körbe a rádiógombokat egy dobozzal, melyben a „Legmagasabb iskolai végzettsége” felirat olvasható!
- A rádiógombok egy csoportot alkotnak, az oldal betöltésekor a „Gimnázium/Középiskola” legyen kiválasztva!

Összefoglaló feladatok

- Készítse el a hobbyk csoportját a megfelelő felirattal!
- Álljanak ezek jelölőnégyzetekből!
- A bemutatkozás szövegbeviteli mezője 40 oszlop széles, 10 sor magas legyen! Legyen benne már egy sor szöveg!
- Az önéletrajznál lehessen több fájlt is feltölteni egyszerre!
- Helyezzen el az űrlap küldésére és alaphelyzetbe állítására szolgáló gombokat!

Összefoglaló feladatok

Az eredmény a megadott adatoktól, a vezérlők nevétől függően az alábbihoz hasonló lehet:

Általános űrlapadatok:

```
Array
(
    [n_nev] => Dr. Gép Elek
    [n_eletkor] => 42
    [n_nyelv] => Array
        (
            [0] => ak
            [1] => af
            [2] => na
        )

    [n_iskola] => tudomany
    [n_bicikli] => on
    [n_konyv] => on
    [n_bemutatkozas] => Nagyon szeretek hosszú HTML oldalakat készíteni.
)
```

Feltöltött fájl adatai:

```
Array
(
    [n_cv] => Array
        (
            [name] => mini.jpg
            [type] => image/jpeg
            [tmp_name] => /tmp/phpjvMyE7
            [error] => 0
            [size] => 207558
        )

)
```

Web-technológia

Cascading Style Sheets, I. rész

Dr. Hatwagner F. Miklós

Széchenyi István Egyetem, Győr

https://github.com/wajzy/GKxB_INTM049.git

2021. október 16.

SZÉCHENYI EGYETEM
UNIVERSITY OF GYŐR

INFORMATIKA TANSZÉK
DEPARTMENT OF COMPUTER SCIENCE

CSS: Cascading Style Sheets

- ≈ lépcsőzetes/sorba kapcsolt stíluslapok
- *formázás, megjelenés* leírásának elválasztása a *tartalomtól* (HTML), előnyei:
  - külön fájlban tárolható, ami több weboldalhoz is használható, így csökken az összesített kódméret,
  - egységessé válik ezen oldalak megjelenése,
  - egymástól függetlenül, egyidejűleg lehet szerkeszteni a formát és a tartalmat,
  - gyorsabban módosítható a megjelenés, mert csak egy helyen kell változtatni,
  - hatékonyabbá válik a gyorstárazás,
- különféle médiára eltérő formázás lehetséges (pl. képernyő, nyomtatás)
- a CSS ereje
- hivatalos W3C oldal

Elavult módszer (htmlFormazas.html)

```
1  <!DOCTYPE html>
2  <html lang="hu-HU">
3    <head>
4      <title>Elavult HTML formázás</title>
5      <meta charset="utf-8" />
6    </head>
7    <body>
8      <p><font size="+1"><i>Elavult HTML formázás</i></font></p>
9    </body>
10 </html>
```

Formázás CSS-sel (cssFormazas.html)

```
3   <head>
4     <title>CSS formázás</title>
5     <link rel="stylesheet" type="text/css" href="cssFormazas.css">
6     <meta charset="utf-8" />
7   </head>
8   <body>
9     <p class="kiemelt">CSS formázás</p>
10  </body>
```

Formázás CSS-sel (cssFormazas.css)

```
1 .kiemelt {
2   font-size: larger;
3   font-style: italic;
4 }
```

Szelektor (selector)

```
.kiemelt {
  font-size: larger;
  font-style: italic;
}
```

Érték (value)

Tulajdonság (property)

Deklaráció sablonja

```
szelektor {
  tulajdonság1: érték(ek);
  tulajdonság2: érték(ek);
  ...
  tulajdonságN: érték(ek);
}
```

Szelektor
Mit akarunk formázni?

Tulajdonság
Milyen tulajdonságán változtassunk?

Érték
Milyen legyen az új állapot?

Megjegyzések a CSS-ben:

- `/* megjegyzes */`
- végleges kódból célszerű elhagyni
- Lehet több soros is

CSS ellenőrző

JSFiddle - Code Playground

Böngésző Webfejlesztő menüje (F12), stílusszerkesztő

Szelektorok

HTML elem neve

```
p { font-style: italic; }
```

Egyedi azonosító (id attribútum) alapján

```
#lablec { font-size: 10pt; }
```

Az `id` nem kezdődhet számjegy karakterrel!

Univerzális szelektor, mindenre illeszkedik

```
* { font-size: smaller; }
```

Osztály (class attribútum alapján)

```
*.kisbetus { font-size: small; } /* bármilyen HTML elemhez */
.kisbetus { font-size: small; } /* bármilyen HTML elemhez, rövid alak */
p.voros { color: red; } /* csak adott (pl. <p>) HTML elemhez */
```

A class értéke nem kezdődhet számjeggyel, de lehet egyszerre több, szóközzel elválasztott értéke:

```
<p class="kisbetus voros">Apróbetűs piros bekezdés</p>
```

Elemek csoportosítása

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

egyszeruSzelektor1.html

```
3   <head>
4     <title>Egyszerű szelektorok</title>
5     <link rel="stylesheet" type="text/css" href="egyszeruSzelektor1.css">
6     <meta charset="utf-8" />
7   </head>
8   <body>
9     <article>
10      <header>
11        <h1>Ma van a magyar informatika születésnapja</h1>
12        <p class="forras␣bevezeto">Forrás: <a href="https://njszt.hu/hu/news
            ↪ /2020-01-21/ma-van-magyar-informatika-szuletesnapja">NJSZT</a>,
            ↪  <time datetime="2020-01-21␣00:00">2020. január 21.</time></p>
13      </header>
```

egyszeruSzelektor1.html

```
14      <p class="bevezeto">A magyarországi informatika kezdete a első
            ↪ elektronikus számítógép elkészültétől eredeztethető. A szobányi m
            ↪ éretű M–3–at szovjet tervek alapján építették meg, a magyar
            ↪ szakembergárda pedig ezen kísérletezte ki, hogy miként lehet egy
            ↪ számítógépet a népgazdaság érdekeit szolgáló mérnöki–, és gazdasá
            ↪ gi számításokra felhasználni – például az  Erzsébet–híd statikai
            ↪ számításait ellenőrizték vele.</p>
15      <p>A mintegy kétévnyi építést követően hivatalosan 1959. január 21–én á
            ↪ tadott <a href="https://itf.njszt.hu/termek_hardware/m-3-
            ↪ szamitogep">M–3 számítógépre</a> a <a href="https://ajovomultja.
            ↪ hu/news/60-eves-az-elso-magyar-szamitogep">korabeli sajtóbeszámol
            ↪ óban</a> még számológépként hivatkoztak, mert a számítógép szó ké
            ↪ sőbb honosodott meg a magyar nyelvben.</p>
```

egyszeruSzelektor1.css

```
h1, h2, h3, h4, h5, h6 {
  font-family: Arial, Sans-Serif;
}
p { font-family: Garamond, "Times New Roman", Serif; }
a { color: navy; }
.forras {font-style: italic; }
.bevezeto { font-weight: bold; }
```

Ma van a magyar informatika születésnapja

Forrás: NJSZT, 2020. január 21.

A magyarországi informatika kezdete a első elektronikus számítógép elkészültétől eredeztethető. A szobányi méretű M-3-at szovjet tervek alapján építették meg, a magyar szakembergárda pedig ezen kísérletezte ki, hogy miként lehet egy számítógépet a népgazdaság érdekelt szolgáló mérnöki-, és gazdasági számításokra felhasználni - például az Erzsébet-híd statikai számításait ellenőrizték vele.

A mintegy kétévnyi építést követően hivatalosan 1959. január 21-én átadott M-3 számítógépre a korabeli sajtóbeszámolóban még számológépként hivatkoztak, mert a számítógép szó később honosodott meg a magyar nyelvben.

Bár ún. nulladik generációs számítógépet már az M-3 előtt néhány évvel is építettek a Budapesti Műszaki Egyetemen, Kozma László MESz-1 nevű gépe még telefonközpontokban használatos elektromágneses relékből épült fel, az M-3-mal ellentétben még nem felelt meg a Neumann-elveknek, amelyek előírják, hogy egy számítógépnek teljes egészében elektronikus módon kell működnie, és egy belső tárban kell tárolnia a programutasításokat.

A Magyar Tudományos Akadémia Kibernetikai Kutatócsoportjában megépített, első generációs M-3-ast tudományos számítások embernél gyorsabb végrehajtására tervezték. Noha a gép csupán másodpercenként 30 számítási műveletet tudott elvégezni (míg a mai gépek processzorai esetében már több tízezer millió utasítással számolhatunk másodpercenként), az M-3-ra írt programok révén így is számos gyakorlati feladatra használták fel. E gépen készült az épülő Erzsébet híd statikai számításaival kapcsolatos program, de például szállítási költségek optimalizálásához, részecskék vizsgálatához szükséges számításokat is végeztek vele.

Egy elembe tetszőleges mélységben beágyazott másik elemek kiválasztása

leszarmazott.html

```
7        section p { background-color: yellow; }

11     <section>
12       <div>
13         <p>Első bekezdés.</p>
14         <p>Második bekezdés.</p>
15       </div>
16       <p>Harmadik bekezdés.</p>
17     </section>
18     <p>Negyedik bekezdés.</p>
```

Első bekezdés.

Második bekezdés.

Harmadik bekezdés.

Negyedik bekezdés.

Egy elembe közvetlenül beágyazott gyermek elemek kiválasztása

gyermek.html

```
7        section > p { background-color: yellow; }

11     <section>
12       <div>
13         <p>Első bekezdés.</p>
14       </div>
15       <p>Második bekezdés.</p>
16       <p>Harmadik bekezdés.</p>
17     </section>
18     <p>Negyedik bekezdés.</p>
```

Első bekezdés.

Második bekezdés.

Harmadik bekezdés.

Negyedik bekezdés.

Egy elemet közvetlenül követő testvér elem kiválasztása

testver.html

```
7        div + p { background-color: yellow; }

11     <section>
12       <p>Első bekezdés.</p>
13       <div>
14         <p>Második bekezdés.</p>
15       </div>
16       <p>Harmadik bekezdés.</p>
17       <p>Negyedik bekezdés.</p>
18     </section>
19     <p>Ötödik bekezdés.</p>
```

Első bekezdés.

Második bekezdés.

Harmadik bekezdés.

Negyedik bekezdés.

Ötödik bekezdés.

Egy elemet közvetlenül követő összes testvér kiválasztása

testver2.html

```
7        div ~ p { background-color: yellow; }

11     <section>
12       <p>Első bekezdés.</p>
13       <div>
14         <p>Második bekezdés.</p>
15       </div>
16       <p>Harmadik bekezdés.</p>
17       <p>Negyedik bekezdés.</p>
18     </section>
19     <p>Ötödik bekezdés.</p>
```

Első bekezdés.

Második bekezdés.

Harmadik bekezdés.

Negyedik bekezdés.

Ötödik bekezdés.

Látszólagos osztályok (pseudo-class): egy elem adott állapota esetén alkalmazandó formázása, referencia

Az egér alatti elem kiválasztása

hover.html

```
7    p:hover { background-color: yellow; }

11   <p>Mozgassa ide az egeret!</p>
```

Mozgassa ide az egeret!

Mozgassa ide az egeret!

Azon elemek kiválasztása, melyek a szülőjük első gyermekei

firstchild.html

```
7        p:first-child { background-color: yellow; }

10   <body>
11     <p>Első bekezdés.</p>
12     <p>Második bekezdés.</p>
13     <p>Harmadik bekezdés.</p>
14   </body>
```

Első bekezdés.

Második bekezdés.

Harmadik bekezdés.

Nyelvfüggő beállítások alkalmazása

lang.html

```
7        q:lang(hu) { quotes: "\0201E" "\0201D"; }

11     <p><q>Magyar idézet.</q></p>
12     <p><q lang="en-US">Angol idézet.</q></p>
```

„Magyar idézet.”

“Angol idézet.”

Adott típus n-edik előfordulása (részleteket ld. később `nth-child()`-nál)

nthoftype.html

```
7        li:nth-of-type(2n) { background-color: yellow; }

11     <ul>
12       <li>Első elem</li>
13       <li>Második elem</li>
14       <li>Harmadik elem</li>
15       <li>Negyedik elem</li>
16     </ul>
```

- Első elem
- Második elem
- Harmadik elem
- Negyedik elem

Látszólagos elemek (pseudo-elements): elemek bizonyos részeinek kiválasztása

Első sor kiválasztása: `::first-line`

- Csak blokkszintű elemekkel használható
- Csak bizonyos tulajdonságokkal használható:
  `font`, `word-spacing`, `letter-spacing`, `line-height`, `text-decoration`, `text-transform`, `vertical-align`, `color`, `background`, `clear`.

CSS3 előtt `:` állt ott, ahol most `::`.

Szöveg első betűjének kiválasztása: `::first-letter`

- Csak blokkszintű elemekkel használható
- Csak bizonyos tulajdonságokkal használható:
  `font`, `line-height`, `text-decoration`, `text-transform`, `color`, `background`, `margin`, `border`, `padding`, `vertical-align` (ha nincs lebegtetés), `float`, `clear`.

Kijelölt szöveg: `::selection`

- Csak bizonyos tulajdonságokkal használható:
  `color`, `background`, `cursor`, `outline`.

Tartalom valami előtt: `::before`, mögött: `::after`

Tartalom megadása: `content` tulajdonsággal, értéke:

- idézőjelek/aposztrófok közötti szöveg
- karakterek unicode kódja, pl. `'\02192'` $\equiv \rightarrow$
- `url()` függvénnyel adott kép

pseudoelements.html

```
7   h1::before {
8     content: "\02192";
9     color: red;
10  }
11  h1::after {
12    content: "\02190";
13    color: red;
14  }
```

```
15  p::first-line {
16    font-variant: small-caps;
17    font-weight: bold;
18  }
19  p::first-letter {
20    font-size: 200%;
21    color: blue;
22  }
23  p::selection {
24    color: red;
25    background-color: yellow;
26  }
```

```
30  <h1>Nemes Tihamér</h1>
31  <p>Nemes Tihamér (Budapest, 1895. április 29. – Budapest, 1960.
32  március 30.) feltaláló, gépészmérnök, MTA doktora, villamosmérnök,
33  a műszaki tudományok doktora, a kibernetika egyik hazai
34  úttörője.</p>
```

→Nemes Tihamér←

Nemes Tihamér (Budapest, 1895. április 29. - Budapest, 1960. március 30.) feltaláló, gépészmérnök, MTA doktora, villamosmérnök, a műszaki tudományok doktora, a kibernetika egyik hazai úttörője.

Szülei Nemes Mihály és Auguszt Mária voltak. Oklevelét 1917-ben Budapesten szerezte a Műegyetemen. Rövid ideig dolgozott a Lloyd repülőgépgyárnál. 1921-től a Telefonhírmondóhoz került, majd annak főmérnökének nevezték ki. 30 éves korában az Elektromos és Finommechanikai Rt. igazgatójává léphetett elő. 1929-től a Posta Kísérleti Állomásán dolgozott. Az 1930-as évek közepétől megkezdte a televíziózás elméleti és gyakorlati tanulmányozását. Részt vett az első hazai kísérletekben. 1938-ban szabadalmi bejelentést tett színes televízió megvalósítására. 1950-ben a Távközlési Kutató Intézet tudományos munkatársa.

Érdeklődési körébe tartoztak az: elektrotechnika, logika, szimuláció, modellezés, de leginkább a logikai gép. Nemes Tihamér sakkozó- és sakkfeladványokat megoldó gépei is elsősorban az emberi gondolkozás modellezésére szolgáltak. Számtalan televíziós szabadalma szimulálta az emberi szem funkcióit. Postamérnökként egyike volt azoknak, akik a magyar televíziózást megteremtették. 1953-ban készítette az első magyar kísérleti tv-adó berendezést.

pseudoelements.html

Kiválasztás attribútumok alapján

Adott attribútummal rendelkező elemek kiválasztása

attributum1.html

```
7       a[target] { background-color: yellow; }

12    <p><a href="https://hu.wikipedia.org/wiki/Nemes_Tihamér">Nemes Tihamér</a></p>
13    <p><a href="https://hu.wikipedia.org/wiki/Neumann_János" target="_blank">Neumann János</a></p>
```

Néhány hivatkozás híres magyar tudósok Wiki oldalaira

Nemes Tihamér

Neumann János

Pontosan az adott értékű attribútummal rendelkező elemek kiválasztása

attributum2.html

```
7       a[target="_blank"] { background-color: yellow; }

12    <p><a href="https://hu.wikipedia.org/wiki/Nemes_Tiham%C3%A9r" target="_self">
          ↪ Nemes Tihamér</a></p>
13    <p><a href="https://hu.wikipedia.org/wiki/Neumann_J%C3%A1nos" target="_blank">
          ↪ Neumann János</a></p>
```

Néhány hivatkozás híres magyar tudósok Wiki oldalaira

Nemes Tihamér

Neumann János

Adott teljes szót (is) tartalmazó értékű attribútummal rendelkező elemek kiválasztása

attributum3.html

```
7   a[title~="Nemes"] { background-color: yellow; }

12  <p><a href="https://hu.wikipedia.org/wiki/Nemes_Tiham%C3%A9r" title="Nemes Tihamér">Nemes Tihamér</a></p>
13  <p><a href="https://hu.wikipedia.org/wiki/Nemes_Nagy_%C3%81gnes" title="Ágnes, Nemes Nagy">Nemes Nagy Ágnes</a></p>
14  <p><a href="https://hu.wikipedia.org/wiki/Nemesg%C3%A1zok" title="Nemesgázok">Nemesgázok</a></p>
```

Érdekes Wiki oldalak

Nemes Tihamér

Nemes Nagy Ágnes

Nemesgázok

Kiválasztja az elemet, ha az adott attribútumnak a megadott szó az egyetlen értéke, esetleg ezzel kezdődik és kötőjellel folytatódik

attributum4.html

```
7   a[class|="nemes"] { background-color: yellow; }

12  <p><a href="https://hu.wikipedia.org/wiki/Nemes_Tiham%C3%A9r" class="nemes tihamer">Nemes Tihamér</a></p>
13  <p><a href="https://hu.wikipedia.org/wiki/Nemes_Nagy_%C3%81gnes" class="nemes-nagy-agnes">Nemes Nagy Ágnes</a></p>
14  <p><a href="https://hu.wikipedia.org/wiki/Nemesg%C3%A1zok" class="nemesgazok">Nemesgázok</a></p>
15  <p><a href="https://hu.wikipedia.org/wiki/Magyar_nemess%C3%A9g" class="fold nelkuli nemes">Magyar nemesség</a></p>
16  <p><a href="https://hu.wikipedia.org/wiki/Nemess%C3%A9g" class="nemes">Nemesség</a></p>
```

Érdekes Wiki oldalak

Nemes Tihamér
Nemes Nagy Ágnes
Nemesgázok
Magyar nemesség
Nemesség

Kiválasztja az elemet, ha attribútumának értéke az adott karaktersorozattal kezdődik

attributum5.html

```
7   a[class^="nemes"] { background-color: yellow; }

12  <p><a href="https://hu.wikipedia.org/wiki/Nemes_Tiham%C3%A9r" class="nemes tihamer">Nemes Tihamér</a></p>
13  <p><a href="https://hu.wikipedia.org/wiki/Nemes_Nagy_%C3%81gnes" class="nemes-nagy-agnes">Nemes Nagy Ágnes</a></p>
14  <p><a href="https://hu.wikipedia.org/wiki/Nemesg%C3%A1zok" class="nemesgazok">Nemesgázok</a></p>
15  <p><a href="https://hu.wikipedia.org/wiki/Magyar_nemess%C3%A9g" class="fold nelkuli nemes">Magyar nemesség</a></p>
16  <p><a href="https://hu.wikipedia.org/wiki/Nemess%C3%A9g" class="nemes">Nemesség</a></p>
```

Érdekes Wiki oldalak

Nemes Tihamér
Nemes Nagy Ágnes
Nemesgázok
Magyar nemesség
Nemesség

Kiválasztja az elemet, ha attribútumának értéke az adott karaktersorozattal fejeződik be

attributum6.html

```
7   a[class$="nemes"] { background-color: yellow; }

12  <p><a href="https://hu.wikipedia.org/wiki/Nemes_Tiham%C3%A9r" class="nemes tihamer">Nemes Tihamér</a></p>
13  <p><a href="https://hu.wikipedia.org/wiki/Nemes_Nagy_%C3%81gnes" class="nemes-nagy-agnes">Nemes Nagy Ágnes</a></p>
14  <p><a href="https://hu.wikipedia.org/wiki/Nemesg%C3%A1zok" class="nemesgazok">Nemesgázok</a></p>
15  <p><a href="https://hu.wikipedia.org/wiki/Magyar_nemess%C3%A9g" class="fold nelkuli nemes">Magyar nemesség</a></p>
16  <p><a href="https://hu.wikipedia.org/wiki/Nemess%C3%A9g" class="nemes">Nemesség</a></p>
```

Érdekes Wiki oldalak

Nemes Tihamér
Nemes Nagy Ágnes
Nemesgázok
Magyar nemesség
Nemesség

Kiválasztja az elemet, ha attribútumának értéke tartalmazza az adott karaktersorozatot

attributum7.html

```
7   a[class*="nemes"] { background-color: yellow; }

12  <p><a href="https://hu.wikipedia.org/wiki/Nemes_Tiham%C3%A9r" class="nemes tihamer">Nemes Tihamér</a></p>
13  <p><a href="https://hu.wikipedia.org/wiki/Nemes_Nagy_%C3%81gnes" class="nemes-nagy-agnes">Nemes Nagy Ágnes</a></p>
14  <p><a href="https://hu.wikipedia.org/wiki/Nemesg%C3%A1zok" class="nemesgazok">Nemesgázok</a></p>
15  <p><a href="https://hu.wikipedia.org/wiki/Magyar_nemess%C3%A9g" class="fold nelkuli nemes">Magyar nemesség</a></p>
16  <p><a href="https://hu.wikipedia.org/wiki/Nemess%C3%A9g" class="nemes">Nemesség</a></p>
```

Érdekes Wiki oldalak

Nemes Tihamér

Nemes Nagy Ágnes

Nemesgázok

Magyar nemesség

Nemesség

Stílusok forrása

Háromféle helyen lehet stílusokat megadni:

1. Külső fájlban (`css` kiterjesztés, `<link>` elem)
2. A `<head>` elembe ágyazott `<style>` elemben. Csak akkor ajánlott, ha egyetlen HTML fájlt kívánunk formázni ezekkel a stílusokkal.
3. Soron belül: a HTML elemek `style` attribútumának értékeként. Ismét keveredik a tartalom a stílussal, ezért általában nem ajánlott a használata!

egyszeruSzelektor2.html

```
3   <head>
4     <title>Egyszerű szelektorok</title>
5     <meta charset="utf-8" />
6     <style>
7       p { font-family: Garamond, "Times␣New␣Roman", Serif; }
8       a { color: navy; }
9       .forras {font-style: italic; }
10      .bevezeto { font-weight: bold; }
11    </style>
12  </head>

16        <h1 style="font-family:␣Arial,␣Sans-Serif;">Ma van a magyar
          ↪ informatika születésnapja</h1>
```

Ütközések feloldása

- Ha több előírás is vonatkozik ugyanannak az objektumnak a formázására, elsőként a forrás prioritása dönt (csökkenő sorrendben):
  1. soron belüli formázások
  2. külső és belső (`<link>`, `<style>` elemek) formázások
  3. böngésző alapértelmezése
- Ha forrás alapján nem lehet különbséget tenni, megvizsgálják a *specifikusságot*.
- Ha ez sem segít, a később betöltött szabály felülírja a korábbit.

utkozes1.html

```
6       <style>
7         p { color: red; }
8       </style>
9       <link rel="stylesheet" type="text/css" href="utkozes1.css" />
10    </head>
11    <body>
12      <p>Első bekezdés.</p>
13      <p style="color: green;">Második bekezdés</p>
```

utkozes1.css

```
1  p { color: blue; }
```

Első bekezdés.

Második bekezdés

utkozes2.html

```
6     <link rel="stylesheet" type="text/css" href="utkozes1.css" />
7     <style>
8       p { color: red; }
9       p { color: orange; }
10    </style>
11  </head>
12  <body>
13    <p>Első bekezdés.</p>
```

utkozes1.css

```
1  p { color: blue; }
```

Első bekezdés.

Specifikusság meghatározása

- A = 1, ha a stílus a `style` attribútumban található, egyébként 0 (pl. `<p style="color: red;">...</p>`)
- B = a szelektorban lévő ID attribútumok száma (pl. `#bekezdes`)
- C = a szelektorban lévő osztályok, attribútumok és látszólagos osztályok száma (pl. `.bekezdes`, `a[target="_blank"]`, `a:hover`)
- D = az elemnevek és látszólagos elemek száma (pl. `p`, `p::first-line`)

Az univerzális szelektor (`*`) és ami ebből származik (pl. `body *`) 0 specifikusságú.

Ütközések feloldása

Szelektor	B	C	D	Specifikusság
`*`	0	0	0	0
`p`	0	0	1	1
`div p`	0	0	2	2
`ul ol + li`	0	0	3	3
`h1 + *[title]`	0	1	1	11
`ul ol li.voros`	0	1	3	13
`li.voros.szint`	0	2	1	21
`#oszlopfej`	1	0	0	100

Az érték részei:

1. elhagyható előjel (+, −)
2. kulcsszó (pl. `red`, `thin`), szám, függvény (pl. `url()`, `rgb()`, stb.) vagy karakterlánc
3. mértékegység (`%`, `cm`, `pt`, `px`, `em`, stb.)

Használat:

- fenti három elem között nem lehet fehér karakter
- 0 érték esetén a mértékegység elhagyható
- rövidítések esetén (pl. `border` a `border-width`, `border-style` és `border-color` helyett) az értékeket fehér karakterek szeparálják

Abszolút mennyiségek

- `cm`, centiméter
- `mm`, milliméter
- `in`, inch (1in = 2,54cm)
- `px`, képpont (1px = 1 fizikai képpont a legfeljebb 96dpi-nél felbontású eszközökön, nagyfelbontású kijelzőkön több)
- `pt`, nyomdai pont (1pt = 1/72in)
- `pc`, pica (1pc = 12pt)

Relatív mennyiségek

- `em`, a karakterkészlet betűmérete
- `rem`, a `<html>` elem betűmérete
- `ex`, az x betű magassága
- `ch`, a 0 karakter szélessége
- `vw`, a viewport szélességének %-a
- `vh`, a viewport magasságának %-a
- `vmin`, a viewport kisebbik méretének 1%-a
- `vmax`, a viewport nagyobbik méretének 1%-a
- `%`, a szülő elem méretének %-a

Általános érték kulcsszavak

- `inherit`: az értéket meg kell örökölni a szülő elemtől.
- `initial`: a tulajdonság eredeti értéke.

Fekete bekezdésben piros szó.

Kék bekezdésben kék szó.

Zöld bekezdésben fekete szó.

inherit.html

```
7        span { color: red; }
8        .kek { color: blue; }
9        .zold { color: green; }
10       .kek span { color: inherit; }
11       .zold span { color: initial; }

15     <p>Fekete bekezdésben <span>piros</span> szó.</p>
16     <p class="kek">Kék bekezdésben <span>kék</span> szó.</p>
17     <p class="zold">Zöld bekezdésben <span>fekete</span> szó.</p>
```

Színek

Számtalan dolognak beállítható a színe CSS tulajdonságokkal, pl.:

`color`
: Szöveg írásszíne

`background-color`
: Háttérszín

Szín, mint a tulajdonság értéke megadható:

kulcsszavakkal
: Pl. `red` (vörös), `green` (zöld), `blue` (kék), `white` (fehér), `black` (fekete), ...
140 szabványos színkód

Hexadecimálisan, RGB összetevőkkel
: Pl. narancsszín: `#ff7f00`, ahol # jelzi a 16-os számrendszerbeli alakot, `ff` a vörös (Red), `7f` a zöld (Green) és `00` a kék (Blue) összetevő intenzitása 8 biten előjel nélkül, fixpontosan. Additív színkeverés.

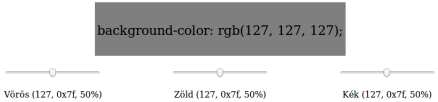
Színek

rgb() függvénnyel

`rgb(red, green, blue)`, ahol mindhárom összetevő lehet 0-255 közötti decimális egész, vagy 0-100%. Pl. `rgb(255,0,0)` vagy `rgb(100%, 0%, 0%)` vörös színt eredményez.

rgba() függvénnyel

`rgb(red, green, blue, alpha)`, ahol a színösszetevőket egy átlátszóság érték követi ([0, 1]).



szinek1.html

hsl() függvénnyel

`hsl(hue, saturation, lightness)`, ahol `hue` az árnyalat, [0, 360] fok közötti elfordulás a színkeréken. Pl. 0° a vöröshöz, 120° a zöldhöz, 240° a kékhez tartozik. `saturation` a telítettség, százalékban. A 0% a színinformáció hiányát (szürkeség) jelzi, 100% a teljes színezettséget. `lightness` a fényesség, szintén százalékban. A 0% mindig fekete, a 100% mindig fehér színt ad.

hsla() függvénnyel

A fentiek kiegészülnek átlátszósággal.

szinek2.html, papir.jpg

Az elemek átlátszósága az opacity tulajdonsággal is állítható [0, 1] intervallumban; a kisebb értékekhez nagyobb átlátszóság tartozik.

atlatszosag1.html (sze_logo.svg, it_logo.svg)

```
7        img        { opacity: 0.4; }
8        img:hover  { opacity: 1; }

12     <p><a href="https://uni.sze.hu"><img src="sze_logo.svg" alt="SZE logo"
          ↪ width="400" /></a>
13       <a href="https://it.sze.hu"><img src="it_logo.svg" alt="IT logo" width=
            ↪ "400" /></a></p>
```

Színek

Induljon ki a szinezes.html fájlból! Kapcsolja ezt össze egy külső stíluslappal, majd érje el, hogy a jobb oldali ábrának megfelelő színekben pompázzon! Próbáljon minél több féle szín megadási módszert alkalmazni! Törekedjen a lehető legtömörebb CSS szabályok megalkotására!

szinezes-mo.html, szinezes-mo.css

Az első szintű címsorok legyenek barnák!

Ez a bekezdés legyen vörös!

Ez pedig zöld, de az utolsó szava legyen viágoszöld!

Az első szintű címsorok legyenek barnák!

A második és harmadik szintű címsorok legyenek kékek!

Ez egy szürke stílusú bekezdés.

Ez itt egy harmadik szintű címsor.

Ez is szürke.

Ennek meg a háttere legyen citromsárga!

HTML elemek hátterével kapcsolatos tulajdonságok:

`background-color`
: A háttér színe. Alapértelmezetten `transparent`, azaz átlátszó.

`background-image`
: Háttérkép, amivel alapértelmezés szerint kicsempézi az elem teljes területét (margókat nem). Alapértéke `none`, nincs háttérkép. Az `url()` függvény paramétereként adható meg a képfájl, pl.
`background-image: url("hatter.png");`
Megadhatók színátmenetek is.
A szöveg maradjon olvasható a háttéren!

background-repeat

Háttérkép csempézési iránya

- repeat mindkét irányban, túlnyúló részek levágásával, alapértelmezés
- repeat-x csak vízszintesen
- repeat-y csak függőlegesen
- no-repeat csak egyszer, alapértelmezetten a bal felső sarokban
- round torzítja a képet a vágás elkerülésére
- space csak annyiszor ismétel, ami vágás nélkül elfér, közöttük helyet hagy

Két érték megadásakor az első a vízszintes, második a függőleges irányra vonatkozik.

Háttér

background-repeat:

- (●) Mindkét irányban csempéz (repeat)
- ( ) Vízszintesen csempéz (repeat-x)
- ( ) Függőlegesen csempéz (repeat-y)
- ( ) Csak egyszer jelenik meg (no-repeat)
- ( ) Elosztja térközökkel (space)
- ( ) Torzít (round)

hatter.html, css3.svg

`background-position`

Igazítás, a *vízszintes* és a *függőleges* pozíciót várja. Ha egyet kap, a másik `center` lesz.

- Függőlegesen: `left`, `center`, `right`
- Vízszintesen: `top`, `center`, `bottom`
- Mindkettőnél lehet százelékot, vagy egyéb CSS mértékegységet (pl. képpont) használni.

pozicio1.html

```
7         p {
8           background-image: url("css3.svg");
9           background-size: 10%;
10          background-repeat: no-repeat;
11        }

15      <h1>Balra fent</h1>
16      <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.

24      <h1>Jobbra lent</h1>
25      <p style="background-position: right bottom">Aenean ut suscipit

37      <h1>Középen</h1>
38      <p style="background-position: center">Ut semper, ante pretium

53      <h1>Jobb alsó saroktól 10%-ra</h1>
54      <p style="background-position: 90% 90%">Lorem ipsum dolor sit
```

background-attachment

- `scroll` a háttér együtt gördül az oldallal, alapértelmezés
- `fixed` rögzített háttér
- `local` az elem tartalmával együtt gördül a háttér

A logo mindig a jobb alsó sarokban: rogzites1.html
Két bekezdés között kilátszik a háttérben rögzített logo: rogzites2.html

background

Rövidítés: egy összetett tulajdonsággal sok egyszerű tulajdonság értéke állítható be.

Értékek sorrendje rögzített, de tetszőleges számú érték elhagyható!

background: *background-color background-image background-repeat background-attachment background-position*

pozicio1.html

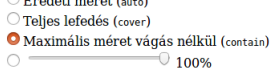
```
p {
  background-image: url("css3.svg");
  background-size: 10%;
  background-repeat: no-repeat;
}
```

pozicio2.html

```
p {
  background: url("css3.svg") no-repeat;
  background-size: 10%;
}
```

background-size

- `auto`: Alapértelmezés, eredeti méret.
- *szélesség, magasság*: utóbbi elhagyásával `auto`-t feltételez. Használhatók CSS mértékegységek és százalékok (a szülő elem mérete a 100%, nem a sajátja!).
- `cover` Addig nyújt és vág, amíg le nem fedi a szülő elem teljes területét.
- `contain` Addig nyújt, amíg egyszer bele nem fér a háttér a szülő elembe.



meret.html, css3.svg

Háttér

Induljon ki a rogzites2.html fájlból, és alakítsa át a jobb oldali ábrának megfelelően!

- Az írásszín legyen világos szürke!
- A teljes oldal háttere legyen kék (RGB-összetevők: 0, 145 és 190)!
- A `<div>` elem hátteréként állítsa be a HTML5sticker.png fájl!
- Ennek helyzete ne függjön a görgetéstől!
- Helyezze el azt a képernyő közepén!
- A képet méretezze aránytartó módon úgy, hogy éppen kitöltse a rendelkezésre álló helyet!
- Próbálja mindezt a lehető legkevesebb CSS tulajdonság felhasználásával elérni!

rogzites3-mo.html, HTML5sticker.png

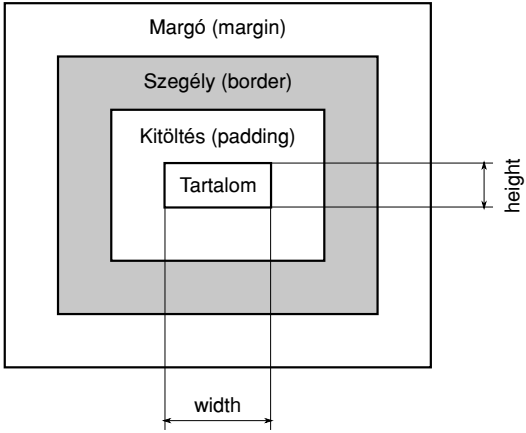
Dobozmodell

Minden HTML elemet egy *doboznak* tekintünk. Ezek szerkezete belülről kifelé:

- Az elem tartalma (szöveg, kép, ...)
- Kitöltés (`padding`; átlátszó)
- Szegély (`border`)
- Margó (`margin`; átlátszó)

Megjegyzések

- A szélesség (`width`) és magasság (`height`) tulajdonságok a tartalmi rész méreteire vonatkoznak.
- Soron belüli elemek méretét a böngésző határozza meg, nem méretezhetőek át.



dobozMeret.html

```
7   body {
8     margin: 0px;
9   }
10  p {
11    background-color: yellow;
12    padding: 20px;
13    border: 5px solid red;
14    margin: 10px;
15  }
16  span {
17    background-color: orange;
18  }
```

Ez a bekezdés a szülő (body) szélességéhez igazodik, magassága a tartalom függvénye.

Ez egy 200x100 képpont méretű bekezdés.

A bekezdésben lévő soron belüli elemek mérete nem módosítható!

Kivéve, ha a megjelenítés módját blokkszintűre módosítjuk.

```
22  <p>Ez a bekezdés a szülő (<code>body</code>) szélességéhez igazodik, magassága a tartalom
      ↪ függvénye.</p>
23  <p style="width: 200px; height: 100px;">Ez egy 200x100 képpont méretű bekezdés.</p>
24  <p>A bekezdésben lévő <span style="width: 120px; height: 50px;">soron belüli elemek</span>
      ↪ mérete nem módosítható!</p>
25  <p>Kivéve, ha a <span style="display: block; width: 120px; height: 50px;">megjelenítés mó
      ↪ dját</span> blokkszintűre módosítjuk.</p>
```

Mit számol bele a böngésző a méret (`width`, `height`) adatokba? → `box-sizing`

`content-box`
: Csak a tartalom méretét

`border-box`
: Tartalom + kitöltés + szegély

Kényelmes:

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

Név: Pl. Gipsz Jakab

Végzettség: középiskola/gimnázium

meretezes.html

```
11      input[type=text] {
12        width: 200px;
13        box-sizing: border-box;
14      }
15      select {
16        width: 200px;
17      }
```

Blokk szintű elemek szélessége (`width`) és magassága (`height`) megadható:

- `auto`: alapértelmezett
- valós világbeli, relatív vagy megjelenítőtől függő mértékegység (pl. `cm`, `ex`, `px`)
- a tartalmazó blokk %-ában megadva
- Érdekes lehetőség az abszolút és relatív méretek vegyítésére: `calc()` függvény. Négy alapműveletes kifejezések kiértékelhetők vele.

calc.html

```
7        body { background-color: red; }
8        p {
9          width: calc(100% - 100px);
10         background-color: yellow;
11       }

14   <body>
15     <p>calc függvény használata</p>
16   </body>
```

A merev méretek helyett megadhatók intervallumok is:

`max-width`
: Ennél csak keskenyebb lehet. Felülírja `width` értékét.

`min-width`
: Ennél csak szélesebb lehet. Ha a tartalom ennél szélesebb, nem veszik figyelembe. A szélesség változásával a magasság is változhat.

`max-height`
: Ennél csak alacsonyabb lehet. Ha a tartalom ennél magasabb, a viselkedés az `overflow`-tól függ. Felülírja `height` értékét.

`min-height`
: Ennél csak magasabb lehet. Ha a tartalom ennél alacsonyabb, akkor ekkorára növeli meg a magasságot.

Túlcsordulások kezelése: `overflow`

`visible`
: A túllógó részek is megjelennek, esetleg rálógva más tartalmakra. Alapértelmezés.

`hidden`
: A túllógó részeket levágják.

`scroll`
: Görgetősávok jelennek meg a túllógó részek megjelenítéséhez. Némelyik böngésző mindig mutatja, mások csak akkor, ha szükséges.

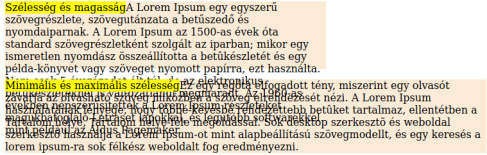
`auto`
: Csak akkor jelennek meg görgetősávok, ha nem fér el a tartalom.

Léteznek `overflow-x` és `overflow-y` tulajdonságok csak az egyik irány viselkedésének megadásához.

tulnyulas.html – Ellenőrizze a méretezés hatását, túlcsordulásokat!

```
11      <p style="width: 500px; height: 105px;"><mark>Szélesség és
12      magasság</mark>A Lorem Ipsum egy egyszerű szövegrészlete,

20      legutóbb softwarekkel mint például az Aldus Pagemaker.</p>
21      <p style="min-width: 400px; max-width: 750px;"><mark>Minimális és
22      maximális szélesség</mark>Ez egy régóta
```



Az első bekezdés tartalma rálóg a másodikra, túl alacsony a blokk. A második bekezdés maximális méreten. Ha keskenyre állítjuk az ablakot, vízszintes görgetősáv jelenik meg a böngészőablak alján.

Szegélyek

A szegélyeknek állítható a

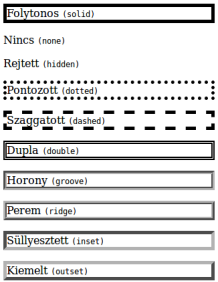
- stílusa (`border-style`),
- szélessége (`border-width`), és a
- színe (`border-color`).

Megjegyzések:

- Utóbbi kettő csak a stílus beállítása esetén működik.
- Minden paraméter állítható külön az egyes oldalakra is.

szegelyek1.html

```
14      <p style="border-style: solid">Folytonos <code>(solid)</code></p>
15      <p style="border-style: none">Nincs <code>(none)</code></p>
16      <p style="border-style: hidden">Rejtett <code>(hidden)</code></p>
17      <p style="border-style: dotted">Pontozott <code>(dotted)</code></p>
```



Oldalankénti szegélystílusok megadhatók:

- 1-4 érték megadásával, pl. `border-style: dotted dashed solid none;`
- Oldalakra vonatkozó tulajdonságokkal: `border-*-style`, ahol `*` helyén állhat `top`, `right`, `bottom`, `left`.

Ha a `boder-style`-nak

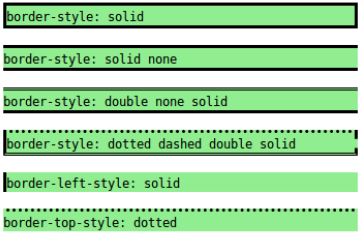
1 értéke van
: `felül-jobb-alul-bal` (minden oldalra ugyanazt a stílust állítja)

2 értéke van
: `felül-alul jobb-bal`

3 értéke van
: `felül jobb-bal alul`

4 értéke van
: `felül jobb alul bal` (óramutató járása szerint)

Hasonlóképpen lehet oldalanként szabályozni a margókat és kitöltéseket is.



szegelyek2.html

Ha táblázatok szomszédos cellái közös, de eltérő stílusú szegélyeket használnak, akkor

`none`
: ha a szomszédnak be van állítva a szegélye, az fog megjelenni

`hidden`
: még ha be is van állítva a szomszéd szegélye, akkor sem fog megjelenni

A1	B1
A2	B2

szegelyek3.html

```
21  <td style="border-style: hidden
    ↪ ">A1</td>

26  <td style="border-style: none">
    ↪ B2</td>
```

Rövidítések

`border: width style color`

Minden oldalon beállítja a szegély szélességét, stílusát, színét.

`border-*: width style color`

A * lehet `top`, `right`, `bottom` és `left`; csak ezekre állítja a fenti három tulajdonságot.

Szegélyek lekerekítése

`border-top-left-radius`
: Bal felső

`border-top-right-radius`
: Jobb felső

`border-bottom-right-radius`
: Jobb alsó

`border-bottom-left-radius`
: Bal alsó

`border-radius`
: Rövidítés fentiek gyors beállításához

`border-radius` paraméterezése

1 érték megadásával

mind a négy sarkot állítja

2 érték megadásával

bal felső + jobb alsó, jobb felső + bal alsó

3 érték megadásával

bal felső, jobb felső + bal alsó, jobb alsó

4 érték megadásával

bal felső, jobb felső, jobb alsó, bal alsó

lekerekites.html

```
7        p {
8          border-style: solid;
9          border-width: 3px;
10         padding: 10px;
11       }
12       #mind {
13         border-color: red;
14         border-radius: 10px;
15       }
16       #szemkozti {
17         border-color: green;
18         border-radius: 10px 20px;
19       }
20       #jobbalso {
21         border-color: blue;
22         background-color: navy;
23         color: white;
24         border-radius: 5px;
25         border-bottom-right-radius: 30px;
26       }
```

Minden sarok kerekítve

Szemközti sarkok eltérő lekerekítési sugárral

Jobb alsó sarok erősen lekerekítve

Margók

A margók mindig átlátszók, csak a szélességük állítható:

- 1-4 érték megadásával, pl.
  `margin: 10px 20px 30px 40px;`
  (Fent, jobbra, lent, balra; további esetek mint `border-style`-nál.)
- Oldalakra vonatkozó tulajdonságokkal:
  `margin-*`, ahol `*` helyén állhat `top`, `right`, `bottom`, `left`.

Margók

A margó szélessége lehet:

- `auto`: a tartalom által fel nem használt helyet felosztja egyenlően a bal és jobb oldal közt → középre igazít
- `inherit`: a befoglaló, szülő elem beállításait örökli
- CSS mértékegységgel (pl. `px`, `cm`) adott
- %: a szülő elem méretének százaléka

Negatív értékek is használhatók.

Margók

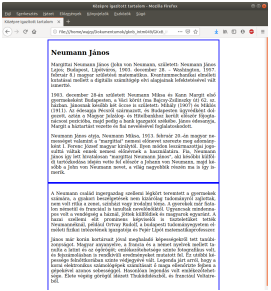
Az ablak keskenyebb, mint az elem *rögzített* szélessége? → gördítősáv. *Maximális* szélesség → csökkenthető. Középre igazítás: `margin: auto`-val.

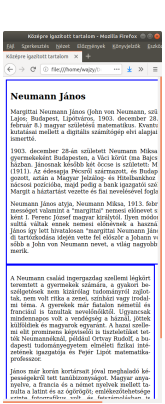
kozepre.html

```
7       section {

12        margin: auto;
13      }
14      section.fixszel { width: 600px; }
15      section.maxszel { max-width: 600px; }

19    <section class="fixszel">
20      <h1>Neumann János</h1>
21      <p>Margittai Neumann János (John von
          ↪ Neumann, született: Neumann
```





Margók

A blokkok felső és alsó margói időnként összeolvadnak, és a kettő közül csak a nagyobb marad meg:

- szülő szomszédos gyerekei között (szélső gyerekek margói túlnyúlnak a szülőn)
- ha nincs olyan megjeleníthető szegély, kitöltés, stb., ami elválasztaná a szülő és valamely gyerekének alsó/felső margóját
- üres blokkok alsó és felső margóját is összevonják

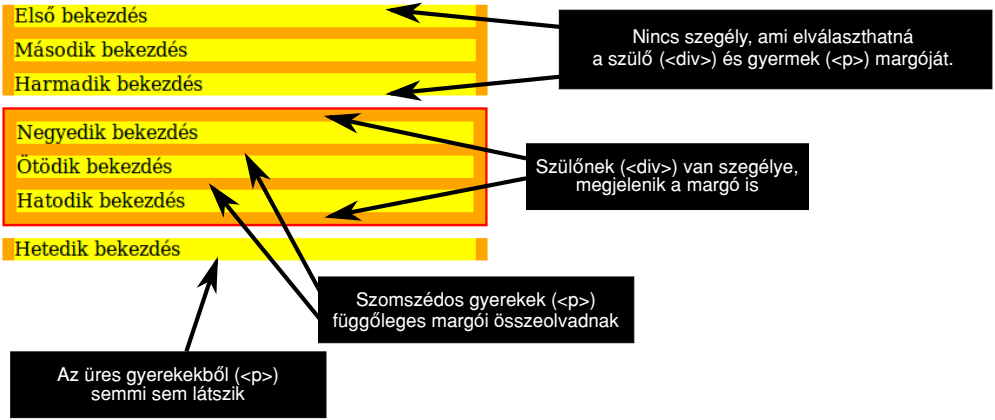
További részletek

margok.html

```
7   div {
8     background-color: orange;
9   }
10  div.keretes {
11    border: 2px solid red;
12  }
13  p {
14    background-color: yellow;
15    margin: 10px;
16  }
```

```
20  <div>
21    <p>Első bekezdés</p>
22    <p>Második bekezdés</p>
23    <p>Harmadik bekezdés</p>
24  </div>
25  <div class="keretes">
26    <p>Negyedik bekezdés</p>
27    <p>Ötödik bekezdés</p>
28    <p>Hatodik bekezdés</p>
29  </div>
30  <div>
31    <p>Hetedik bekezdés</p>
32    <p></p>
33    <p></p>
34  </div>
```

Margók


Első bekezdés
Második bekezdés
Harmadik bekezdés
Negyedik bekezdés
Ötödik bekezdés
Hatodik bekezdés
Hetedik bekezdés
Nincs szegély, ami elválaszthatná a szülő ( ) és gyermek (<p>) margóját.
Szülőnek ( ) van szegélye, megjelenik a margó is
Szomszédos gyerekek (<p>) függőleges margói összeolvadnak
Az üres gyerekekből (<p>) semmi sem látszik

Kitöltés

A kitöltések mindig átlátszók, csak a szélességük állítható:

- 1-4 érték megadásával, pl.
  `padding: 10px 20px 30px 40px;`
  (Fent, jobbra, lent, balra; további esetek mint `border-style`-nál.)
- Oldalakra vonatkozó tulajdonságokkal:
  `padding-*`, ahol `*` helyén állhat `top`, `right`, `bottom`, `left`.

Kitöltés

A kitöltés szélessége lehet:

- `inherit`: a befoglaló, szülő elem beállításait örökli
- CSS mértékegységgel (pl. `px`, `cm`) adott
- %: a szülő elem méretének százaléka

Negatív értékek nem használhatók.

Kitöltés

Próbálja meg elkészíteni az ábrának megfelelően a dobozokat!

Ezt a részt azzal emeljük ki a szövegből, hogy 15 képpont széles kitöltése van a bal oldalon, mellette egy ugyanolyan széles, narancssárga, egyszínű szegély található.

A héttérszín szürkéjének színösszetevői 240 értékűek.

Fekete alapon fehérrel írva, körben 5 képpontnyi kitöltéssel valami nagyon súlyos aggodalomra okot adó tényt közlünk.

A fejléc írásszín komponensei: 237, 234, 142. A háttérszín összetevők: 173, 116, 36, kitöltés körben 5 képpont, felső margó 10 képpont, a többi oldalon nincs. A szegély körben 1 képpont széles, folytonos, 107, 89, 65 színösszetevőjű vonal.

Az elemek háttérszín-összetevői: 242, 201, 143. A kitöltés körben 5 képpont, margók nincsenek.

A szegélyek itt is ugyanolyan szélességűek és színűek, mint a fejlécben, de csak a két oldalon és alul vannak megrajzolva.

dobozok.html

Körvonal

Körvonal (outline): az elemet a szegélyen kívülről körülöleli, kiemeli környezetéből. Rálóghat más elemekre.

`outline-style`
: Stílus, mint `border-style`, pl. `solid`, `dotted`, `double`, ...
: A többi tulajdonság beállítása hatástalan a stílus megadása nélkül!

`outline-color` Körvonal színe. Értéke lehet `invert`, ami minden háttéren látható.

`outline-width` Szélesség CSS mértékegységekben, vagy `thin`, `medium`, `thick`.

Rövidítés:

`outline: outline-width outline-style outline-color`

Sorrend tetszőleges, bármelyik érték elhagyható.

`outline-offset`

A körvonal távolsága a szegélytől. Ez a terület áttetsző.

Körvonal

korvonal.html

```
7        p {
8          background-color: antiquewhite;
9          border: 1px solid black;
10         width: 90%;
11         margin: auto;
12       }

20     leírására is, mint például az SVG, XUL stb.</p>
21     <p style="outline: 15px solid red; outline-offset: 5px;">A CSS
22     specifikációját a World Wide Web Consortium felügyeli.</p>
23     <p>A CSS-t a weblapok szerkesztői és olvasói egyaránt
```

A CSS (Cascading Style Sheets, magyarul: lépcsőzetes stíluslapok) a számítástechnikában egy stíluslelíró nyelv, mely a HTML vagy XHTML típusú strukturált dokumentumok megjelenését írja le. Ezenkívül használható bármilyen XML alapú dokumentum stílusának leírására is,

A CSS specifikációját a World Wide Web Consortium felügyeli.

megjelenéshez kapcsolódó elemeit. A tervezése során a legfontosabb szempont az volt, hogy elkülönítsék a dokumentumok struktúráját (melyet HTML vagy egy hasonló leíró nyelvben lehet megadni) a dokumentum megjelenésétől (melyet CSS-sel lehet megadni). Az ilyen elkülönítésnek több haszna is van, egyrészt növeli a weblapok használhatóságát, rugalmasságát és a megjelenés kezelhetőségét, másrészt csökkenti a dokumentum tartalmi struktúrájának komplexitását. A CSS ugyancsak alkalmas arra, hogy a dokumentum stílusát a megjelenítési módszer függvényében adja meg, így elkülöníthető a dokumentum formája a képernyőn, nyomtatási lapon, hangos böngészőben (mely beszédszintetizátor segítségével olvassa fel a weblapok szövegét), vagy Braille-készüléken megjelenítve.

Általános fontcsaládok: nagyon hasonló megjelenésű karakterkészletek

Serif

„Talpas” betűkészletek; főleg bekezdések szövegéhez, mert „vezeti a szemet” az alapvonalon, de képernyőn sokan nehezen olvassák

Sans-serif

„Talp nélküli” betűkészletek, főleg címsorokhoz

Monospace

„Egyenközű”, azonos szélességű betűkből álló betűkészletek, főleg forrásszövegekhez

`font-family`: karakterkészlet kiválasztása

- Karakterkészletek listája; ha valamelyik nincs telepítve, a következővel próbálkozik → érdemes egy általános fontcsalád nevét tenni a végére
- Ha a névben szóköz van, idézőjelek közé kell tenni
- Jól bejáratott kombinációk, pl.
  - `"Times New Roman", Times, serif`
  - `Arial, Helvetica, sans-serif`
  - `"Courier New", Courier, monospace`

`font-style`: álló és dőlt betűk

`normal`
: Álló betűk, alapértelmezés

`italic`
: Dőlt betűk

`oblique`
: „Kevésbé dőlt”, gyenge támogatás

`font-variant`: változatok

`normal`
: Normál betűk, alapértelmezett.

`small-caps`
: Kiskapitális, a kisbetűket kicsinyített nagybetűkkel helyettesíti.

`font-size`: méretezés

Abszolút méretekben

A felhasználó nem méretezheti át. Pl. `px` (CSS képpont), `pt` (nyomdai pont).

Relatív méretekben

Felhasználó átméretezheti. Pl. `em` (1em a bekezdések alapértelmezett mérete = `16px`), `%` (a szülő elem betűkészletének méretéhez viszonyítva), `vw` (`1vw` = a *viewport* szélességének 1%-a; átméreteződik az ablak méretezésével)

Kulcsszavakkal

Előre definiált méretek: `xx-small`, `x-small`, `small`, `medium`, `large`, `x-large`, `xx-large`.
Átméretezés: `smaller`, `larger`.

`font-weight`: „vastagság”, „súly”

`normal`
: Normál szélesség (400), alapértelmezett.

`bold`
: Félkövér (700)

`bolder`, `lighter`
: Növeli, csökkenti a vastagságot

`100, 200, 300, ..., 900`
: Különféle vastagságok, de többnyire csak a normál és a félkövér támogatott.

Rövidítés:

```
font: font-style font-variant font-weight font-size/line-height
    font-family | caption | icon | menu | message-box |
    small-caption | status-bar | initial | inherit;
```

A méret és a karakterkészlet megadása kötelező. A `caption`, `icon`, ... kulcsszavakkal lehet a böngésző által valamilyen célra már használt beállításokat kérni egy adott helyen.

karakter.html

```
6   <style>
7     h1 { font-family: Arial, Helvetica, sans-serif; }
8     p { font-family: "Times New Roman", Times, serif; }
9     p.prg { font-family: "Courier New", Courier, monospace; }
10  </style>

13  <h1>Ez egy címsor</h1>
14  <p>Ez egy bekezdés.</p>
15  <p class="prg">printf("Hello World!\n");</p>
16  <p>Ebben a bekezdésben <span style="font-style: italic;">döntött
17  </span> és <span style="font-style: oblique;">kissé döntött</span>
18  részek is előfordulnak. <span style="font-variant: small-caps;">
19  Néha kiskapitális formázást használunk.</span>
20  <span style="font-size: 20px;">20px</span>,
21  <span style="font-size: 20pt;">20pt</span>,
22  <span style="font-size: 1.5em;">1.5em</span>,
23  <span style="font-size: 5vw;">5vw</span>,
24  <span style="font-size: smaller;">smaller</span>,
25  <span style="font-size: larger;">larger</span>.</p>
```

Ez egy címsor

Ez egy bekezdés.

```
printf("Hello World!\n");
```

Ebben a bekezdésben *döntött* és *kissé döntött* részek is előfordulnak. NÉHA KISKAPITÁLIS FORMÁZÁST HASZNÁLUNK.

20px, 20pt, 1.5em, 5vw, smaller, larger.

Karakterkészletek letölthetők a hálózatról: `@font-face`

- egyedi megjelenést kölcsönöz
- mindenki ugyanazt a készletet használja, garantáltan azonos megjelenés mindenhol (sok eszközön hiányosak a készletek, főleg a ritkán használt karakterek)

Megbízhatóan használható formátumok:

- TrueType Font (TTF)
- OpenType Font (OTF)
- Web Open Font Format (WOFF)

A `@font-face` szabályban használható tulajdonságok:

`font-family`
: Ezzel a névvel lehet majd hivatkozni a karakterkészletre később, kötelező.

`src`
: A fájl forrását adja meg `url()` CSS függvénnyel, kötelező.

`font-stretch`
: Ha a karakterkészletnek készültek különféle sűrűségű változatai, ezzel lehet kiválasztani, hogy valamelyiket milyennek tekintsen a böngésző (pl. `normal`, `condensed`, `expanded`). Ennek hiányában a böngészőnek kell előállíttatnia a speciális formákat a normálból kiindulva.

`font-style`
: Hasonlóan, dőlt változatokhoz (pl. `normal`, `italic`).

`font-weight`
: Hasonlóan, a „kövérséghez” (pl. `normal`, `bold`).

webfont.html

```
7        @font-face {
8          font-family: devil;
9          src: url("devil/Devil-East-Free-Font.woff");
10       }
11       p {
12         font-family: devil;
13         font-size: 80pt;
14       }
15       p.felkover {
16         font-weight: bold;
17       }

21     <p>Devil East Free Font</p>
22     <p class="felkover">Devil East Free Font</p>
```

Devil East Free Font

Devil East Free Font

Google Fonts

- Több száz ingyenes karakterkészlet
- Könnyű kereshetőség
- Egyszerű integráció a weboldalba

googleFonts.html

```
6    <link href="https://fonts.googleapis.com/css?
         family=Baloo+2&display=swap" rel="
         stylesheet" />
7    <style>
8      p {
9        font-family: 'Baloo 2', cursive;
10       font-size: 20pt;
11     }
12   </style>

15   <p>Google Fonts - Baloo</p>
```

Google Fonts - Baloo

Készítse el Semmelweis Ignác oldalát a Wiki oldal szövegét felhasználva!

- Töltse le a Ballerina karakterkészletet!
- Használja ezt az első szintű címsorban szereplő név kiírására, 42 nyomdai pont méretben!
- A bekezdések szövegét írja Libre Baskerville karakterkészlettel, 12 nyodai pont mérettel!
- Készítsen stílusokat a félkövér és dőlt betűs részek megjelöléséhez!

semmelweis.html

Semmelweis Ignác

Semmelweis Ignác Fülöp (*Buda, 1818. július 1. – Döbling, 1865. augusztus 13.*) magyar orvos, „az anyák megmentője".

1818. július 1-jén a tabáni Palota utca (ma Apród utca) 1–3. alatti Meindl-házban látta meg a napvilágot, ahol édesapjának – a hienc (nyugat-magyarországi német) ősökkel rendelkező Semmelweis Józsefnek (1778–1846) – jól menő fűszerüzlete is volt. Édesanyja Müller Teréz, Müller Fülöp módos krisztinavárosi sváb kocsigyártó és Anderl Teréz leánya. A szülők 1810. január 14-én kötöttek házasságot a Krisztinavárosban. Ignác (a tízből) ötödik gyerekként született, két bátyja ismert pesti kereskedő, Károly öccse pedig a Belvárosi római katolikus plébánia lelkésze volt. Ignácot a közeli Tabáni Plébániatemplomban[1] keresztelték meg.[2] (A kor következetlen anyakönyvezésére jellemző, hogy a tíz testvér vezetéknevét tízféleképpen írták; Semmelweiss, Semmelweis, Semelveisz stb.)

Web-technológia

Cascading Style Sheets, II. rész

Dr. Hatwagner F. Miklós

Széchenyi István Egyetem, Győr

https://github.com/wajzy/GKxB_INTM049.git

2021. október 16.

SZÉCHENYI EGYETEM
UNIVERSITY OF GYŐR

INFORMATIKA TANSZÉK
DEPARTMENT OF COMPUTER SCIENCE

text-align

Vízszintes igazítás: `left` (balra), `center` (középre), `right` (jobbra), `justify` (sorkizárt)

vizszintes.html

```
7        .balra { text-align: left }
8        .kozepre { text-align: center }
9        .jobbra { text-align: right }
10       .sorkizart { text-align: justify }

15     <p class="balra">
16       A HTML (angolul: HyperText Markup Language=hipersz
         ↪ öveges jelölőnyelv) egy leíró nyelv,
         ↪ melyet weboldalak készítéséhez
         ↪ fejlesztettek ki, és mára már internetes
         ↪ szabványnyá vált a W3C (World Wide Web
         ↪ Consortium) támogatásával.
```

`hyphens`: elválasztások, hogy a szöveg tördelése még finomabb legyen

`none`
: nincs elválasztás, alapértelmezés

`manual`
: elválasztás csak a kézzel előre megjelölt helyeken (`‐`, ``)

`auto`
: automatikus elválasztás

elvalasztas.html

```
8   <p>Nincs elválasztás. Nincs elválasztás. Nincs elválasztás.

12  <p style="hyphens: manual;">El&hyphen;vá&hyphen; lasz&hyphen;tás
13  kézzel megjelölt he&shy;lye&shy;ken. El&hyphen;vá &hyphen;lasz

16  <p style="hyphens: auto;">Automatikus elválasztás. Automatikus
```

Nincs elválasztás. Nincs elválasztás. Nincs elválasztás. Nincs elválasztás. Nincs elválasztás. Nincs elválasztás. Nincs elválasztás. Nincs elválasztás. Nincs elválasztás. Nincs elválasztás.

El-vá-lasz-tás kézzel megjelölt helyeken. El-vá-lasz-tás kézzel megjelölt helyeken. El-vá-lasz-tás kézzel megjelölt helyeken.

Automatikus elválasztás. Automatikus elválasztás. Automatikus elválasztás. Automatikus elválasztás. Automatikus elválasztás. Automatikus elválasztás. Automatikus elválasztás. Automatikus elválasztás.

`vertical-align`: tetszőleges elem függőleges igazítása

`baseline`
: szülő szövegének alapvonalához

távolság
: tetszőleges mértékű süllyesztéshez/emeléshez, negatív érték is elfogadott

`%`
: sormagasság %-ában megadott emelés/süllyesztés, negatív érték is elfogadott

`sub`
: szülő alsó indexéhez

`super`
: szülő felső indexéhez

`top`
: sor legmagasabb eleméhez

`text-top`
: szülő elem szövegének tetejéhez

`middle`
: szülő közepéhez

`bottom`
: sor legalsó eleméhez

`text-bottom`
: szülő szövegének aljához

fuggoleges.html

```
7       p, span, code { border: 1px solid black; }
8       span { border-color: red; }
9       code { border-color: orange; }

13      <p><code>baseline</code><span style="vertical-
          ↪ align:␣baseline;">
14        <img src="sze_logo.svg" width="100" alt="SZE
            ↪ ␣logo">
15        alapvonal</span></p>
```

baseline alapvonal

távolság -10px

% 50%

sub alsó index

super felső index

top sor legfelső eleméhez

text-top szülő szövegének tetejéhez

middle szülő közepéhez

bottom sor legalsó eleméhez

text-bottom szülő szövegének aljához

text-indent

Első sor behúzása: *távolság* (a bekezdés bal szélétől számított behúzás), % (szülő elem szélességének százalékában adott behúzás)

behuzas.html

```
7         p {
8           text-indent: 1cm;
9           text-align: justify;
10        }

14      <p>
15        A HTML (angolul: HyperText Markup Language=hipersz
            ↪ öveges jelölőnyelv) egy leíró nyelv,
            ↪ melyet weboldalak készítéséhez
            ↪ fejlesztettek ki, és mára már internetes
            ↪ szabvánnyá vált a W3C (World Wide Web
            ↪ Consortium) támogatásával.
16      </p>
```

A HTML (angolul: HyperText Markup Language=hiperszöveges jelölőnyelv) egy leíró nyelv, melyet weboldalak készítéséhez fejlesztettek ki, és mára már internetes szabvánnyá vált a W3C (World Wide Web Consortium) támogatásával.

Az aktuális változata a 4.01, mely az SGML általános jelölőnyelv egy konkrét alkalmazása (azaz minden 4.01-es HTML dokumentum egyben az SGML dokumentumszabványnak is meg kell, hogy feleljen).

`white-space`: fehér karakterek értelmezése

`normal`
: szomszédos fehér karaktereket összevonja, alapértelmezés

`nowrap`
: nem tördeli a hosszú sorokat, de a szomszédos fehér karaktereket összevonja

`pre`
: utánozza a `<pre>` HTML elem működését, minden fehér karaktert megőriz

`pre-line`
: szomszédos fehér karaktereket összevonja, de tördeli a sorokat, ha szükséges

`pre-wrap`
: minden fehér karaktert megőriz, és tördel, ha szükséges

Nem választ magától monospace karakterkészletet!

fahrcels2.html

```
7        p {
8           white-space: pre;
9           font-family: monospace;
10       }

14    <p>
15 Fahrenheit - Celsius
16 ---------------------
```

```
Fahrenheit - Celsius
--------------------
       300     148.9
       280     137.8
       260     126.7
       240     115.6
       220     104.4
       200      93.3
       180      82.2
       160      71.1
       140      60.0
       120      48.9
       100      37.8
        80      26.7
        60      15.6
        40       4.4
        20      -6.7
         0     -17.8
```

`letter-spacing`: betűk közötti távolság

`normal`
: szokásos távolság, alapértelmezés

távolság
: betűk közötti távolság, negatív érték is elfogadott

`word-spacing`: szavak közötti távolság

`normal`
: szokásos távolság (betűmagasság negyede), alapértelmezés

távolság
: szavak közötti távolság, negatív érték is elfogadott

tavolsag.html

```
7        .betusuru { letter-spacing: -.1em }
8        .betunormal { letter-spacing: normal }
9        .beturitka { letter-spacing: .5em }
10       .szoritka { word-spacing: .5cm }
11       .szosuru { word-spacing: -.2em }

15     <p>Néhány szóban a <span class="beturitka">betűk távolságát
16     megnöveltük</span>, <span class="betusuru">másutt
```

Néhány szóban a b e t ű k t á v o l s á g á t m e g n ö v e l t ü k , másutt lecsökkentettük, de néhány helyen megőriztük az eredeti állapotot. Hasonlóan a szavak köztiszóközöketislehetsűríteni, de ritkítani is.

`text-transform`: szöveg átalakítása

`normal`
: nincs átalakítás, alapértelmezés

`capitalize`
: minden kezdőbetűt nagybetűvel nyomtat

`uppercase`
: csupa nagybetűvel nyomtat

`lowercase`
: csupa kisbetűvel nyomtat

nagybetu.html

```
7        .nagykezdo { text-transform: capitalize }
8        .nagybetu { text-transform: uppercase }
9        .kisbetu { text-transform: lowercase }

13     <p>Ebben a bekezdésben nincsen semmi különös.</p>
14     <p class="nagykezdo">Itt viszont minden szó első betűje nagybetűs.</p>
```

Ebben a bekezdésben nincsen semmi különös.

Itt Viszont Minden Szó Első Betűje Nagybetűs.

LEHET KIABÁLNI IS.

vagy úgy járni, mint szegény nemecsek.

`text-decoration-line`: vonal húzása a szöveggel párhuzamosan

`none`
: nincs vonalazás, alapértelmezés. Pl. hivatkozások aláhúzásának eltávolításához használható.

`underline`
: aláhúzza a szöveget; félrevezetheti az olvasót, ha nem csak a hivatkozások jelennek meg aláhúzással!

`overline`
: a szöveg fölött húz vonalat

`line-through`
: áthúzza a szöveget

`text-decoration-style`: a vonal stílusa

`solid`
: folytonos vonal

`double`
: dupla vonal

`dotted`
: pontvonal

`dashed`
: szaggatott vonal

`wavy`
: hullámos vonal

`text-decoration-color`: a vonal színe

szín

tetszőleges CSS színmegadási móddal

Rövidítés:

`text-decoration:`

`text-decoration-line text-decoration-color text-decoration-style`

Akár többféle vonal is megadható, tetszőleges rész elhagyható, sorrend tetszőleges

dekoracio.html

```
7        .ala { text-decoration: underline; }
8        .fole { text-decoration: overline; }
9        .at { text-decoration: line-through; }
10       .helytelen { text-decoration: wavy red underline; }
11       .kombinalt { text-decoration: overline underline blue dotted; }

15     <p>A szavak <span class="ala">alá</span> és <span class="fole">
16     fölé</span> is lehet vonalat húzni, de <span class="at">ki is
17     húzhatjuk</span>, a <span class="helytelen">hejtelenül</span>
```

A szavak alá és fölé is lehet vonalat húzni, de ~~ki is húzhatjuk~~, a hejtelenül írt szavakat megjelölhetjük és mindenféle kombinációkat is kitalálhatunk.

`text-shadow`: szöveg árnyéka

`h-shadow v-shadow blur-radius color`

vízszintes eltolás, függőleges eltolás, elmosás mértéke, szín.

Az elmosás mértéke elhagyható, a többi kötelező. Az eltolásoknál negatív értékek megengedettek. Vesszővel elválasztva több árnyék is megadható egyszerre.

`none` nincs árnyék, alapértelmezés

arnyek.html

```
7        .dupla { text-shadow: #666666 1px 2px, #AAAAAA 2
             px 4px; }
8        .neon { text-shadow: 0px 0px 10px red; }
9        .lathatatlan {
10         color: white;
11         text-shadow: 2px 2px 10px black;
12       }

16     <p class="dupla">Dupla árnyékot vet a szöveg.</p>
17     <p class="neon">Tiszta '80-as évek!</p>
18     <p class="lathatatlan">Csak az árnyék miatt
           olvasható.</p>
```

Dupla árnyékot vet a szöveg.

Tiszta '80-as évek!

Csak az árnyék miatt olvasható.

`line-height`: sormagasság

`normal`
: betűméretből következő sormagasság, alapértelmezett

szám
: az aktuális betűméretet ezzel szorozva kapja meg a sormagasságot

távolság
: rögzített sormagasság, CSS mértékegységben

%
: az aktuális betűméret %-a

sormagassag.html

```
7         p { border: 1px solid black; }
8         .masfeles { line-height: 1.5; }
9         .dupla { line-height: 200%; }

13      <p>Normál sormagasság.</p>
14      <p class="masfeles">Másfeles sormagasság.</p>
15      <p class="dupla">Kétszeres sormagasság.</p>
```

Normál sormagasság.

Másfeles sormagasság.

Kétszeres sormagasság.

Többféle írásirány támogatott egyazon oldalon

irasirany.html

```
7        .jobbrolBalra {
8          direction: rtl;
9          unicode-bidi: bidi-override;
10       }

14     <p>Balról jobbra.</p>
15     <p class="jobbrolBalra">Jobbról balra.</p>
```

Balról jobbra.

.arlab lórbboJ

Szövegformázás

Készítse el az ábrán látható weboldalt!

- Induljon ki a szoveg.txt fájlból!
- A címsor betűi között hagyjon 5-5 képpontnyi helyet,
- írja csupa nagybetűvel, és
- jelenítsen meg alatta 3-3 képpontnyival jobbra és lefelé eltolt árnyékot, mely kék színű, és elmosásának sugara 10 képpont!
- A bekezdés legyen sorkizárt igazítású,
- a sormagasság másfélszeres,
- az első sor behúzása 20 képpontnyi,
- és automatikusan elválasztott!
- Az emberek neveit emelje ki zöld színű, dupla aláhúzással!

szoveg.html

ATARI

Az Atari egy amerikai vállalat, a világ egyik leghíresebb videójáték-konzol, illetve videójáték fejlesztő/gyártó cége. 1972-ben Nolan Bushnell és Ted Dabney alapította. Megalapításától videójátékokkal foglalkozott, ekkor jelentek meg a híres, népszerű Computer Space és Pong játékok. (A kettő közül a Pong számított a legforradalmibbnak, hiszen mérföldkőnek számít a videójáték-történelemben.) Első konzoljuk az Atari VCS (Video Computer System) volt, amely később az Atari 2600 nevet kapta. 1976-tól otthoni számítógépeket is gyártottak (Atari 400/800, később XL-sorozat). Az 1983-as videójáték-válság alaposan keresztbe tett a cég működésének. A nehéz helyzetbe került céget 1984-ben Jack Tramiel, a Commodore alapítója és korábbi tulajdonosa vásárolta meg. Az ő vezetése alatt mutatták be 1985-ben a 16 bites, Motorola 68000 processzoron alapuló Atari ST számítógépcsaládot, amely a nyolcvanas-kilencvenes évek fordulóján igen népszerű volt. Az olcsó PC-k elterjedésével az Atari folyamatosan piacot veszített, és hiába mutatta be az igen fejlett, Motorola 68030 alapú Atari Falcon számítógépet és a 64 bites játékkonzolok előfutárát, az Atari Jaguart, a csődöt nem kerülhette el. Az Atari Corporation 1996-ban átalakult, majd megszüntették. Az értékes Atari márkanevet 1998-ban a Hasbro játékgyártó világcég vette meg. 2001-ben az Infogrames szoftvercég vette meg a Hasbro Interactive nevű leányvállalatot az Atari márkához kapcsolódó jogokkal együtt. A cég jelenleg Atari SA. néven működik.

Ikonok

Ikonok:

- feldobják az oldal megjelenését
- lényegretörő, kifejező

Használat: `<i>` vagy `<span>` elemek + `class` attribútum

CSS-sel formázhatóak: méret, szín, árnyék, szegély, stb.

Több ikon könyvtár is elérhető.

Font Awesome

- 1588 ikon ingyen + 7842 fizetős
- Kit kódot kell igényelni a használathoz (az enyém pl. 5e6bd4a2fc)
- Egy `<script>` elemet kell beszúrni az oldalba a használathoz
- `class` attribútum értékei az ikonkészlet választáshoz:
  - `fas` → Font Awesome Solid
  - `fab` → Font Awesome Brand
  - `Regular`, `Light`, `Duotone` a fizetős változatban
- `class` attribútum értékei méretezéshez (enélkül a betűmérethez igazodik):
  - `fa-xs`, `fa-sm`, `fa-lg`
  - `fa-2x`, `fa-3x`, ..., `fa-10x`

Ikonok

- Léteznek azonos szélességű ikonok, pl. függőlegesen elrendezett menühöz (`fa-fw`)
- Felsorolásjelekként állhatnak (pl. `fa-ul`, `fa-li`, `fa-check-square`)
- Forgathatóak, tükrözhetőek (pl. `fa-rotate-90`, `fa-flip-vertical`)
- Animálhatóak (`fa-spin`, `fa-pulse`)
- Fizetős változat további szolgáltatásokkal: méretezés, pozicionálás, maszkolás, rétegek, ...
- Egy lehetséges alternatíva: Line Awesome

fontawesome.html

```
7         .nagy { font-size: 32pt; }
8         .kek { color: blue; }

12      <p class="nagy">Mit tanulunk?
13        <i class="fab fa-css3"></i></p>
14      <p class="kek">Tetszik?
15        <i class="fas fa-thumbs-up fa-5x"></i></p>
16      <p>Jó ötlet <a href="https://fontawesome.com">
17        Font Awesome</a>-ot használni?
18        <i class="fas fa-lightbulb"></i></p>
19      <p>Megmozgatja a fantáziát?
20        <i class="fas fa-cog fa-spin"></i></p>
21      <script src="https://kit.fontawesome.com/5e6bd4a2fc.js"
22        crossorigin="anonymous"></script>
23    </body>
```

Mit tanulunk?

Tetszik?

Jó ötlet Font Awesome-ot használni?

Megmozgatja a fantáziát?

Google Material Icons

- Egységes, letisztult UI készítéséhez használható ikonkészlet, és UI tervezési elvek
- Kevesebb ikon, egyszerűbb szolgáltatások, ligatúrákat használ.
- Ikonlista
- Felhasználási útmutató
- Egy stíluslap betöltése után használható
- Az oldal szerkesztőjének kell stílusokat definiálni a méretezéshez (pl. `md-48`), színezéshez (pl. `md-dark`), stb.

material.html

```
6     <link rel="stylesheet" href="https://fonts.googleapis.
        ↪ com/icon?family=Material+Icons">
7     <style>
8       /* Rules for sizing the icon. */
9       .material-icons.md-18 { font-size: 18px; }
10      .material-icons.md-24 { font-size: 24px; }
11      .material-icons.md-36 { font-size: 36px; }
12      .material-icons.md-48 { font-size: 48px; }

24    <p>Méretek:
25      <i class="material-icons␣md-18">android</i>
26      <i class="material-icons␣md-24">android</i>
27      <i class="material-icons␣md-36">android</i>
28      <i class="material-icons␣md-48">android</i>
29    </p>
```

Méretek: android android android android

Aktív: thumb_up, inaktív: thumb_up

A *látszólagos osztály* (pseudo class) egy, a szelektor utáni :-ot követő kulcsszó, amivel a kiválasztott elem(ek) különféle állapotaiban alkalmazandó formázás adható meg.

Látszólagos osztályok referenciája

Hivatkozásoknál alkalmazható:

`link`
: Még nem követték a hivatkozást

`visited`
: Már követték a hivatkozást

`hover`
: Egér a hivatkozás felett

`active`
: Már rákattintottak, de az új tartalom még nem töltődött be

Ebben a sorrendben kell definiálni őket!

hivatkozas.html

```
7        a:link    { color: red; }
8        a:visited { color: green; }
9        a:hover   { color: orange; }
10       a:active  { color: brown; }

14     <p><a
15     href="https://hu.wikipedia.org/wiki/Semmelweis_Ign%C3%A1c">Olvasgasson</a>
16     Semmelweis Ignácról és a kézmosás fontosságáról!</p>
```

Olvasgasson Semmelweis Ignácról és a kézmosás fontosságáról!

Egérkurzor alakja

- Rengeteg előre beépített egérkurzor közül lehet választani a `cursor` tulajdonság értékének állításával.
- Saját kurzort is lehet betölteni az `url()` függvénnyel, de célszerű utána vesszővel elválasztva egy beépített kurzort is megnevezni, betöltési hiba esetére.
- Kurzor akár online is szerkeszthető.

kurzor.html, cursor.cur

`list-style-type`

Felsorolásjel típusa

- Számozott listákhoz, pl. `decimal` (arab számok), `lower-alpha` (kisbetűk), `upper-roman` (római számok nagybetűkkel)
- Nem számozott listákhoz, pl. `disc` (körlemez), `circle` (körvonal), `square` (négyzet)
- Eltüntetés: `none`

szamozott.html

Válasszon a felsorolásjelek közül!

- decimal
- armenian
- cjk-ideographic
- decimal-leading-zero
- georgian
- hebrew
- hiragana
- hiragana-iroha
- katakana
- katakana-iroha
- lower-alpha
- lower-greek
- lower-latin
- lower-roman
- upper-alpha
- upper-greek
- upper-latin
- upper-roman
- none

1. Répa
2. Retek
3. Mogyoró

nemszamozott.html

Válasszon a felsorolásjelek közül!

- disc
- circle
- square
- none

- Répa
- Retek
- Mogyoró

`list-style-image`

A felsorolásjel egy kép `url()` függvénnyel, vagy `none`. Érdemes megadni a `list-style-type`-ot is, hátha nem lehet megjeleníteni a képet.

sajatjelolo.html

```
7        ul { list-style-image: url("tick.svg"); }
```

- Répa
- Retek
- Mogyoró

list-style-position

A felsorolásjel helyzete

- outside: a bekezdés bal széle előtt, alapértelmezés.
- inside: a bekezdésen belül, a szöveg részeként.

kivulbelul.html

```
7   ul.kivul { list-style-position: outside; }
8   ul.belul { list-style-position: inside; }
9   li { border: 1px solid black; }
```

Jelölő kívül:

- Répa
  Répa
  Répa
- Retek
  Retek
  Retek
- Mogyoró
  Mogyoró
  Mogyoró

Jelölő belül:

- Répa
  Répa
  Répa
- Retek
  Retek
  Retek
- Mogyoró
  Mogyoró
  Mogyoró

Rövidítés:

`list-style: list-style-type list-style-position list-style-image`

Ha bármelyik hiányzik, az alapértelmezett értéket fogják használni.

Készítse el a mellékelt ábrának megfelelő weboldalt! A felső rész felsorolásjeleként használja a rouge.png fájlt!

Táblázatok alapértelmezett formázása:

- Cellaméretek a tartalomhoz igazodnak
- Nincsenek szegélyek
- Fejléc (`<th>`) cellák félkövérek, középre zártak
- Normál cellák (`<td>`) balra zártak

tablazat01.html

Termék	Ár (Ft)
AdeZ növényi alapú ital	599
After Eight Classic desszert	899
Agri Garden professional tömlő	3999

Kaphat szegélyt a teljes táblázat ...

tablazat02.html

```
7        table { border: 1px solid black; }

11     <table>
12       <tr><th>Termék</th><th>Ár (Ft)</th></tr>
13       <tr><td>AdeZ növényi alapú ital</td><td>599</td></tr>
14       <tr><td>After Eight Classic desszert</td><td>899</td></tr>
15       <tr><td>Agri Garden professional tömlő</td><td>3999</td></tr>
16     </table>
```

Termék	Ár (Ft)
AdeZ növényi alapú ital	599
After Eight Classic desszert	899
Agri Garden professional tömlő	3999

...vagy annak cellái ...

tablazat03.html

```
7        th, td { border: 1px solid black; }
```

Termék	Ár (Ft)
AdeZ növényi alapú ital	599
After Eight Classic desszert	899
Agri Garden professional tömlő	3999

...vagy mindkettő.

tablazat04.html

```
7        table, th, td { border: 1px solid black; }
```

Termék	Ár (Ft)
AdeZ növényi alapú ital	599
After Eight Classic desszert	899
Agri Garden professional tömlő	3999

A cellák közötti távolság változtatható a <table> `border-spacing` tulajdonságával ...

tablazat05.html

```
7        table { border-spacing: 15px; }
8        table, th, td { border: 1px solid black; }
```

Termék	Ár (Ft)
AdeZ növényi alapú ital	599
After Eight Classic desszert	899
Agri Garden professional tömlő	3999

...de teljesen el is tüntethető a <table> `border-collapse` tulajdonságával:

`separate` elkülönített cellák, alapértelmezés.

`collapse` összevont cellaszegélyek.

tablazat06.html

```
7        table { border-collapse: collapse; }
8        th, td { border: 1px solid black; }
```

Termék	Ár (Ft)
AdeZ növényi alapú ital	599
After Eight Classic desszert	899
Agri Garden professional tömlő	3999

A cellák szegélye és tartalma közötti távolság a `<td>`, `<th>` `padding` tulajdonságával állítható.

tablazat07.html

```
7        table { border-collapse: collapse; }
8        th, td {
9          border: 1px solid black;
10         padding: 5px;
11       }
```

Termék	Ár (Ft)
AdeZ növényi alapú ital	599
After Eight Classic desszert	899
Agri Garden professional tömlő	3999

A cellákon belüli igazítás a `text-align`, `vertical-align` tulajdonságokkal állítható.

tablazat08.html

```
7       table { border-collapse: collapse; }
8       th, td {
9         height: 2em;
10        border: 1px solid black;
11        text-align: left;
12        vertical-align: bottom;
13      }
```

Termék	Ár (Ft)
AdeZ növényi alapú ital	599
After Eight Classic desszert	899
Agri Garden professional tömlő	3999

Ha a táblázat szélesebb, mint amit a tartalom indokol, a maradék hely arányosan lesz elosztva.

tablazat09.html

```
7        table {
8          border-collapse: collapse;
9          width: 100%
10       }
11       th, td {
12         border: 1px solid black;
13         text-align: left;
14       }
```

Termék	Ár (Ft)
AdeZ növényi alapú ital	599
After Eight Classic desszert	899
Agri Garden professional tömlő	3999

Az olvashatóság javításához pl. kiemelhetjük az egér alatti sort a `:hover` látszólagos osztállyal. Figyeljük meg a „csíkozást”!

Termék	Ár (Ft)
AdeZ növényi alapú ital	599
After Eight Classic desszert	899
Agri Garden professional tömlő	3999

tablazat10.html

```
table    { border-collapse: collapse; }
tr:hover { background-color: yellow; }
th, td {
  border-bottom: 1px solid lightgray;
  padding: 5px;
}
th {
  text-align: left;
  border-top: 1px solid lightgray;
}
```

Az olvashatóság a sorok váltakozó háttérszínével is javítható. Egy szülő elem (pl. `<table>`) gyerekei (sorok) közül meghatározottak kiválasztása: `nth-child()` látszólagos osztállyal.

`An+b`

$n = 0 \ldots$, de a gyerekek számozása 1-től indul!

`even`

Páros elemek $\equiv$ `2n`

`odd`

Páratlan elemek $\equiv$ `2n+1`

Termék	Ár (Ft)
AdeZ növényi alapú ital	599
After Eight Classic desszert	899
Agri Garden professional tömlő	3999

tablazat11.html

```
7       table { border-collapse: collapse; }
8       tr:nth-child(even) { background-color: #f0f0f0; }
9       th, td {
10        border-top: 1px solid lightgray;
11        padding: 5px;
12      }
13      th {
14        text-align: left;
15        background-color: yellow;
16      }
17      td { border-bottom: 1px solid lightgray; }
```

Görgethető táblázat, trükkök:

- Rögzített fejléc → két külön táblázat, csak az alsó görgethető
- Görgetés: táblázat beágyazva egy túl alacsony elembe, és `overflow-y: scroll` (hasonlóan a vízszintes irányú görgetés is lehetséges az `overflow-x` tulajdonsággal)
- Legyenek a két táblázat oszlopai rendre azonos szélességűek!

Termék	Ár (Ft)
After Eight Classic desszert	899
Agri Garden professional tömlő	3999
Kit Kat, Smarties vagy Boci Buborék	549
Bords Eve margarin	419
Kunsági dobozos tej	189

tablazat12.html

```
28      <table>
29        <colgroup>
30          <col class="bal" />
31          <col class="jobb" />
32        </colgroup>
33        <tr><th>Termék</th><th>Ár (Ft)</th></tr>
34      </table>
35      <div class="gorget">
36        <table>
37          <colgroup>
38            <col class="bal" />
39            <col class="jobb" />
40          </colgroup>
41          <tr><td>AdeZ növényi alapú ital</td><td>599</td></tr>

51          <tr><td>Xixo Ice Tea</td><td>398</td></tr>
52        </table>
53      </div>
```

tablazat12.html

```
7       table { border-collapse: collapse; }
8       tr:nth-child(even) { background-color: #f0f0f0; }
9       th, td {
10        border-top: 1px solid lightgray;
11        padding: 5px;
12      }
13      th {
14        text-align: left;
15        background-color: yellow;
16      }
17      td { border-bottom: 1px solid lightgray; }
18      div.gorget {
19        height: 8em;
20        overflow-y: scroll;
21        display: inline-block;
22      }
23      col.bal  { width: 350px; }
24      col.jobb { width: 80px; }
```

Táblázat címkéjének elhelyezése: `caption-side`

`top`
: Fent, alapértelmezés.

`bottom`
: Lent.

Üres cellák szegélyeinek rajzolása: `empty-cells`

`show`
: Megrajzolja, alapértelmezés.

`hide`
: Nem rajzolja meg.

Táblázat megjelenítési algoritmus: `table-layout`

`auto`

A cellák szélessége a legszélesebb nem tördelhető tartalmi elem függvénye. Alapértelmezés.

`fixed`

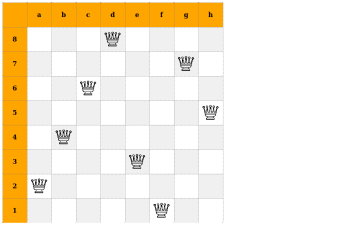
Az oszlopok szélességét vagy

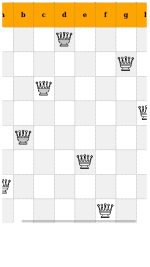
- a `<table>` és `<col>` elemek szélességei adják meg, vagy
- az első sor celláinak szélességei. Ha ilyen nincs, egyforma szélesek lesznek a cellák.

Nagy táblázatoknál jelentősen gyorsabb megjelenés.

Készítse el a queen.png fájl felhasználásával az alábbi weboldalt, ami a Nyolckirálynő probléma egy lehetséges megoldását mutatja! Ügyeljen rá, hogy keskeny kijelzőkön a táblázat vízszintesen görgethető legyen! (A cellák 50x50, a képek 40x40px méretűek.)

sakk.html





A `display` tulajdonsággal állítható egy elem *megjelenése*:

`block`
: Blokkszintű megjelenítés

`inline`
: Soron belüli megjelenítés

`inline-block`
: Soron belüli blokk megjelenítés

`none`
: Nincs megjelenítés

Blokkszintű elemek

Pl. <p>, . Új sorban kezdődik, vízszintesen elfoglalja a teljes rendelkezésre álló helyet (átállítható).

Soron belüli elemek

Pl. <a>, . Sor (blokk) belsejében kezdődik, annyi helyet foglal, amennyit a tartalom kíván.

Soron belüli blokkok

A két módszer vegyítése. Beállítható a szélesség, magasság, alsó és felső margók, kitöltések, mint a blokkszintű megjelenítésnél. Viszont az elem nem kerül új sorba.

JavaScript programok gyakran használják a `none` értéket bizonyos tartalmak ideiglenes elrejtésére/megjelenítésére a DOM módosítása helyett.

meretezes.html

```
7        label {
8          display: inline-block;
9          width: 100px;
10       }
11       input[type=text] {
12         width: 200px;
13         box-sizing: border-box;
14       }
15       select {
16         width: 200px;
17       }
```

Név: Pl. Gipsz Jakab

Végzettség: középiskola/gimnázium

A `visibility` tulajdonsággal állítható egy elem *láthatósága*:

`visible`
: Látható.

`hidden`
: Rejtett, de az elhelyezéséhez szükséges helyet a böngésző fenntartja!

A `position` tulajdonsággal befolyásolható az elemek elhelyezési módja az oldalon. Értékei:

`static`
: Alapértelmezett elrendezés. A böngésző dönt az elemek helyéről és ált. a méretezéséről is (pl. blokkszintű elemek egymás alatt/egymáson belül, széltében kitöltik a rendelkezésre álló helyet, soron belüliek a blokkszintűek belsejében, stb.)

static.html

```
p {                                  7
  /* position: static; */            8
  border: 1px solid black;           9
  padding: 10px;                     10
  background-color: yellow;          11
}                                    12

<p>Első bekezdés.</p>                16
<p>Második bekezdés.</p>             17
<p>Harmadik bekezdés.</p>            18
```

Első bekezdés.

Második bekezdés.

Harmadik bekezdés.

relative

Az elem az eredeti helyéhez képest eltolható, de ezt a régi helyet a böngésző fenntartja. Megadható az elem tetejének (top), aljának (bottom), bal (left) és jobb oldalának (right) relatív helyzete.

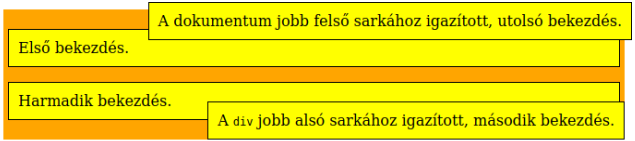
relative.html

```
7  p {
8    border: 1px solid black;
9    padding: 10px;
10   background-color: yellow;
11 }
12 p.eltolas {
13   position: relative;
14   left: 30px;
15   top: 45px;
16 }

20 <p>Első bekezdés.</p>
21 <p class="eltolas">Második bekezdés.</p>
22 <p>Harmadik bekezdés.</p>
```

absolute

A legközelebbi, nem statikusan elhelyezett szülő elemhez, annak hiányában a dokumentum testéhez képest relatívan megadott helyre igazítja az elemet. Az elem eredeti helyét nem őrzi meg a böngésző.



absolute.html

absolute.html

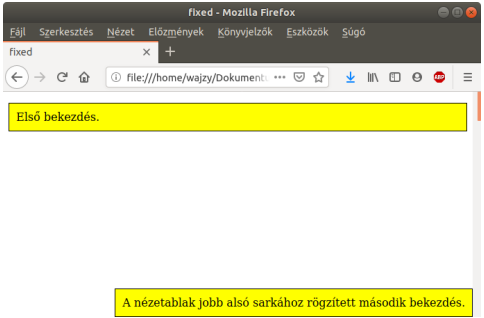
```
p {
  border: 1px solid black;
  padding: 10px;
  background-color: yellow;
}
p.jobbFenn {
  margin: 0px;
  position: absolute;
  right: 0px;
  top: 0px;
}
div {
  position: relative;
  padding: 5px;
  background-color: orange;
}
```

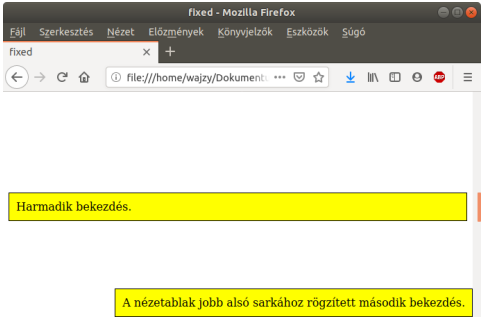
```
p.jobbLenn {
  margin: 0px;
  position: absolute;
  right: 0px;
  bottom: 0px;
}


<div>
  <p>Első bekezdés.</p>
  <p class="jobbLenn">A <code>div
    </code> jobb alsó sarkához
    igazított, második bekezdés.</p>
  <p>Harmadik bekezdés.</p>
</div>
<p class="jobbFenn">A dokumentum jobb
  felső sarkához igazított, utolsó
  bekezdés.</p>
```

fixed

Az elem pozíciója a nézetablakhoz képest relatív, görgetés közben is a helyén marad.





fixed.html – Figyeljék a gördítősávot!

fixed.html

```
7   p {
8     border: 1px solid black;
9     padding: 10px;
10    background-color: yellow;
11    margin-bottom: 1000px;
12  }
13  p.jobbLenn {
14    margin: 0px;
15    position: fixed;
16    right: 0px;
17    bottom: 0px;
18  }
```

```
22  <p>Első bekezdés.</p>
23  <p class="jobbLenn">A nézetablak jobb
24    alsó sarkához rögzített második
25    bekezdés.</p>
26  <p>Harmadik bekezdés.</p>
```

sticky

A `relative` és a `fixed` kombinációja; az elem gördül a nézetablak tartalmával, amíg el nem ér egy adott helyre, ahová „odaragad”.

Részleges böngésző támogatás; pl. Safarin a `-webkit-sticky` értékkel működik csak.

sticky.html – Próbálják görgetni a tartalmat!

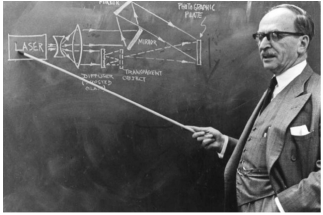
sticky.html

```
7       p, h1, h2 {
8         border: 1px solid black;
9         padding: 10px;
10      }
11      p {
12        background-color: yellow;
13        padding-bottom: 500px;
14      }
15      h1 {
16        background-color: blue;
17      }
18      h2 {
19        background-color: navy;
20        color: white;
21        padding-top: 15px;
22        padding-bottom: 5px;
23      }
```

```
    h1, h2 {                          24
      position: sticky;               25
      top: 0px;                       26
      height: 40px;                   27
      margin: 20px 0px;               28
    }                                 29

  <h1>1. fejezet</h1>                 33
  <h2>1.1. fejezet</h2>               34
  <p>Bekezdés...</p>                  35
  <h2>1.2. fejezet</h2>               36
  <p>Bekezdés...</p>                  37
  <h1>2. fejezet</h1>                 38
  <p>Bekezdés...</p>                  39
```

Próbálja meg elkészíteni az alábbi weboldalt a gabor.txt és gabor.jpeg fájlok felhasználásával!



Gábor Dénes

Gábor Dénes (Dennis Gabor, született Günszberg; Budapest, Terézváros, 1900. június 5. - London, 1979. február 9.) Nobel-díjas magyar fizikus, gépészmérnök, villamosmérnök, a holográfia feltalálója.

Élete

A Terézvárosban született, zsidó családban. Édesapja, Günszberg Bernát a Magyar Általános Kőszénbánya Részvénytársaságnál (MÁK Rt.) dolgozott, főkönyvelőként, majd cégvezetőként, végül igazgatóként, 1928-ig. Édesanyja Jakobovits Adél. A szülők 1899-ben kötötték házasságot, s három gyermekük született: Dénes (1900), György (1901) és Endre (1903). Az apa 1902. március 8-án fiaival együtt engedélyt kapott, hogy családnevüket „Gábor"-ra változtassák.

Gábor Dénes

Gábor Dénes (Dennis Gabor, született Günszberg; Budapest, Terézváros, 1900. június 5. - London, 1979. február 9.) Nobel-díjas magyar fizikus, gépészmérnök, villamosmérnök, a holográfia feltalálója.

Élete

A Terézvárosban született, zsidó családban. Édesapja, Günszberg Bernát a Magyar Általános Kőszénbánya Részvénytársaságnál (MÁK Rt.) dolgozott, főkönyvelőként, majd cégvezetőként, végül igazgatóként, 1928-ig. Édesanyja Jakobovits Adél. A szülők 1899-ben kötötték házasságot, s három gyermekük született: Dénes (1900), György (1901) és Endre (1903). Az apa 1902. március 8-án fiaival együtt engedélyt kapott, hogy családnevüket „Gábor"-ra változtassák.

A Szemere utcai községi elemi népiskolába járt (1906-1910), a középiskolát a Budapesti V. ker. Markó utcai Magyar Királyi Állami Főreáliskolában (1910–1918) végezte. 1911. november 14-én lelt az Aeroplán körhinta szabadalmi leírása (a bejelentés napja: 1910. október 8.). 1918. március 6-án érettségizett. 1918. március 15-én behívták katonának, az észak-italiai fegyverszünet után tért haza. Ehhez kapcsolódik olasz nyelvtanulása, amely negyedik nyelvismerete lett. Novemberben beiratkozott a Magyar királyi József nádor Műszaki és Gazdásagtudományi Egyetem gépészmérnöki osztályába. 1919. május 24-én áttért az evangélikus vallásra. 1920-tól Berlinben folytatta tanulmányait a charlottenburgi Technische Hochschule elektromérnöki karán. Itt rendszeresen látogatta a tudományegyetem előadásait, többek között Albert Einstein szemináriumát, mely Szilárd Leó kezdeményezésére jött létre, és aki az előadásokra meghívta Wigner Jenőt, Neumann Jánost és Gábor Dénest. Később a magyar baráti kör Polányi Mihállyal, Kösztler Artúrral bővült. 1921. június 24-én tette le első szigorlatát a Műegyetemen. 1924-ben mérnöki diplomát szerzett a berlini Humboldt Egyetemen.

Gábor Dénes
Élete

A Terézvárosban született, zsidó családban. Édesapja, Günszberg Bernát a Magyar Általános Kőszénbánya Részvénytársaságnál (MÁK Rt.) dolgozott, főkönyvelőként, majd cégvezetőként, végül igazgatóként, 1928-ig. Édesanyja Jakobovits Adél. A szülők 1899-ben kötötték házasságot, s három gyermekük született: Dénes (1900), György (1901) és Endre (1903). Az apa 1902. március 8-án fiaival együtt engedélyt kapott, hogy családnevüket „Gábor"-ra változtassák.

A Szemere utcai községi elemi népiskolába járt (1906-1910), a középiskolát a Budapesti V. ker. Markó utcai Magyar Királyi Állami Főreáliskolában (1910–1918) végezte. 1911. november 14-én lelt az Aeroplán körhinta szabadalmi leírása (a bejelentés napja: 1910. október 8.). 1918. március 6-án érettségizett. 1918. március 15-én behívták katonának, az észak-italiai fegyverszünet után tért haza. Ehhez kapcsolódik olasz nyelvtanulása, amely negyedik nyelvismerete lett. Novemberben beiratkozott a Magyar királyi József nádor Műszaki és Gazdásagtudományi Egyetem gépészmérnöki osztályába. 1919. május 24-én áttért az evangélikus vallásra. 1920-tól Berlinben folytatta tanulmányait a charlottenburgi Technische Hochschule elektromérnöki karán. Itt rendszeresen látogatta a tudományegyetem előadásait, többek között Albert Einstein szemináriumát, mely Szilárd Leó kezdeményezésére jött létre, és aki az előadásokra meghívta Wigner Jenőt, Neumann Jánost és Gábor Dénest. Később a magyar baráti kör Polányi Mihállyal, Kösztler Artúrral bővült. 1921. június 24-én tette le első szigorlatát a Műegyetemen. 1924-ben mérnöki diplomát szerzett a berlini Humboldt Egyetemen.

Az 1920-as években a nagyfeszültségű hálózatok üzemében fellépő tranziens jelenségek sok problémát okoztak, de a vizsgálatukhoz sem módszerek, sem eszközök nem álltak rendelkezésre. 1927-ben disszertációjában tranziens jelenségek rögzítése érdekében az oszcillográf érzékenységének növelését dolgozta ki. 1927-1932 között Siemensstadtban, a Siemens és Halske kutatólaboratóriumában, 1932-1933-ban pedig Erlangenben, a Siemens-Reiniger-Veifa nevű cégnél dolgozott. 1933-ban, a náci hatalomátvétel után elhagyta Németországot és hazatért Magyarországra. 1933-tól 1934-ig az Egyesült Izzó kutatólaboratóriumában a gázkisülés fizikájával foglalkozott. 1934-ben végleg letelepedett Angliában, brit állampolgár lett. 1934-től 1948-ig a British Thomson-Houston Társaság kutatólaboratóriumában dolgozott Rugbyben (Warwickshire).

1936. augusztus 8-án feleségül vette Marjorie Louise Butlert, akivel haláláig harmonikus házasságban élt, gyermekük nem született. 1942. október 25-én Budapesten meghalt édesapja. 1946-ban édesanyja hozzá és testvéréhez Angliába költözött. 1947-ben itt találta fel a holográfiát, amiért később, 1971-ben fizikai Nobel-díjat kapott. A holográfia azonban 1960-ig, a lézer feltalálásáig nem terjedt el. 1949-1958 között az Imperial College-ban elektronikát adott elő. 1956-ban a Royal Society a tagjává választotta. 1958-tól 1967-ig az alkalmazott elektronfizika professzora volt az Imperial College-ban.

Gábor Dénes az emberi kommunikációt és a hallást is tanulmányozta.

1962-ben látogatott haza Magyarországra. 1967-ben nyugalomba vonult. 1968-ban részt vett a Római Klub alapításában. „A holografikus módszer feltalálásáért és a kifejlesztéséhez való hozzájárulásáért" 1971-ben megkapta a fizikai Nobel-díjat. 1974-ben súlyos agyvérzést szenvedett. 1979. február 9-én hunyt el Londonban.

Emlékezete

- Nevét megőrökíti a Gábor-érem (Gabor Medal).
- A „NOVOFER Alapítvány a Műszaki-Szellemi Alkotásért" 1989-ben hozta létre a Gábor Dénes-díjat, amelyet minden évben azoknak a - határainkon belül és azokon kívül élő - magyar természettudósoknak, mérnököknek, feltalálóknak, kutatóknak, oktatóknak adományoznak, akik kiemelkedő

Az oldal elvárt viselkedése a következő:

- A kép egyhelyben áll a görgetés hatására is. A felfelé gördülő szöveg lassan rácsúszik, majd teljesen kitakarja azt.
- Az első szintű címsor addig csúszik felfelé, amíg el nem éri a nézetablak tetejét. Utána ottmarad, és a bekezdések szövege alágördül.
- A második szintű címsor hasonlóan viselkedik az első szintűhöz, de az alatt áll meg, nem takarja le azt. A második szintű címsorok viszont letakarhatják egymást.
- A cikk maximális szélessége 640 képpont. Ha ennél több hely áll rendelkezésre, középre kell igazítani.
- A szöveg sorkizárt igazítású, automatikusan elválasztott.
- Az idézetek előtt és után magyar stílusú („ ”) idézőjeleket alkalmaz.
- A hivatkozás új oldalon nyílik meg.

Számlálók: mint egy program változói, melyek értéke változik (ált. nő) a használat hatására
Használható pl. fejezetek vagy felsorolások számozására

`counter-reset`: számláló létrehozása, újraindítása vagy kezdőértékkel ellátása.

`none`
: Számlálók nem lesznek inicializálva, alapértelmezés.

`id number`
: Az *id* azonosítójú számláló felveszi a *number* értéket. Utóbbi elhagyható, alapértelmezetten 0.

`counter-increment`: számláló értékének léptetése

`none`
: Nem változtat a számlálók értékén, alapértelmezés.

`id number`
: Hozzáad *number*-t (alapértelmezetten 1, de lehet akár negatív érték is) az *id* számláló értékéhez.

A `content` tulajdonságot használják a számlálók értékének kijelzésére, jellemzően a `::before` látszóleges elemben (`::marker` gyengén támogatott). Ugyanitt ált. számláló növelésre is sor kerül.

`counter(id)` vagy `counter(id, style)`: számláló értékének lekérdezése

`id`
: A `counter-reset`-tel létrehozott és `counter-increment`-tel léptetett változó azonosítója.

`style`
: A számláló megjelenítési módja, pl. `lower-alpha`, `upper-roman`, `decimal-leading-zero`, stb., vagy a `symbols()` függvénnyel adott szimbólumok.

szamozottFejezet.html

```
7       body { counter-reset: cimsor1; }
8       h1   { counter-reset: cimsor2; }
9       h1::before {
10        counter-increment: cimsor1;
11        content: counter(cimsor1) "␣";
12      }
13      h2::before {
14        counter-increment: cimsor2;
15        content: counter(cimsor1) "."
            ↪ counter(cimsor2, upper-
            ↪ alpha) "␣";
16      }
```

```
<body>                                                    19
  <h1>Első fejezet</h1>                                   20
  <h2>Első fejezet első alfejezete</                      21
      ↪ h2>
  <p>Bekezdés...</p>                                      22
  <h2>Első fejezet második alfejezete                     23
      ↪ </h2>
  <p>Bekezdés...</p>                                      24
  <h1>Második fejezet</h1>                                25
  <h2>Második fejezet első alfejezete                     26
      ↪ </h2>
  <p>Bekezdés...</p>                                      27
</body>                                                   28
```

1 Első fejezet

1.A Első fejezet első alfejezete

Bekezdés...

1.B Első fejezet második alfejezete

Bekezdés...

2 Második fejezet

2.A Második fejezet első alfejezete

Bekezdés...

szamozottFejezet.html

Egymásba ágyazott azonos típusú elemeknél (pl. `<ol>`, `<ul>`) rekurzívan mindig új számláló keletkezik. Megjelenítésük: `counters(id, string)` vagy `counters(id, string, style)` függvénnyel.

`id`
: A számláló neve.

`string`
: A számlálók értékeit elválasztó karakterlánc.

`style`
: A számlálók stílusa.

szamozottLista.html

```
7   ol {
8     counter-reset: szint;
9     list-style-type: none;
10  }
11  li::before {
12    color: blue;
13    font-weight: bold;
14    counter-increment: szint;
15    content: counters(szint, ".") " ";
16  }
```

```
21  <ol>
22    <li>Első fejezet
23    <ol>
24      <li>Első fejezet első alfejezet</li>
25      <li>Első fejezet második alfejezet</li>
26    </ol></li>
27    <li>Második fejezet
28    <ol>
29      <li>Második fejezet első alfejezet</li>
30    </ol></li>
31  </ol>
```

Tartalomjegyzék

1 Első fejezet
- 1.1 Első fejezet első alfejezet
- 1.2 Első fejezet második alfejezet

2 Második fejezet
- 2.1 Második fejezet első alfejezet

szamozottLista.html

Készítse el a mellékelt teendok.html oldalt!

- A kétszintű felsorolás külső szintjén alkalmazzon nagybetűs római számokat, a belsőn arab számokat!
- Utóbbiak számozása az egész oldalon legyen folytonos (1-6)!
- A számokat helyezze sárga hátterű ellipszisek közepébe, kék színnel és félkövér betűkkel megjelenítve!

Teendők

I Takarítás
1 Mosogatás
2 Porszívózás
II Tanulás
3 HTML
4 CSS
III Bevásárlás
5 Tej
6 Kenyér

A *lebegtetés* egy speciális oldalelrendezési módszer, amivel egy elemet a tárolóján belül mozgathatunk. A `float` tulajdonsággal állítható a lebegtetés iránya:

`left`
: A tároló bal széle felé mozdítja az elemet. Ha nincs ott elég hely, egy „sorral” lejjebb helyezi el.

`right`
: Jobb oldalra mozgat.

`none`
: Nincs lebegtetés, alapértelmezés.

Legegyszerűbb és jellemző felhasználás: kép körbefolyatása szöveggel. Hibalehetőség: a lebegtetett elem túlnyúlhat a tárolón.

lebeg1.html

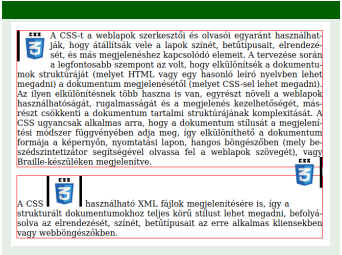
```
7   div {
8     max-width: 600px;
9     margin: auto;
10  }
11  img {
12    width: 10%;
13    border-style: solid;
14    border-width: 0 5px 0 5px;
15  }
```

```
16  img.bal {
17    float: left;
18    border-width: 0 0 0 5px;
19  }
20  img.jobb {
21    float: right;
22    border-width: 0 5px 0 0;
23  }
24  p {
25    text-align: justify;
26    hyphens: auto;
27    border: 1px solid red;
28  }
```

lebeg1.html

```
32      <div>
33        <p><img class="bal" src="css3.svg" alt="CSS3 logo"
34        title="Balra lebegtetett" />A CSS-t a weblapok
            ↪ szerkesztői és
35        olvasói egyaránt használhatják, hogy átállítsák
            ↪ vele a lapok


48        fel a weblapok szövegét), vagy Braille-készüléken
49        megjelenítve.<img class="jobb" src="css3.svg" alt=
            ↪ "CSS3 logo"
50        title="Jobbra lebegtetett" /></p>
51        <p>A CSS <img src="css3.svg" alt="CSS3 logo" title
            ↪ ="Soron belül"
52        /> használható XML fájlok megjelenítésére is, így
            ↪ a strukturált
```



Lebegtetés

Gyakran használják a lebegtetést az oldalelrendezés kialakítására, pl. tartalmak helytől függően egymás mellé vagy alá igazításához.

Első fejezet

Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés.

Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés.

Második fejezet

Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés.

Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés.

lebeg2.html

lebeg2.html

```
7        p {
8          text-align: justify;
9          hyphens: auto;
10         border: 1px solid red;
11         padding: 0 10px;
12         float: left;
13         box-sizing: border-box;
14         max-width: 50%;
15         min-width: 400px;
16       }
```

```
<h1>Első fejezet</h1>                                                    20
<p>Első bekezdés. Első bekezdés. Első                                    21
    ↪ bekezdés. Első bekezdés. Első
bekezdés. Első bekezdés. Első bekezdés                                   22
    ↪ . Első bekezdés. Első
bekezdés. Első bekezdés. Első bekezdés                                   23
    ↪ . Első bekezdés.</p>
<p>Második bekezdés. Második bekezdés.                                    24
    ↪ Második bekezdés. Második
bekezdés. Második bekezdés. Második                                      25
    ↪ bekezdés. Második bekezdés.
Második bekezdés. Második bekezdés. Má                                   26
    ↪ sodik bekezdés. </p>
<h1>Második fejezet</h1>                                                 27
<p>Első bekezdés. Első bekezdés. Első                                    28
    ↪ bekezdés. Első bekezdés. Első
```

Az oldalt lekicsinyítve viszont ismét kellemetlen mellékhatás lép fel:

lebeg2.html

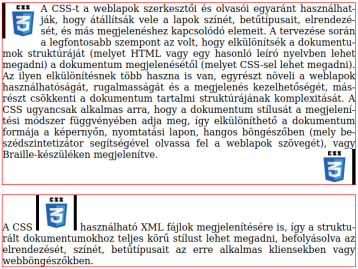
Hogyan lehet úrrá lenni a gondokon?

1. lehetőség: az `overflow` tulajdonság használata, de néha nem kívánt görgetősáv jelenik meg.

lebeg3.html

```
29      .clearfix { overflow: auto; }

34      <p class="clearfix"><img class="bal" src
            ↪ ="css3.svg" alt="CSS3 logo"
35      title="Balra lebegtetett" />A CSS-t a
            ↪ weblapok szerkesztői és
```



Lebegtetés

2. lehetőség: a `clear` tulajdonsággal megadható, hogy egy elem valamely oldalán nem szerepelhet lebegtetett elem.

none
: Az elem mindkét oldalára kerülhet lebegtetett másik elem, alapértelmezés.

left
: Az elem bal oldalára nem kerülhet lebegtetett elem.

right
: Ugyanaz a jobb oldalon.

both
: Egyik oldalon sem lehet lebegtetett elem.

```
17        h1 { clear: left; }
```

lebeg4.html

Első fejezet

Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés.

Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés.

Második fejezet

Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés. Első bekezdés.

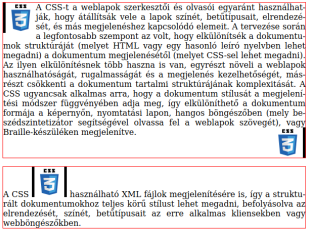
Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés. Második bekezdés.

3. lehetőség: a *clearfix hack* (széles körben alkalmazott)

lebeg5.html

```
29        .clearfix::after {
30          content: "";
31          clear: both;
32          display: table;
33        }

38        <p class="clearfix"><img class="bal" src="
            ↪ css3.svg" alt="CSS3 logo"
39        title="Balra lebegtetett" />A CSS-t a
            ↪ weblapok szerkesztői és
```


A CSS-t a weblapok szerkesztői és olvasói egyaránt használhatják, hogy átállítsák vele a lapok színét, betűtípusait, elrendezését, és más megjelenéshez kapcsolódó elemeit. A tervezése során a legfontosabb szempont az volt, hogy elkülönítsék a dokumentumok struktúráját (melyet HTML vagy egy hasonló leíró nyelvben lehet megadni) a dokumentum megjelenésétől (melyet CSS-sel lehet megadni). Az ilyen elkülönítésnek több haszna is van, egyrészt növeli a weblapok használhatóságát, rugalmasságát és a megjelenés kezelhetőségét, másrészt csökkenti a dokumentum tartalmi struktúrájának komplexitását. A CSS ugyancsak alkalmas arra, hogy a dokumentum stílusát a megjelenítési módszer függvényében adja meg, így elkülöníthető a dokumentum formája a képernyőn, nyomtatási lapon, hangos böngészőben (mely beszédszintetizátor segítségével olvassa fel a weblapok szövegét), vagy Braille-készüléken megjelenítve.

A CSS használható XML fájlok megjelenítésére is, így a strukturált dokumentumokhoz teljes körű stílust lehet megadni, befolyásolva az elrendezését, színét, betűtípusait az erre alkalmas klienseken vagy webböngészőkben.

Lebegtetés

Készítse el a Futókalandorok weboldalának nyitóoldalát az eredeti oldal egyszerűsített változataként! (Nyers szöveg.)

- Készítsen egy vízszintes elrendezésű menüt az oldal tetejére! Ha egy sorban nem férnek el a menüpontok, akkor azokat új sor(ok)ban kell elhelyezni.
- A menüsáv háttere halvány szürke.
- A „Futókalandorok” főcím fekete, a menüpontok (most még sehová sem mutató) hivatkozásai középszürkék.
- Ha az egér a hivatkozás fölé kerül, színe feketére vált.
- A menü és a tartalmi rész között hagyjon ki egy kis helyet!
- A tartalmi rész két hasábos, melyek az oldal szélességének 50-50%-át foglalják el. Hagyjon mindegyikben egy kis belső kitöltést!
- A bal hasábban jelenítse meg a logót!
- A jobb hasáb utolsó bekezdését igazítsa jobbra!

Lebegtetés

futokalandorok.html

Oldalelrendezési módszer: az elemeket egy tároló belsejében jelenítik meg, egy (vagy több) tengely mentén felfűzve. A tároló (pl. elem) `display` tulajdonságának értékei:

`flex`
: Blokkszintű elemként

`inline-flex`
: Soron belüli elemként

flexbox.html

```
div { /* Tároló */
  display: flex;
  border: 1px solid black;
  margin-top: 10px;
  padding: 0px;
}
div > p { /* Tárolt elemek */
  border: 1px solid black;
  margin: 10px;
  padding: 5px;
}
```

A tengely és bejárásának iránya a `flex-direction` tulajdonsággal változtatható:

`row`
: Vízszintes, balról jobbra (alapértelmezés)

`row-reverse`
: Vízszintes, jobbról balra

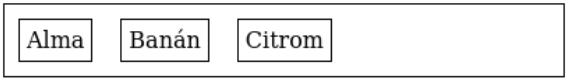
`column`
: Függőleges, fentről le

`column-reverse`
: Függőleges, lentről fel

flexbox.html

```
<p>Balról jobbra</p>
<div style="flex-direction: row">
  <p>Alma</p>
  <p>Banán</p>
  <p>Citrom</p>
</div>
```

Balról jobbra



Ha az elemek szélességeinek összege kisebb, mint a tengely hossza, akkor a `justify-content` tulajdonsággal beállítható, hová helyezzük az elemeket a tengely mentén:

`flex-start`
: Alapértelmezés, a tengely elejére

`flex-end`
: A tengely végére

`center`
: Tengely közepére

`space-between`
: A tárolt elemek között egyforma méretű helyet hagy

`space-around`
: Minden tárolt elem mindkét oldalán azonos helyet hagy, a szomszédos üres részeket nem vonja össze

flexbox.html

```
    <p>Egyenletesen elosztva, kitöltés csak az elemek között</p>
    <div style="justify-content: space-between">
      <p>Alma</p>
      <p>Banán</p>
      <p>Citrom</p>
    </div>
    <p>Egyenletesen elosztva, kitöltés az elemek előtt és után is</p>
    <div style="justify-content: space-around">
      <p>Alma</p>
      <p>Banán</p>
      <p>Citrom</p>
    </div>
```

Egyenletesen elosztva, kitöltés csak az elemek között

Egyenletesen elosztva, kitöltés az elemek előtt és után is

Ha az elemek nem férnek el egymás mellett a tengely mentén, akkor a `flex-wrap` tulajdonságtól függ, mi történik:

`nowrap`
: Alapértelmezés, a tárolt elemek kilógnak / nem jelennek meg

`wrap`
: További párhuzamos tengelyeket hoz létre

`wrap-reverse`
: Mint `wrap`, de a tengelyek sorrendje fordított

A `flex-direction` és `flex-wrap` tulajdonságok egy összetett tulajdonságon keresztül is állíthatók:
`flex-flow:` *`flex-direction flex-wrap`*

flexbox.html

```
107     <p>Új tengelyt kezd, ha szükséges</p>
108     <div style="flex-wrap: wrap;">
109       <p>Alma</p>

116       <p>Hónapos retek</p>
117     </div>
```

Új tengelyt kezd, ha szükséges

Alma | Banán | Citrom | Dinnye | Eper | Fokhagyma | Görögdinnye | Hónapos retek

Új tengelyt kezd, ha szükséges

Alma | Banán | Citrom | Dinnye | Eper | Fokhagyma | Görögdinnye | Hónapos retek

Az elemeknek a tengelyre merőleges irányban történő méretezését / elhelyezését befolyásolja az `align-items` tulajdonság:

`stretch`
: Kinyújtás a tároló széléig

`flex-start`
: Az elemeket balra / felfelé igazítja

`flex-end`
: Jobbra / lefelé igazít

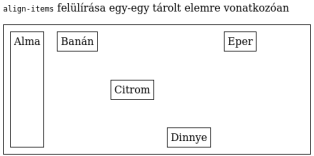
`center`
: Középre igazít

`baseline`
: Az alapvonalhoz igazít

A tárolóra beállított `align-items` minden elem megjelenítését befolyásolja, de ez az elemek `align-self` tulajdonságán keresztül egyedileg felülírható.

flexbox.html

```
99    <p><code>align-items</code> felülírása egy-egy
        ↪ tárolt elemre vonatkozóan</p>
100   <div style="min-height: 200px">
101     <p>Alma</p>
102     <p style="align-self: flex-start">Banán</p>
103     <p style="align-self: center">Citrom</p>
104     <p style="align-self: flex-end">Dinnye</p>
105     <p style="align-self: baseline">Eper</p>
106   </div>
```



Az `align-content` a tengelyek tárolón belüli elhelyezését befolyásolja. Értékei ugyanazok, mint `align-items`-nek. Akkor látványos a hatása, ha több tengelyre van szükség az elemek elhelyezéséhez.

Ha az elemek HTML-ben specifikált sorrendjétől eltérő sorrendben szeretnénk azokat a tengelyre felfűzni, minden elemhez megadhatjuk a kívánt pozíciót az `order` tulajdonság értékeként.

flexbox.html

```
151     <p>Tárolt elemek sorrendjének módosítása</p>
152     <div>
153       <p style="order: 1">Alma</p>
154       <p>Banán</p>
155       <p style="order: -2">Citrom</p>
156     </div>
```

Tárolt elemek sorrendjének módosítása

Citrom | Banán | Alma

Az elemek között arányosan szétosztható a konténer területe, azok `flex` tulajdonságainak beállításával: `flex:` *`flex-grow flex-shrink flex-basis`*

flexbox.html

```
163     <p>Tárolt elemek arányos méretezése</p>
164     <div>
165       <p style="flex: 1">Alma</p>
166       <p style="flex: 2">Banán</p>
167       <p style="flex: 3">Citrom</p>
168     </div>
```

Tárolt elemek arányos méretezése

Alma	Banán	Citrom

Beállítások hatása gyakorlatban kipróbálható: flextest.html

Fejlessze tovább a Wiki által ihletett, CSS-t ismertető weboldalt!

cssFormazott.html, css.css

Cascading Style Sheets

A Wikipédiából, a szabad enciklopédiából

A CSS a számítástechnikában egy stílusleíró nyelv, mely a HTML vagy XHTML típusú strukturált dokumentumok megjelenését írja le. Ezenkívül használható bármilyen XML alapú dokumentum stílusának leírására is, mint például az SVG, XUL stb.

Tartalomjegyzék

1 Áttekintés
2 A CSS használata
2.1 Példa XHTML dokumentumra beágyazott CSS stílussal
2.2 Példa a felhasználói stílusra

Áttekintés

Az elemek stílusát különböző CSS szelektorokkal

1. Induljon ki a korábban létrehozott css.html fájlból! Ha kell, egészítse ki a HTML kódot új (pl. `class`, `id`) attribútumokkal!
2. Kapcsolja ezt össze egy `css.css` nevű stíluslappal, amit készítsen el az alábbi pontoknak megfelelően!
3. A címsorok betűtípusa legyen *Linux Libertine*, ennek hiányában *Georgia*, *Times*, de legalább valamilyen talpas betűkészlet!
4. A bekezdések betűmérete legyen 14 nyomdai pont, a sortávolság legyen 1,5-szeres!
5. Az első szintű címsor betűmérete legyen a bekezdések betűméretének 1,8-szerese, a második szintűké 1,5-szeres, a harmadik szintűeké pedig 1,2-szeres! Ez utóbbit szedje félkövér betűkkel!
6. A hiperhivatkozások csak akkor legyenek aláhúzva, ha éppen felettük van az egér!

Összefoglaló feladatok

cssFormazott.html, css.css

Cascading Style Sheets

A Wikipédiából, a szabad enciklopédiából

A CSS a számítástechnikában egy stílusleíró nyelv, mely a HTML vagy XHTML típusú strukturált dokumentumok megjelenését írja le. Ezenkívül használható bármilyen XML alapú dokumentum stílusának leírására is, mint például az SVG, XUL stb.

Tartalomjegyzék

1 Áttekintés
2 A CSS használata
  2.1 Példa XHTML dokumentumra beágyazott CSS stílussal
  2.2 Példa a felhasználói stílusra

Áttekintés

Az elemek stílusát különböző CSS szelektorokkal

7. A tartalomjegyzéket vegye körül 1 képpont széles, folytonos, szürke színű szegéllyel!
8. A háttérszín is legyen szürke, melynek minden RGB komponense 240 értékű!
9. Maga a tartalomjegyzék ne legyen szélesebb, mint amit a tartalma indokol!
10. A jobb oldalon állítsa 20 képpontra a kitöltést!
11. A „Tartalomjegyzék” felirat legyen középre igazítva!
12. A felsorolások bal oldali kitöltése legyen 20 képpont!
13. A számozás szintenként folytonos legyen, a minta szerint!

cssFormazott.html, css.css

Áttekintés

Az elemek stílusát különböző CSS szelektorokkal lehet kiválasztani:

- *Minden elemre* – a `*` szelektor használatával
- *Az elem neve alapján* – például minden `'p'` vagy `'h2'` HTML-elemhez
- *Leszármazottak alapján* – például az olyan `'a'` elemekre, melyek egy `'li'` elem részei, a szelektor `"li a"`
- *class vagy id attribútumok alapján* – például .class és/vagy `#id` a `class="osztály"` vagy `id="azonosító"` elemekhez

A CSS egyszerű szintaxissal rendelkezik, csak néhány angol nyelvű kulcsszót használ a stílusok tulajdonságaihoz. A stíluslap maga a stílust leíró **szabályok** sora. Minden szabályhoz tartozik egy **szelektor** és egy **deklarációs** szakasz. Ez utóbbi kapcsos zárójelek között pontosvesszővel elválasztott **deklarációkat** tartalmaz. A deklarációk formája a következő: a **tulajdonság** neve, egy kettőspont, majd az adott tulajdonság **értéke**.

Például:

```
p {
  font-family: "Garamond", serif;
}
p#bevezetes {
  border: 3px dashed black;
}
```

A CSS használata

14. A programkódok, akár egy soron belül helyezkednek el, akár előformázott szövegként, kapjanak világosszürke hátteret, melynek RGB összetevői 16-os számrendszerben rendre `f8`, `f9` és `fa`!
15. A szegélyek 1 képpont széles, folytonos vonalak, melyek színének komponensei: `ea`, `ec` és `f0`!
16. A szöveg legyen 14 képpont méretű, félkövér betűkkel szedett!
17. Ha a kód soron belül található, a szegélye legyen 2 képpont sugarú lekerekítéssel ellátva, továbbá fent és lent 1-1, a két oldalon pedig 4-4 képpont belső kitöltéssel rendelkezzen!
18. Ezzel szemben az előformázott szövegeknek minden oldalán hagyjon akkora kitöltést, mint amekkora az alapértelmezett betűméret!

cssFormazott.html, css.css

```
p {
  font-family: "Garamond", serif;
}
p#bevezetes {
  border; 3px dashed black;
}
```

A CSS használata

A CSS használatával a `h2` elem csak strukturális információt hordozhat, míg a stíluslap megadja a prezentációs tulajdonságokat. A fenti kód itt így nézne ki:

```
<h2>A CSS használata</h2>
```

Példa XHTML dokumentumra beágyazott CSS stílussal

Minta...

Példa a felhasználói stílusra

A *cimsorkiemeles.css* fájl tartalma:

```
h1 {color: white; background: orange !important; }
h2 {color: white; background: green !important; }
```

Egy ilyen fájl helyileg tárolódik, és a böngésző beállításaiban lehet aktiválni. Az "!important" azt jelenti, hogy a stílus a szerző specifikációja fölött is érvényesül.

A szintaktikai kiemeléssel ellátott CSS forrásszövegben

19. az elemnevek, tulajdonságok és kulcsszavak színe legyen zöld,
20. az `id` legyen kék,
21. a mértékegység és a karakterlánc típusú adat legyen piros,
22. a mennyiség pedig szürke!

Bevezetés a JavaScript használatába

1. rész

Hatwágner F. Miklós

Széchenyi István Egyetem, Győr

2015. november 11.

A JavaScript fontosabb tulajdonságai

- Web oldalba ágyazható → dinamikus HTML (de: PDF, asztali widget, szerver oldali szolgáltatások is)
- Objektum-orientált, de nincs öröklődés! (objektum alapú, vagy prototípus alapú)
- Gyengén típusos, dinamikus típusosság (változók futásidőben típust válthatnak) → kényelmes, de veszélyes az implicit típuskonverziók miatt
- Böngésző által értelmezett (interpreter) nyelv (script)
- HTML DOM (Document Object Model) elérése
- Eseménykezelés
- Dokumentáció: ECMA, Mozilla, Microsoft, W3Schools, ...

Történelem

- 1995: Brendan Eich, Netscape, eredeti elképzelés: séma értelmező → Mocha, majd LiveScript (Navigator 2.0b) és JavaScript (Navigator 2.0b3), Microsoft: JScript, VBScript
- 1996: Szabványosítás megkezdődik, ECMA (European Computer Manufacturers Association) → ECMAScript
- 1997: ECMAScript v1
- 1998: ECMAScript v2
- 1999: ECMAScript v3, MS: XMLHTTP Active X (AJAX kezdete)
- 2000: Mozilla: XMLHTTPRequest (XHR, AJAX)
- 2002: JSON (JavaScript Object Notation)
- 2004: Firefox v1 megjelenik, Safari: XMLHTTPRequest támogatás
- 2005: Opera: XMLHTTPRequest támogatás
- 2006: MS: XMLHTTPRequest támogatás (IE7+), megjelenik a JQuery, GWT (Google Web Toolkit), Firebug
- 2008: ECMAScript v4-et elvetik
- 2009: ECMAScript v5 megjelenik → 2010: JS 1.8.5, kompatibilitás

Mire használható?

- '90-es évek: felugró ablakok, médialejátszás, objektumok mozgatása, státuszsor tartalmának változtatása, ... → „játéknyelv"
- De jól használható űrlapok ellenőrzéséhez!
- Szerver oldali ellenőrzés is kell: JS kikapcsolható, hackerek
- az AJAX (Asynchronous JavaScript and XML) új értelmet ad a JS-nek
  - Előnyök: a munka nagy részét a böngészők végzik, a lehető legkevesebb adatot kell továbbítani
  - Hátrány: az oldalak *sokkal* bonyolultabbak lesznek
- Probléma: inkompatibilis böngészők
- Ma már nem csak XML lehet az adat: JSON, nyers szöveg
- Példa: regisztráció egy weblapon

Keretrendszerek (framework)

- Előnyök:
  - böngészőfüggetlenség
  - kódismétlés elkerülése
  - új funkciók, pl. dátumválasztó
- Hátrányok:
  - ált. sokkal több kódot kell letölteni, mint amire szükség van
  - emiatt gyakran lassabb is
  - ettől még ugyanúgy meg kell tanulni a JS-et is
- Példák:
  - script.aculo.us, (Yahoo! User Interface), jQuery, MooTools, ExtJS, The Dojo Toolkit, Prototype ...

JavaScript elhelyezése a weboldalon

- script HTML elem segítségével
- type attribútum értéke: text/javascript. HTML5: a *type* attribútum elhagyható
- A JS kód elhelyezhető a nyitó és záró *script* elem között, vagy az src attribútumban (de egyszerre csak az egyikben). Pl. <script src="forras.js"></script>
- A forrás megadható abszolút és relatív úttal is
- Tetszőleges számú *script* elem használható egy oldalon,
- de több fájl letöltése növeli a letöltési időt!
- A kód azonnal lefut, amint a böngésző elér hozzá (kivéve függvények, amiket hívni kell)

JavaScript elhelyezése a weboldalon

- Bárhol elhelyezhető, de célszerű az oldal végére tenni:
  - az oldal megjelenítését nem lassítja a JS letöltése, feldolgozása
  - az oldal betöltése közben a DOM tartalma változik!
- A JS kód a HTML elemek eseménykezelést beállító attribútumában is beállíthatóak, de nem ajánlott, mert
  - a HTML nehezebben áttekinthetővé válik,
  - nehéz lesz karbantartani az oldalakat,
  - amiken sok, ismétlődő kódrészlet szerepel majd.
  - nem vezet szemantikus webhez, mert a JS kód nem az adat/struktúra része
  - `<input type="button" value="Kattints rám!" onClick="alert('Hahó!')">`
- Ajánlott:
  - külső fájl használata,
  - mert áttekinthető, és a böngésző gyorstárába kerül a fájl, így nem kell többször letölteni

JavaScript elhelyezése a weboldalon

- XHTML: a script nem elemezhető, ezért CDATA részbe kell helyezni:

XHTML és JavaScript

```
<script>//<![CDATA[
// Ide kell írni a JavaScript kódot.
//]]></script>
```

Fejlesztési megközelítések

- Feltesszük, hogy mindenhol használható JavaScript
  - rossz ötlet, mert kikapcsolhatja a felhasználó,
  - valamilyen beépülő modul, pl. biztonsági okokból,
  - konzolos eszközökön (pl. `wget`) nincs,
  - de néhány mobil eszközön sincs,
  - szoftver ágensek egy részének (pl. kereső bot) sincs JS értelmezője, stb.
- Elegáns visszalépés (degrade gracefully)
  - feltételezünk bizonyos képességeket, aztán ha ezek nem állnak rendelkezésre, akkor
  - probléma jelzése, pl. <noscript>Nincs JS támogatásod!</noscript>
- Fokozatos fejlődés (progressive enhancement)

Fejlesztési megközelítések

- Fokozatos fejlődés (progressive enhancement)
  - Csak alapfunkciókat biztosítunk, de
  - ha a böngésző képes valamit jobban is megoldani, átváltunk arra.
  - Böngésző típus ellenőrzés vagy funkció ellenőrzés?
- Tartózkodó JavaScript (unobtrusive JS)
  - nem változtatjuk meg a szokásos működést

Változók deklarálása, definiálása

- Gyengén típusos nyelv: az értékadásnál dől el a változó típusa
- Futásidőben bármikor típust válthat
- Deklarálás: var kulcsszóval, vagy anélkül (hatókör!)

Változók deklarálása

```
valt1 = 3;
var valt2 = 4, valt3 = "szöveg";
var valt4;

valt2 = 'béka';
```

Változók deklarálása, definiálása

- Változó neve tartalmazhat:
  - betűket, számokat, hangosközt és néhány egyéb karaktert is, de
  - betűvel, hangosközzel vagy $ jellel kell kezdődnie,
- Nem tartalmazhat pl. szóközt, pontot és még néhány egyéb jelet. Nem használhatóak a foglalt szavak (pl. JavaScript), előredefiniált változók nevei (pl. window)
- A nevek kis- és nagybetűre érzékenyek → konvenció: *camel-case* névadás, mint Java-ban, pl. hallgatoNeve.
- Konstansok: const kulcsszóval hozhatók létre, de nem minden böngésző támogatja! Írásmód: csupa nagybetű, hangosköz szeparátorral, pl. const NEPTUN_HOSSZ = 6;

Típusok

- Egyszerű típusok (literal) létrehozhatók objektumként (object) is, de így lassabb a működés. Bizonyos feladatok implicit objektum létrehozással járnak.
- Egyszerű
  - Number, lebegőpontos számok
  - String, idézett karakter literál
    - 'szöveg', "szöveg"
    - "Guns 'n' Roses", 'ha "idézni" kell'
    - Escape szekvenciák: `\'`, `\"`, `\n`, `\t`, `\r`, `\f`, `\a`, `\b`, `\\`
  - Boolean: true, false
  - null
  - undefined: nem „igazi" típus, deklaráció után, illetve értéket vissza nem adó fv. esetén
- Összetett

Számok

- Aritmetikai operátorok: +, -, *, / (lebegőpontos osztás), % (lebegőpontos maradék képzés)
- Növelés és csökkentés: ++, -- (prefix és postfix formában)
- Hiba esetén: NaN (Not a Number), Infinity
- Létrehozás: var sz = 3; var sz2 = new Number(3.14);
- Gyakran használt metódusok:
  - toFixed() tizedesjegyek száma
  - toPrecision() összes számjegykarakter száma
- Konstansok:
  - MAX_VALUE, MIN_VALUE

Operátorok

Precedencia	Operátor	Megjegyzés
1	. []	tagok elérése
1	new	példányosítás
2	()	fv. hívás
3	++ --	növel, csökkent
4	!	log. nem
4	+ -	előjelek
4	typeof void delete	
5	* / %	aritmetikai op.
6	+ -	összead, kivon
8	< <= > >=	relációk
9	== != === !==	egyezőség
13	&&	log. és
14	||	log. vagy
15	?:	feltételes op.
16	= += -= *= /= %= <<= >>= >>>= &= ^= |=	összetett op.

Számpélda

fahrenheit.html

```
<!DOCTYPE html>
<html lang="hu">
  <head>
    <meta charset="utf-8">
    <title>Fahrenheit - Celsius átváltó</title>
  </head>
  <body>
    <form id="valtoUrlap">
       <fieldset>
          <legend>Adja meg a Fahrenheit fokot!</legend>
          <div><label for="fahrenheit">Fahrenheit</label>
               <input type="text" name="fahrenheit" id="fahrenheit"
               value="72" pattern="[-+]?[0-9]*\.?[0-9]+"
               required="required" title="lebegőpontos szám" /></div>
          <div><label for="celsius">Celsius</label>
               <input type="output" name="celsius" id="celsius"
               title="számolás eredménye" /></div>
          <div><input type="submit" value="Átvált" id="submit" /></div>
        </fieldset>
    </form>
    <script src="atvalto.js"></script>
  </body>
</html>
```

Számpélda

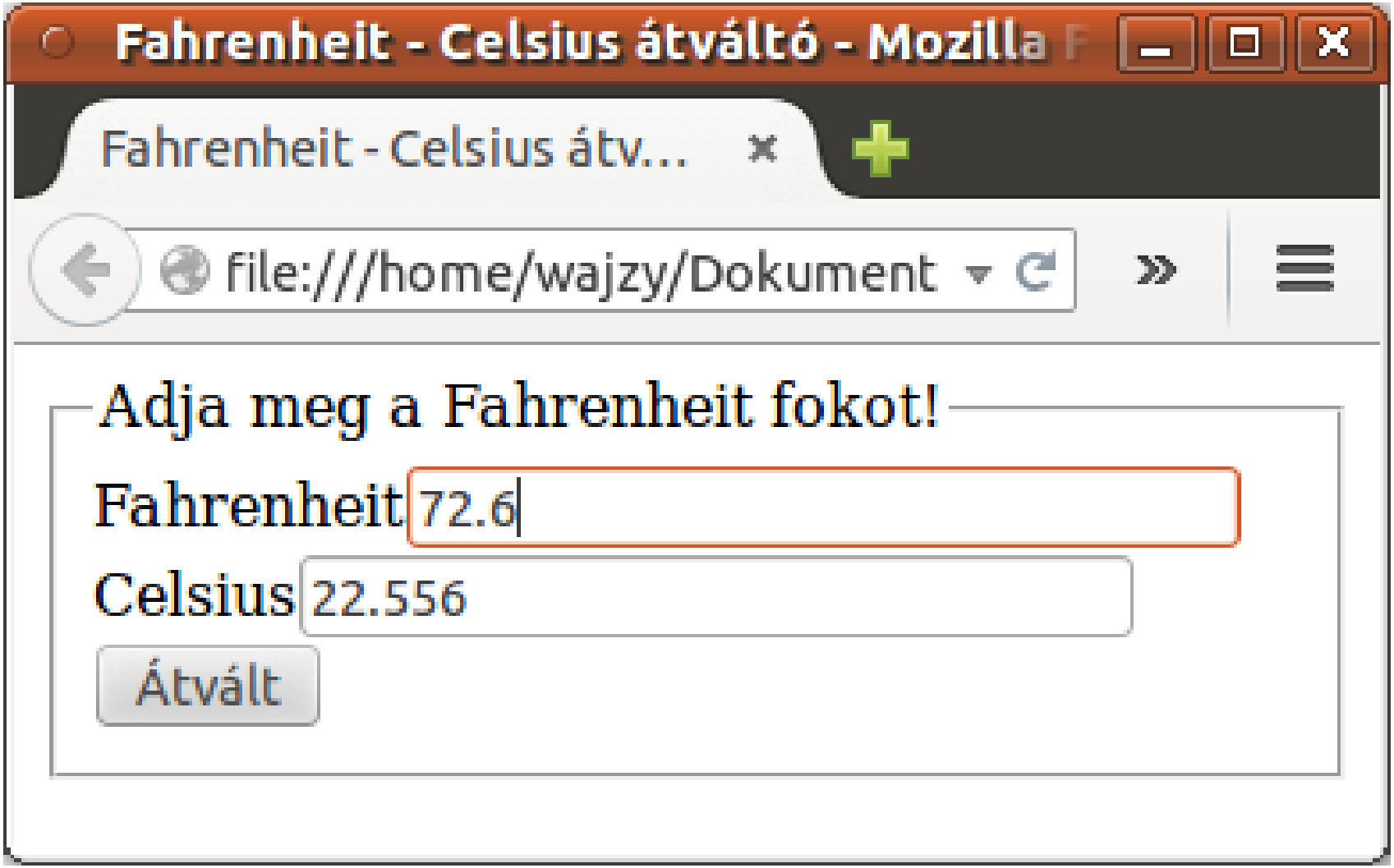
atvalto.js

```
function atvalt() {
  'use strict';
  var cels;
  var fahr = document.getElementById("fahrenheit").value;
  cels = 5/9*(fahr-32);
  document.getElementById("celsius").value = cels.toFixed(3);
  return false;
}

function init() {
  'use strict';
  var urlap = document.getElementById("valtoUrlap");
  urlap.onsubmit = atvalt;
}

window.onload = init;
```

Számpélda


Fahrenheit - Celsius átváltó - Mozilla F
Fahrenheit - Celsius átv...
file:///home/wajzy/Dokument
Adja meg a Fahrenheit fokot!
Fahrenheit
72.6
Celsius
22.556
Átvált

Math objektum

Konstansok:

- PI, 3.1415...
- E, 2.71...

Metódusok:

- abs(), abszolút értéket adja vissza
- ceil(), felfelé kerekít
- floor(), lefelé kerekít
- min(), a tetszőleges számú paraméter közül a legkisebbet szolgáltatja
- max(), mint *min()*, csak a legnagyobbat adja
- pow(), hatványozás (alap, kitevő)
- round(), kerekítés a legközelebbi egészre
- random(), véletlen szám a [0;1) intervallumból
- sin(), cos(), tan(), trigonometrikus fv.-ek, argumentum radiánban
- asin(), acos(), atan(), az előzőek inverzei
- exp(), log(), exponenciális fv., természetes alapú logaritmus
- sqrt(), négyzetgyök vonása

String objektum

Immutable object, mint Java-ban

Létrehozás:

- var str = new String("egy");
- var str2 = "kettő";
- var str3 = 'három';

Tulajdonság (nyilvános konstans adattag):

- length, a karakterlánc hossza

Metódusok:

- charAt(), adott indexű karaktert ad vissza, pl.
  "JavaScript".charAt(0); // "J"

String objektum

Metódusok:

- indexOf(), rész-karakterlánc keresése, pl.
  ```
  "JavaScript".indexOf("a"); // 1
  "JavaScript".indexOf("Script"); // 4
  "JavaScript".indexOf("a", 2); // 3
  "JavaScript".indexOf("A"); // -1
  ```
- lastIndexOf(), rész-karakterlánc keresése visszafelé, pl.
  ```
  "JavaScript".lastIndexOf("a"); // 3
  ```
- slice(), rész-karakterlánc visszaadása, pl.
  ```
  "JavaScript".slice(4); // "Script"
  "JavaScript".slice(0, 4); // "Java"
  "JavaScript".slice(0, -6); // "Java"
  "JavaScript".slice(-6); // "Script"
  ```
- substring(), mint *slice()*, kis eltérésekkel
- substr(), mint *slice()*, de itt a karakterek darabszámát várja, nem a záró elem indexét, pl.
  ```
  "JavaScript".substr(4,6); // "Script"
  ```

String objektum

Metódusok:

- concat(), összefűzés, pl.
  "Java".concat("Script", " 1.8.5"); // "JavaScript 1.8.5"
  var str = "Java" + "Script"; // "JavaScript"
  str += " 1.8.5"; // "JavaScript 1.8.5"
- toLowerCase(), kisbetűsre alakítás, pl.
  "JavaScript".toLowerCase(); // "javascript"
- toUpperCase(), nagybetűsre alakítás, pl.
  "JavaScript".toUpperCase(); // "JAVASCRIPT"
- trim(), kezdő és záró fehér karakterek törlése, JS 1.8.1+, pl.
  " a ".trim(); // "a"
- split(), elválasztó jel mentén tördelés, tömb készítés, pl.
  "a b c".split(" "); // ["a", "b", "c"]

Típusok közötti átalakítások

Probléma: gyenge típusosság és a + operátor kettős működése

- 12.3 + 4 // 16.3
- "12.3" + 4 // 12.34
- parseFloat("12.3") + 4 // 16.3
- parseInt("12.3") + 4 // 16
- parseInt("12szamár") // 12
- parseInt("szamár12") // NaN
- +"12.3" + 4 // 16.3
- parseInt("0xFF", 16) // 255
- parseInt("FF", 16) // 255
- var sz = 5; sz.toString() // "5"
- new Number(5).toString() // "5"

Feltételek, *if*

- if(*feltétel*) *utasítás*;
- if(*feltétel*) {
  // utasítások
  }
- if(*feltétel*) {
  // igaz ág utasításai
  } else {
  // hamis ág utasításai
  }

- Mikor teljesül a feltétel, és mikor nem?
  - false
  - 0
  - ""
  - NaN
  - null
  - undefined

Operátorok

Feltételes operátorok

- < kisebb
- <= kisebb vagy egyenlő
- > nagyobb
- >= nagyobb vagy egyenlő
- == egyenlő
- != nem egyenlő
- === értéke és típusa is azonos
- !== értéke vagy típusa nem azonos

Logikai operátorok

- && és
- || vagy
- ! nem

Feltételek, *switch*

```
switch(kifejezés) {
    case érték1:
        // utasítások
        break;
    case érték2:
    case érték3:
        // utasítások
        break;
    default:
        // utasítások
        break;
}
```

Az értéknek *és* a típusnak is egyeznie kell!
A *default* ág elhagyható.

Feltételek, három operanduszú operátor

(*kifejezés*) ? *igaz_ág* : *hamis_ág*;

var megisszuk = (ital=="sör") true : false;

Összehasonlítási nehézségek

Lebegőpontos ábrázolás miatt néha az értékek nem egyeznek. Lehetséges megoldások:

- *toFixed()*
- mindig egész számok használatára törekedni, pl. áfa 1.27 helyett 127

De if (NaN === NaN) sosem teljesül! Megoldás:

- isNaN()
- isFinite()

typeof

```
if(typeof valtozo == "string") ...
```

Típus	Eredmény
Undefined	undefined
Null	object
Boolean	boolean
Number	number
String	string
tömb	object
objektum	object

Ciklusok

```
for(előkészítő_kifejezések; ismétlési_feltétel;
    ciklusmag_utáni_kifejezések) {
    // Ciklusmag utasításai
}

while(ismétlési_feltétel) {
    // Ciklusmag utasításai
}

do {
    // Ciklusmag utasításai
} while (ismétlési_feltétel);

break, continue
```

Date objektum

Példány létrehozása:

- new Date(): a kliens órájának állását veszi fel
- new Date(*ezredmp*): a Unix-időszámítás kezdetétől számított ennyi ezredmásodperces időbélyeget reprezentálja
- new Date(*dátumString*): karakterlánccként adott időbélyeg ábrázolása, formátum
- new Date(*év*, *hónap*, *nap* [, *óra*, *perc*, *másodperc*, *ezredmp*]): külön-külön megadott óra, perc, ... értékek megadása. Az utolsó négy közül tetszőleges számú elhagyható, de csak hátulról előre, összefüggő egységként

Figyelem!

- hónap [0;11]
- nap [1;...]
- óra [0;23]

Date objektum

Metódusok

- get / setTime(), Date.now()*: időbélyeg
- get / setFullYear(): év
- get / setMonth(): hónap, január = 0
- get / setDate(): hónapon belüli nap
- getDay(): a hét napja, vasárnap = 0
- get / setHours(): óra
- get / setMinutes(): perc
- get / setSeconds(): másodperc
- get / setMilliseconds(): ezredmp.

*ECMAScript 5

Date objektum

Exportálás különleges formátumokra:

- toDateString() // Wed Oct 24 2012
- toISOString()* // 2012-10-24T15:53:35.970Z
- toJSON()* // 2012-10-24T15:53:35.970Z
- toLocaleDateString() // 2012-10-24
- toLocaleString() // 2012. okt. 24., szerda, 17.53.35 CEST
- toLocaleTimeString() // 17.53.35
- toString() // Wed Oct 24 2012 17:53:35 GMT+0200 (CEST)
- toTimeString() // 17:53:35 GMT+0200 (CEST)
- toUTCString() // Wed, 24 Oct 2012 15:53:35 GMT (ld. getTimezoneOffset())

*ECMAScript 5

Ellenőrzés: if(datum.toJSON) ...

Tömbök létrehozása

Objektumként:

- var tomb = new Array();
- var tomb = new Array(1, 2, 3);
- var tomb = new Array(3.14, "Hahó", false);

Literál:

- var tomb = [];
- var tomb = [1, 2, 3];
- var tomb = [3.14, "Hahó", false];

Tömbök használata

```
var tomb = new Array(1, 2, 3); // [1, 2, 3]
console.log(tomb.length); // 3; A tömb 3 elemű(nek tűnik)
console.log(tomb[1]); // 2; Az 1 indexű elem értéke 2
tomb[3] = 4;
console.log("Kiegészített tömb: " + tomb);
    // Kiegészített tömb: 1,2,3,4
tomb[9] = 10;
console.log("Definiálatlan elemek: " + tomb);
    // Definiálatlan elemek: 1,2,3,4,,,,,,10
console.log(tomb.length); // 10
console.log(tomb[8]); // undefined
tomb[] = 11; // SyntaxError: syntax error

tomb = new Array(3);
console.log(tomb);
    // [undefined, undefined, undefined]
```

Tömbök használata

Hiányos tömbök

```
var ht = [1, , 3]; // [1, undefined, 3]
ht = [1, , 3, ]; // [1, undefined, 3]
ht = [1, undefined, 3, undefined]; // [1, undefined, 3, undefined]
```

Keresés (ECMAScript 5)

```
var sor = ["Kozel", "HB", "Dreher"];
sor.indexOf("HB"); // 1
sor.indexOf("Soproni"); // -1
sor[sor.length] = "HB";
sor.lastIndexOf("HB"); // 3
// Itt is használható lenne az opcionális második paraméter
```

Tömbök használata

Törlés

```
var sor = ["Kozel", "HB", "Dreher", "Kőbányai"];
delete sor[3];
console.log(sor); // ["Kozel", "HB", "Dreher", undefined]
```

Hozzáadás, elvétel...

```
var csajok = ["Rozi", "Marika", "Nárcisz"];
csajok.push("Melinda");
    // 4; ["Rozi", "Marika", "Nárcisz", "Melinda"]
csajok.pop(); // "Melinda"; ["Rozi", "Marika", "Nárcisz"]
csajok.unshift("Melinda");
    // 4; ["Melinda", "Rozi", "Marika", "Nárcisz"]
csajok.shift(); // "Melinda"; ["Rozi", "Marika", "Nárcisz"]
```

Tömbök használata

Többdimenziós tömbök

```
var emberek = [["Rozi", "Marika", "Nárcisz"],
    ["Arnold", "Dávid", "Laci", "Szilárd"]];
console.log(emberek.length); // 2
console.log(emberek[0].length); // 3
console.log(emberek[1].length); // 4
console.log(emberek[1][2]); // "Laci"
```

Új elemek hozzáfűzése

```
emberek.concat(["Zoli", "Tihamér"]);
    // [["Rozi", "Marika", "Nárcisz"],
    // ["Arnold", "Dávid", "Laci", "Szilárd"],
    // "Zoli", "Tihamér"]
```

Tömbök használata

Kivág, beilleszt...

```
var csajok = ["Rozi", "Marika", "Nárcisz"];
console.log(csajok.splice(1, 2)); // ["Marika", "Nárcisz"]
console.log(csajok); // ["Rozi"]
csajok.splice(1, 0, "Marika", "Nárcisz");
    // [ ]; ["Rozi", "Marika", "Nárcisz"]
console.log(csajok.splice(-1, 1)); // ["Nárcisz"]
```

Tehát a a paraméterezés: index, darabszám, új elemek
Módosítja az eredeti tömböt.

Tömbök használata

Kivág, beilleszt...

```
var csajok = ["Rozi", "Marika", "Nárcisz"];
console.log(csajok.slice(0, 2)); // ["Rozi", "Marika"]
console.log(csajok); // ["Rozi", "Marika", "Nárcisz"]
console.log(csajok.slice(1)); // ["Marika", "Nárcisz"]
console.log(csajok.slice(-1)); // ["Nárcisz"]
console.log(csajok.slice(-3, -1)); // ["Rozi", "Marika"]
```

Tehát a a paraméterezés: kezdőIndex, végIndex+1
Nem módosítja az eredeti tömböt.

Tömbök használata

Egyesítés...

```
var csajok = ["Rozi", "Marika", "Nárcisz"];
var lista = "<ul><li>";
lista += csajok.join("</li><li>");
lista += "</li></ul>";
```

...feldarabolás

```
var csajok = "Rozi,Marika,Nárcisz";
var cst = csajok.split(",");
console.log(cst); // ["Rozi", "Marika", "Nárcisz"]
```

Objektumok

Néhány megjegyzés:

- JS-ben nincs öröklés, az objektumok prototípusok alapján készülnek
- megkönnyíti az objektumok létrehozását, de néhány OO technika nehezen kivitelezhető
- az objektumok adattagokat és metódusokat zárnak egybe
- de ezek sorrendjét nem őrzi meg

Létrehozás módjai:

- new operátorral: var obj = new Object();
- literál szintakszissal: var obj = {}; (ajánlott)

Adattagok felvétele

Szintaktika:

- Új adattag: adattag_neve : érték
- Adattagokat , választja el egymástól
- Új adattag utólag is felvehető: auto.szin = "fehér";
- és törölhető: delete auto.szin;

Minta objektum

```
var auto = {
  gyarto: "Hindustan",
  tipus: "Ambassador",
  evjarat: 1958,
  gyartasban: true,
  felszereltseg:
    ["tekerhető kormány", "felhúzható ablak"],
  tulajdonos: {
    vezeteknev: "Gipsz",
    keresztnev: "Jakab"
  }
};
```

Adattagok elérése

Az objektum nevét az adattag nevétől a . operátor választja el.

```
console.log(auto.evjarat) // 1958
console.log(auto.tulajdonos.vezeteknev) // Gipsz
console.log(auto.felszereltseg[1]) // felhúzható ablak
```

Adattag létezésének ellenőrzése

- if(auto.szin) { ... // Nem biztonságos, pl. "" → false
- if("szin" in auto) { ... // true

Kapcsolat a(z asszociatív) tömbök és az objektumok között

- auto.tulajdonos.keresztnev
- auto["tulajdonos"]["keresztnev"]

Utóbbi akkor is használható, ha egy változó tartalmazza az adattag nevét.

Adattagok elérése

A for/in ciklussal minden adattag/metódus elérhető.

Objektum kiírása

```
for(var t in auto) {
  console.log(t + ": " + auto[t]);
}
```

Kimenet

```
gyarto: Hindustan
tipus: Ambassador
evjarat: 1958
gyartasban: true
felszereltseg: tekerhető kormány,felhúzható ablak
tulajdonos: [object Object]
szin: fehér
```

Függvények készítése

Általános tudnivalók:

- A fv. nevének képzése a szokásos szabályok szerint történik (nem kezdődhet számjeggyel, de tartalmazhat alfanumerikus karaktereket, hangosközt, $ jelet, stb., nem használhatóak a foglalt szavak, ...)
- Tetszőleges számú paraméter fogadására felkészíthető
- Nem kötelező értékkel visszatérnie

Egy függvény váza

```
function függvény_neve(par1, par2, ...) {
  // Függvény törzse
  return; // elhagyható --> undefined
}
```

Hívás

```
függvény_neve(par1, par2, ...); // Önálló fv. hívása
objektum.metódus(par1, par2, ...); // Obj. met. hív.
```

Függvények paraméterezése

Fontosabb tudnivalók:

- Gyenge típusosság → csak változónevek szerepelnek, típust nem adjuk meg → nincs típusellenőrzés!
- Ha erre szükség van, nekünk kell megoldani!

Négyzetre emelő fv.

```
function negyzet(alap) {
  return alap * alap;
}
console.log(negyzet(3)); // 9
console.log(negyzet("Malacka")); // NaN
```

Négyzetre emelő fv., típusellenőrzéssel

```
function negyzet(alap) {
  if(typeof alap == "number") {
    return alap * alap;
  }
}
console.log(negyzet(3)); // 9
console.log(negyzet("Malacka")); // undefined
```

Függvények paraméterezése

Fontosabb tudnivalók, folyt.:

- A fv.-t tetszőleges számú paraméterrel lehet hívni (formális paraméterek száma nem számít!)
- Kevesebb aktuális paraméter: undefined
- Több: csak az arguments tömb-szerű objektummal érhetőek el
- Átadás módja: objektumok (tömbök) referencia szerint, minden más típus érték szerint

Tetszőleges mennyiségű szám összeadása

```
function osszeg() {
  var osszeg = 0;
  for(var i=arguments.length-1; i>=0; i--) {
    if(typeof arguments[i] == "number") {
      osszeg += arguments[i];
    }
  }
  return osszeg;
}
console.log(osszeg()); // 0
console.log(osszeg(1, 2, 3)); // 6
```

Függvények paraméterezése

Eltérő számú aktuális paraméter

```
function teszt(par1, par2) {
  console.log("par1: " + par1 + ", par2: " + par2);
}
teszt(); // par1: undefined, par2: undefined
teszt("a"); // par1: a, par2: undefined
teszt("a", "b"); // par1: a, par2: b
teszt("a", "b", "c"); // par1: a, par2: b
teszt(undefined, "b"); // par1: undefined, par2: b
```

A paramétereknek nincs alapértelmezett értéke → nekünk kell gondoskodni róla!

Alapértelmezett érték

```
function teszt(par1, par2) {
  if(typeof par1 == "undefined") {
    par1 = "n/a";
  }
  console.log("par1: " + par1 + ", par2: " + par2);
}
teszt(undefined, "b"); // par1: n/a, par2: b
```

Függvények paraméterezése

Érték szerinti átadás

```
function novel(sz) {
  sz++;
  console.log("sz: " + sz);
}
var szam = 1;
console.log("szam: " + szam); // szam: 1
novel(szam); // sz: 2
console.log("szam: " + szam); // szam: 1
```

Referencia szerinti átadás

```
function novel2(sz) {
  sz.adat++;
  console.log("sz.adat: " + sz.adat);
}
var objektum = {
    adat: 1
};
console.log("objektum.adat: " + objektum.adat); // objektum.adat: 1
novel2(objektum); // sz.adat: 2
console.log("objektum.adat: " + objektum.adat); // objektum.adat: 2
```

Változók hatóköre

Lokális

```
function fv() {
  console.log("fv() -> szam: " + szam);
  var szam = 2; // A "var" lokális változót vezet be!!!
  console.log("fv() -> szam: " + szam);
}

var szam = 1;
console.log("szam: " + szam);
fv();
console.log("szam: " + szam);
```

Kimenet

```
szam: 1
fv() -> szam: undefined
fv() -> szam: 2
szam: 1
```

Változók hatóköre

Globális

```
function fv() {
  console.log("fv() -> szam: " + szam);
  szam = 2; // A globális változó értékét módosítjuk!!!
  console.log("fv() -> szam: " + szam);
}

var szam = 1;
console.log("szam: " + szam);
fv();
console.log("szam: " + szam);
```

Kimenet

```
szam: 1
fv() -> szam: 1
fv() -> szam: 2
szam: 2
```

A függvények objektumok

Ez teszi lehetővé pl. annak lekérdezését is, hogy valamilyen szolgáltatás (objektum, metódus, függvény, stb.) támogatott-e az éppen használt böngészőben.

Függvények létezésének ellenőrzése

```
function min(a, b) {
  return (a<b) ? a : b;
}
console.log(typeof min); // function
console.log(min(1,2)); // 1
console.log(typeof max); // undefined
console.log(typeof Date.now); // function
```

Függvény definíció másképpen

```
var min = function (a, b) {
  return (a<b) ? a : b;
}
console.log(min(1,2)); // 1
```

A függvények objektumok

Emlékezzünk vissza az első JS mintaprogramra!

Eseménykezelő módosítása

```
function init() {
  'use strict';
  var urlap = document.getElementById("valtoUrlap");
  urlap.onsubmit = atvalt;
}
window.onload = init;
```

Ugyanez névtelen (anonymous) függvénnyel

```
window.onload = function () {
  'use strict';
  var urlap = document.getElementById("valtoUrlap");
  urlap.onsubmit = atvalt;
}
```

A függvények objektumok

Névtelen függvényeket szoktunk használni, ha:

- a fv.-t hozzárendeljük egy változóhoz,
- objektum adattagjához, azaz metódust készítünk (ld. később),
- átadjuk egy másik függvénynek paraméterként (ld. később),
- vagy azonnal meg is akarjuk hívni

Azonnal meghívásra kerülő fv.

```
(function() {
  // Függvény törzse
})();
```

A függvények egymásba is ágyazhatóak.

Egymásba ágyazott függvények

```
function kulso() {
  console.log("kívül");
  function belso() {
    console.log("belül");
  }
  belso();
  console.log("kívül");
}
kulso();
```

Kimenet

```
kívül
belül
kívül
```

A függvények objektumok

Függvény paramétere is lehet függvény. Pl. az Array.sort() fogadhat rendezési relációt meghatározó függvényt.

Rendezés növekvő és csökkenő sorrendbe

```
var csajok = ["Rozi", "Marika", "Melinda", "Nárcisz"];
console.log(csajok.sort());
  // ["Marika","Melinda","Nárcisz","Rozi"]
function forditva(a, b) {
  if(a < b) {
    return 1;
  } else if(a == b) {
    return 0;
  } else {
    return -1;
  }
}
console.log(csajok.sort(forditva));
  // ["Rozi","Nárcisz","Melinda","Marika"]
```

A függvények objektumok

Tömbök függvény paramétert fogadó függvényei az ECMAScript 5-ben

- **forEach()** Minden elemet egyesével átad a függvénynek.
- **every()** Ha a tesztelő fv. minden elemre igazzal tér vissza, akkor ez is igazat ad.
- **some()** Ha a tesztelő fv. legalább egy elemre igazzal tért vissza, akkor ez is igazat ad.
- **filter()** Új tömböt készít és ad vissza, amelyben azok az elemek szerepelnek, amelyekre a tesztfüggvény igazat adott.
- **map()** Új tömböt készít és ad vissza, amelynek elemei az eredetinek a paraméter függvénnyel történő átalakításával jöttek létre.
- **reduce()** Balról jobbra összevonja az elemeket egy egyszerű változóba.
- **reduceRight()** Jobbról balra összevonja az elemeket egy egyszerű változóba.

A függvények objektumok

Kis értékű elemek kiszűrése

```
console.log(
  [10, 20, 30].filter(
    function (elem) {
      return elem > 15;
    }
  )
); // [20,30]
```

Metódusok

Metódus: objektum függvénye. Adattagok és metódusok a this segítségével érhetőek csak el!

Téglatest felszínének és térfogatának számítása

```
var teglatest = {
  szel: 2,
  hossz: 3,
  mag: 5,
  felszin: function() {
    return (this.szel*this.hossz +
      this.hossz*this.mag +
      this.mag*this.szel) * 2;
  },
  terfogat: function() {
    return this.szel * this.mag * this.hossz;
  }
};
console.log("Felszín: " + teglatest.felszin());
  // Felszín: 62
console.log("Térfogat: " + teglatest.terfogat());
  // Térfogat: 30
```

Bevezetés a JavaScript használatába

2. rész

Hatwágner F. Miklós

Széchenyi István Egyetem, Győr

2015. november 24.

Eseménykezelés

Az eseménykezelés megvalósításának lehetséges módjai:

- soron belüli eseménykezelővel
- klasszikus módszerrel
- W3C eseménykezeléssel
- IE módszerével

Soron belüli eseménykezelés (DOM Level 0)

HTML elemen belül megadott eseménykezelő

```
<a href="http://uni.sze.hu"
    onClick="return confirm('Biztos?')">Egyetem</a>
```

Előny:

- könnyen, gyorsan elkészíthető

Hátrányok:

- a JS kód szétszórva helyezkedik el az oldalon → ez nehezíti a hibakeresést és a továbbfejlesztést
- nem valósítható meg vele a *fokozatos fejlődés* elve
- a JS mindig, mindenhol használható lesz?

Használata nem ajánlott, de régi dokumentumokban sok helyen fellelhető ez a módszer.

Eseménykezelés klasszikus módszerrel (DOM Level 0)

Klasszikus megadás – Fahrenheit-Celsius átváltó (részletek)

```
var urlap = document.getElementById("valtoUrlap");
urlap.onsubmit = atvalt;
...
window.onload = init;
```

További lehetőségek

```
if(typeof window.onload == "function") {
  window.onload = function() {
    // Névtelen függvénnyel is megadható
  }
}
...
window.onload = null; // Függvény eltávolítása
```

Eseménykezelés klasszikus módszerrel (DOM Level 0)

Előnyök:

- egyszerűen, könnyen megvalósítható
- gyakorlatilag minden böngészőben működik

Hátrányok:

- új eseménykezelő hozzáadása felülírja a régit

Működik, de nem ajánlott!

```
window.init = function() {
  első_eseménykezelő();
  második_eseménykezelő();
}
```

W3C eseménykezelési módszer (DOM Level 2)

Előnyök:

- eseménykezelő fv. hozzárendelése JS kódból történik (vö. soron belüli megadás)
- több eseménykezelő is hozzárendelhető ugyanazon eseményhez (vö. klasszikus módszer)
- rugalmas, logikus, viszonylag egyszerű

Hátrányok:

- kicsit terjengősebb a kód
- IE9 előtti változatok nem támogatják

Használat

```
objektum.addEventListener(
    esemény_típusa, kezelő_függvény, fázis);
objektum.removeEventListener(
    esemény_típusa, kezelő_függvény, fázis);
```

W3C eseménykezelési módszer (DOM Level 2)

Minták

```
window.addEventListener(
  "load", első_eseménykezelő, false);
window.addEventListener(
  "load", második_eseménykezelő, false);
...
window.removeEventListener(
  "load", első_eseménykezelő, false);
```

Megjegyzések:

- a *fázis* értéke általában false (rákerüljön a vezérlés az eseménykezelőre az elfogási fázisban? Ld. később)
- csak pontosan egyező paraméterezés mellett lesz sikeres az eltávolítás
- nem létező eseménykezelő eltávolítása nem okoz hibát

IE eseménykezelés

Problémák:

- IE9 előtti verziók nem támogatják a W3C módszerét
- de hasonló működésű függvények léteznek: attachEvent(), detachEvent().
- az esemény neve itt onnal kezdődik (pl. *onload* a *load* helyett)

Létre kell hoznunk egy általános eseménykezelőt beállító függvényt/metódust!

Eseménykezelő hordozható beállítása

```
function addEsemeny(obj, tipus, fv) {
  if (obj && obj.addEventListener) { // W3C
    obj.addEventListener(tipus, fv, false);
  } else if (obj && obj.attachEvent) { // <IE9
    obj.attachEvent("on" + tipus, fv);
  }
}
```

Eseménytípusok

A továbbiakban csak a leggyakrabban előforduló eseményekkel foglalkozunk. További részletek:Mozilla, W3Schools, ...

Események típusai forrásuk szerint származhatnak:

- grafikus kurzort vezérlő eszköztől
- billentyűzettől
- böngésző programtól
- űrlapoktól

Nem minden böngésző támogatja valamennyi eseményt.
Kompatibilitási táblázat

Grafikus kurzort vezérlő eszköztől származó események

Forrása lehet: egér, touchpad, trackball, tablet érintésérzékeny kijelzője, stb.

A(z egér) gombokkal kiváltható események:

- **mousedown** Gombot lenyomták
- **mouseup** Gombot felengedték
- **click** Kattintottak, azaz a gombot lenyomták majd felengedték, és közben nem mozgatták át más elem fölé a kurzort; a *click* előtt a fenti két esemény is létrejön
- **drag** Húzd és ejtsd; a HTML5 előtt teljesen kézzel kellett megvalósítani
- **dblclick** Dupla kattintás; előzőleg két *click* is keletkezik, emiatt a *click* eseményre általában nem regisztrálnak
- **contextmenu** Környezeti/helyi menü előhívása

Grafikus kurzort vezérlő eszköztől származó események

A(z egér) mozgatásával kiváltható események:

mouseover	Az egér valamilyen elem fölé került
mouseout	Az egér elhagyta az elem területét
mousemove	Az egeret mozgatják (erőforrásigényes)

Grafikus kurzort vezérlő eszköztől származó események

utilities.js

```
// http://www.larryullman.com/downloads/modern_javascript_scripts.zip
// Script 8.1 - utilities.js
// This script defines an object that has some utilitarian functions.
var U = {
    // For getting the document element by ID:
    $: function(id) {
        'use strict';
        if (typeof id == 'string') {
            return document.getElementById(id);
        }
    }, // End of $() function.
    // Function for setting the text of an element:
    setText: function(id, message) {
        'use strict';
        if ( (typeof id == 'string')
        && (typeof message == 'string') ) {
            // Get a reference to the element:
            var output = this.$(id);
            if (!output) return false;
            // Set the text
            if (output.textContent !== undefined) {
                output.textContent = message;
            } else {
                output.innerText = message;
            }
            return true;
        } // End of main IF.
    }, // End of setText() function.
```

Grafikus kurzort vezérlő eszköztől származó események

utilities.js (folytatás)

```
    // Function for creating event listeners:
    addEvent: function(obj, type, fn) {
        'use strict';
        if (obj && obj.addEventListener) { // W3C
            obj.addEventListener(type, fn, false);
        } else if (obj && obj.attachEvent) { // Older IE
            obj.attachEvent('on' + type, fn);
        }
    }, // End of addEvent() function.
    // Function for removing event listeners:
    removeEvent: function(obj, type, fn) {
        'use strict';
        if (obj && obj.removeEventListener) { // W3C
            obj.removeEventListener(type, fn, false);
        } else if (obj && obj.detachEvent) { // Older IE
            obj.detachEvent('on' + type, fn);
        }
    } // End of removeEvent() function.
}; // End of U declaration.
```

Grafikus kurzort vezérlő eszköztől származó események

eger.html

```
<!DOCTYPE html>
<html lang="hu">
  <head>
    <meta charset="utf-8">
    <title>Egéresemények tesztelése</title>
  </head>
  <body>
    <div id="uzenet">Mozgasd ide az egeret!</div>
    <script src="utilities.js"></script>
    <script src="eger.js"></script>
  </body>
</html>
```

eger.js

```
function folotte() {
  'use strict';
  U.setText("uzenet", "Viszed innen!");
}
function mashol() {
  'use strict';
  U.setText("uzenet", "Mozgasd ide az egeret!");
}
window.onload = function() {
  'use strict';
  U.addEvent(U.$("uzenet"), "mouseover", folotte);
  U.addEvent(U.$("uzenet"), "mouseout", mashol);
};
```

Billentyűzettől és böngészőtől származó események

Billentyűzettől származó események:

- **keydown** Billentyű lenyomása
- **keyup** Billentyű felengedése
- **keypress** Egy karakter begépelése

Váltóbillentyűk lenyomását (Shift, Alt, ...) általában nem jelzik eseménnyel a böngészők.

Böngésző események:

- **load, unload** Valamilyen erőforrás betöltését/megsemmisítését jelzik
- **resize** Ablak átméretezése
- **scroll** Görgetés
- **copy, cut, paste** Vágólap kezelés, kevés böngésző támogatja
- **focus, blur** Fókusz megszerzése, elvesztése

Űrlaptól származó események

reset	Az űrlapot alaphelyzetbe állítják; ritkán használt
change	Megváltozik az űrlapmező tartalma; szövegbeviteli mezőknél csak az elem elhagyásakor keletkezik, jelölőknél, rádiógomboknál, legördülő menüknél azonnal
select	Szöveges bevitelre alkalmas vezérlőknél a szöveg kijelölését jelzi
click	Jelölőnégyzetek, rádiógombok ezt is kezelik
focus, blur	Beviteli mező megkapja/elveszti a beviteli fókuszt

Megjegyzés:
Fel kell készülni különféle eszközök támogatására. Pl. nincs minden gépen egér, ezért a *mouseover* mellett a *focus* eseményre is ugyanazokat a tevékenységeket kell elvégezni. Hasonlóan: *mouseout/blur*, űrlapoknál a gomb *click*-je helyett/mellett inkább a *submit*-et szerencsésebb kezelni (események párosítása).

Eseményobjektum fogadása

Sokszor szükséges lenne az esemény körülményeinek ismerete is (pl. mire kattintottak?). Ha az `addEventListener()` támogatott, akkor az eseménykezelő első és egyetlen paramétere az esemény objektum lesz, amit automatikusan megkap. Sajnos az IE8-ig csak az `attachEvent()`, `detachEvent()` használható, és az eseményt a `window.event` objektum jellemzi.

Megoldások hordozható eseménykezelő készítéséhez

```
function eseményKezelő(e) {
  // Első megoldás
  if (typeof e == "undefined") e = window.event;
  // Második megoldás
  e = e || window.event;
  // Harmadik megoldás
  if (!e) e = window.event;
  // Tényleges eseménykezelés, e használatba vétele
}
```

Probléma egyedül a soron belüli eseménykezelésnél léphet fel (nekünk kell paramétereznünk), de ilyet nem használunk, ugye?

Eseményobjektum tulajdonságai

Az eseményobjektumból kiolvashatóak az esemény létrejöttének körülményei, de sajnos az objektum felépítése böngészőfüggő.

target/srcElement (W3C/<IE9) az eseményt létrehozó elem referenciája (`forrás.nodeName` megadja az elem nevét, pl. DIV, INPUT)

type az esemény típusa (pl. click, keydown)

Ha billentyű leütése váltotta ki az eseményt:

keyCode a billentyűt azonosító UNICODE kód, IE: ha `keypress` esemény keletkezett, akkor ebben a karakter UNICODE kódját tárolja

which a begépelt karakter kódja, IE: nem támogatott

Kódból karaktert előállítani a `String.fromCharCode()` metódussal lehet.

Karakterkód hordozható kinyerése keypress esetén

```
var karakterKód = e.which || e.keyCode;
```

Eseményobjektum tulajdonságai

Ha billentyű leütése váltotta ki az eseményt:

- **shiftKey** Logikai érték, az esemény közben a *Shift*-et nyomva tartották-e
- **ctrlKey** Logikai érték, az esemény közben a *Ctrl*-t nyomva tartották-e
- **altKey** Logikai érték, az esemény közben a *Alt*-ot (Mac: *Option*) nyomva tartották-e

Ha egér váltotta ki az eseményt:

- **which, button** A lenyomott/felengedett gomb azonosítója. Csak az a biztos, hogy a jobb gomb száma 2, és a *click* esetén eltérően működnek a böngészők.
- **relatedTarget** W3C; *mouseover* esetén az előző elemet, *mouseout* esetén az új elemet adja meg, ami fölé a kurzor kerül
- **fromElement, toElement** ugyanez IE-ben

Eseményobjektum tulajdonságai

Hordozható megoldás

```
function mouseOverKezelo(e) {
        if (!e) var e = window.event;
        var elem = e.relatedTarget || e.fromElement;
}

function mouseOutKezelo(e) {
        if (!e) var e = window.event;
        var elem = e.relatedTarget || e.toElement;
}
```

Eseménykezelés teszt

esemeny.html

```
<!DOCTYPE html>
<html lang="hu">
  <head>
    <meta charset="utf-8">
    <title>Események vizsgálata</title>
  </head>
  <body>
    <form action="#" id="urlap">
      <fieldset>
        <legend>Válassza ki a tesztelendő eseményeket!</legend>
        <div><input type="checkbox" id="mouseover">
          <label for="mouseover">mouseover</label></div>
        <div><input type="checkbox" id="mouseout">
          <label for="mouseout">mouseout</label></div>
        <div><input type="checkbox" id="click">
          <label for="click">click</label></div>
        <div><input type="checkbox" id="keypress">
          <label for="keypress">keypress</label></div>
        <div><input type="checkbox" id="blur">
          <label for="blur">blur</label></div>
        <div><textarea id="naplo" rows="10" cols="40"></textarea></div>
        <div><input type="submit" id="gomb" value="Érvényesít"></div>
      </fieldset>
    </form>
    <script src="utilities.js"></script>
    <script src="esemeny.js"></script>
  </body>
</html>
```

Eseménykezelés teszt

esemeny.js

```
function naplozas(e) {
  'use strict';
  if(typeof e == "undefined") {
    e = window.event;
  }
  var cel = e.target || e.srcElement;
  var uzenet = cel.nodeName + ": " + e.type + "\n";
  U.$("naplo").value += uzenet;
}

function setEsemenykezelok(e) {
  'use strict';
  var esemenyek = ["mouseover", "mouseout", "click", "keypress", "blur"];
  for(var i=esemenyek.length-1; i>=0; i--) {
    var jelolo = U.$(esemenyek[i]);
    if(jelolo.checked) {
      U.addEvent(document, esemenyek[i], naplozas);
    } else {
      U.removeEvent(document, esemenyek[i], naplozas);
    }
  }
  U.$("naplo").value = "";
  return false;
}

window.onload = function() {
  'use strict';
  U.$("urlap").onsubmit = setEsemenykezelok;
};
```

Böngésző alapértelmezett viselkedésének megváltoztatása

Általában nem cél megváltoztatni a böngésző viselkedését, de néha szükséges: pl. hibás adatokat tartalmazó űrlapot nem küldünk el. Ezt tettük eddig is, az eseménykezelő végén lévő `return false;` segítségével.
Gond: csak akkor működik, ha klasszikus módszerrel állítottuk be az eseménykezelőt!
Mit lehet tenni?

- `e.preventDefault()`, ha a böngésző támogatja
- `e.returnValue = false`, <IE9 esetén

Hordozható megoldás

```
if (typeof e == "undefined") e = window.event;
if (e.preventDefault) {
  e.preventDefault();
} else {
  e.returnValue = false;
}
return false; // Biztos, ami tuti
```

Böngésző alapértelmezett viselkedésének megváltoztatása

Különbségek a `false` visszatérési érték alkalmazásához képest:

- a `return` azonnal megszakítja a fv. futását, az új módszerek nem
- `false` esetén a többi regisztrált eseménykezelő hívása ugyanazzal az esemény objektummal már nem történik meg
- a `false` megszakítja az esemény *bubbling* fázisát

Böngésző alapértelmezett viselkedésének megváltoztatása

teglatest.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Téglatest</title>
  </head>
  <body>
    <form action="#" id="urlap">
      <fieldset>
        <legend>Adja meg a méreteket!</legend>
        <div><label for="szelesseg">Szélesség</label>
          <input type="text" id="szelesseg"></div>
        <div><label for="magassag">Magasság</label>
          <input type="text" id="magassag"></div>
        <div><label for="melyseg">Mélység</label>
          <input type="text" id="melyseg"></div>
        <div><input type="submit" id="szamol" value="Számol"></div>
      </fieldset>
    </form>
    <p id="eredmeny"></p>
    <script src="utilities.js"></script>
    <script src="teglatest.js"></script>
  </body>
</html>
```

Böngésző alapértelmezett viselkedésének megváltoztatása

teglatest.js

```
function szamol(e) {
  'use strict';
  if(typeof e == "undefined") e = window.event;
  var szelesseg = U.$("szelesseg").value;
  var magassag = U.$("magassag").value;
  var melyseg = U.$("melyseg").value;
  if(isNaN(szelesseg) || isNaN(magassag) || isNaN(melyseg)) {
    U.setText("eredmeny", "Adjon meg helyes adatokat!");
  } else {
    var felszin = (szelesseg*magassag + magassag*melyseg +
      melyseg*szelesseg) * 2;
    var terfogat = szelesseg * magassag * melyseg;
    U.setText("eredmeny", "Felszín: " + felszin.toFixed(2) +
      ", térfogat: " + terfogat.toFixed(2));
  }
  if (e.preventDefault) {
    e.preventDefault();
  } else {
    e.returnValue = false;
  }
}
window.onload = function() {
  'use strict';
  U.addEvent(U.$("szelesseg"), "change", szamol);
  U.addEvent(U.$("magassag"), "change", szamol);
  U.addEvent(U.$("melyseg"), "change", szamol);
  U.addEvent(U.$("urlap"), "submit", szamol);
};
```

Eseménykezelési fázisok

Capturing:
[window] → [document] → [body] → [div] → [p] → [a]
Bubbling:
[window] ← [document] ← [body] ← [div] ← [p] ← [a]
Az esemény akkor is elkapható a *bubbling* miatt, ha az esemény az *a*-n keletkezik, de a kezelő pl. a *div*-hez van bejegyezve. A `target/srcElement` a valódi eseményforrást fogja jelölni.

Bubbling kikapcsolása

```
if (e.stopPropogation) { // W3C
  e.stopPropogation();
} else { // <IE9
  e.cancelBubble = true;
}
```

A fv. végi `return false;` is megakadályozza a bubbling-ot, de egyéb mellékhatásai is vannak.

Eseménykezelési fázisok

A böngészők alapértelmezés szerint a bubbling fázisban juttatják el az eseményt a kezelőhöz.
W3C: az `addEventListener()` utolsó paraméterének `true` értéke esetén a capture fázisban kerülnek az események a kezelőhöz.
Nincs minden eseménynek bubble fázisa: focus, blur, change, scroll, submit.

Események delegálása: az eseménykezelőt a szülő elemhez kötik, és minden gyermek elemeknél keletkezett eseményt itt dolgoznak fel
→ gyors

További információ

Beépített dialógusablakok

Típusok:

`alert(szöveg)`	Szöveg megjelenítés + OK gomb
`confirm(szöveg)`	Szöveg megjelenítés + OK/Cancel gombok Logikai értékkel tér vissza (OK = `true`, Cancel = `false`).
`prompt(szöveg, alapérték)`	Szöveg megjelenítés + egysoros szövegbeviteli mező + OK/Cancel gombok Visszatérési érték: `String`, ha az OK-ra kattintottak, `null`, ha a Cancel-re

Tulajdonságaik:

- Nem formázhatóak HTML/CSS-sel, nem bővíthetőek további vezérlőkkel (csak a sortörés engedélyezett `\n`-nel)
- Ahány böngésző, annyiféle megjelenés
- modálisak

Beépített dialógusablakok

Dialógusablak teszt

```
function dialogus() {
  var ital = prompt("Mi a kedvenc italod?", "sör");
  if(ital === null) {
    alert("Mi van veled, fáj a fejed?");
  } else {
    var ivas = confirm("Akkor meghívsz egy "+ital+"re?");
    if(ivas) {
      alert("Haver vagy.");
    } else {
      alert("OK, semmi baj.\nLegközelebb is ráérsz.");
    }
  }
}
```

A window objektum

A window objektum a legmagasabb szintű objektum a JS-ben, így minden objektumon kívül definiált objektum, függvény és változó közvetlenül ebbe kerül.

A window objektum elemei

```
function fv() {
  console.log("Működik!");
}
fv();
window.fv();

var valt = 3;
console.log(valt);
console.log(window.valt);

alert("Hahó!");
window.alert("Hahó!")
```

A window objektum

Megjegyzések:

- Nem szerencsés „teleszemetelni” a window objektumot rengeteg adattaggal → saját „csomagoló” objektumok létrehozása
- Sok eddig látott objektum is a window része: Math, Date, ...
- Adattagjain keresztül lekérdezhető (néha még be is állítható) a böngészőablak néhány eleme. Ma már ritkán használják ezeket a lehetőségeket.

A window érdekes adattagjai

```
if(window.menubar.visible) { ... // Menüsor látható
if(window.toolbar.visible) { ... // Gombsor látható
window.status = "Módosítjuk a böngésző állapotsorát";
  // Ha sikerül...
console.log(window.navigator.userAgent);
  // Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:16.0)
  // Gecko/20100101 Firefox/16.0
```

A böngészőablak mérete és helye

`screenX` / `screenLeft` Bal felső sarok X koordinátája (*/IE)

`screenY` / `screenTop` Bal felső sarok Y koordinátája (*/IE)

`innerHeight` / `document.body.clientHeight` Kliens terület magassága (*/<IE9)

`innerWidth` / `document.body.clientWidth` Kliens terület szélessége (*/<IE9)

`outerHeight` Böngészőablak magassága

`outerWidth` Böngészőablak szélessége

`moveTo(x, y)` Böngészőablak mozgatása adott helyre

`moveBy(x, y)` Böngészőablak elmozgatása adott képpontnyival

A képernyő tulajdonságai

`window.screen. ...`

`height`	Képernyő magassága (px)
`width`	Képernyő szélessége (px)
`availHeight`	Felhasználható magasság (px)
`availWidth`	Felhasználható szélesség (px)
`colorDepth`	Színmélység (bit)
`pixelDepth`	Színmélység (bit)

Új böngészőablak létrehozása

```
var ablak = window.open(url, név, tulajdonságok);
```

- `url` Ezt fogja tartalmazni az új ablak
- `név` Az ablak neve, `window.name`
- `tulajdonságok` Új ablak tulajdonságai később nem módosíthatóak. Formátum: tulajdonságnév=érték,... Az „érték" csak egész szám, `yes` vagy `no` lehet.

Fontosabb tulajdonságok:

- `height` Kliens terület magassága
- `width` Kliens terület szélessége
- `outerHeight` Az ablak magassága
- `outerWidth` Az ablak szélessége
- `top` Új ablak tetejének szülőhöz képesti helye
- `left` Új ablak bal oldalának szülőhöz képesti helye
- `location` A böngésző eszköztár láthatósága
- `menubar` A menüsor láthatósága
- `scrollbars` Gördítősávok láthatósága
- `status` Állapotsor láthatósága
- `toolbar` Eszköztár láthatósága
- `resizable` Átméretezhetőség

Visszatérési érték: az új ablak objektum, vagy `null`, ha nem jött létre.

Új böngészőablak létrehozása

Bezárás:

`window.close()`

Csak a kódból megnyitott ablakon működik.

Új ablak nyitása és bezárása

```
var sze = open("http://uni.sze.hu", "sze",
  "height=200,width=200,location=no," +
  "resizable=yes,scrollbars=yes");
alert("Kattintson, ha bezárhatjuk.");
if((sze !== null) && (!sze.closed)) {
    sze.close();
}
```

Másik ablakra fókuszálás:

`window.focus()`

Az új ablak elérése

Az új ablak csak JS segítségével elérhető, viszont

- nincs mindig bekapcsolva a JS,
- egyszerű böngészők, internetes keresők sem futtatnak JS-et, ...

Megoldás: az ablak megnyitását tegyük lehetővé hivatkozással is, majd definiáljunk eseménykezelőt a hivatkozás `click` eseményére!
JS ki/be: URL: about:config / „Óvatos leszek, megígérem!" / javascript.enabled → false (Firefox 25)

ablak.html

```
<!DOCTYPE html>
<html lang="hu">
  <head>
    <meta charset="utf-8">
    <title>Felugró ablakok</title>
  </head>
  <body>
    <a href="felugro.html" id="felugro" target="felugro_ablak">
      Kattintson ide új ablak megnyitásához!</a>
    <script src="ablak.js"></script>
  </body>
</html>
```

Az új ablak elérése

felugro.html

```
<!DOCTYPE html>
<html lang="hu">
  <head>
    <meta charset="utf-8">
    <title>Új ablak</title>
  </head>
  <body>
    <p>Hurrá, sikerült!</p>
  </body>
</html>
```

ablak.js

```
function kattintas() {
  'use strict';
  var felugro = open("felugro.html", "felugro_ablak",
    "height=200,width=200,top=100,left=100");
  if ((felugro !== null) && !felugro.closed) {
    felugro.focus();
    return false;
  }
}
window.onload = function() {
  'use strict';
  document.getElementById("felugro").onclick = kattintas;
};
```

Ablakok közötti kommunikáció

A szülő és gyermek ablakok kölcsönösen elérik egymás globális `window` objektumait, és ezen keresztül kommunikálhatnak egymással.

Szülő ismeri a gyereket: `var gyerek = open("gyerek.html");`
Gyerek ismeri a szülőt: `var szulo = window.opener;`

Ezután hívhatják egymás metódusait, használhatják egymás változóit, stb.

```
// Szülőben:
gyerek.window.feldolgoz(
  document.getElementById("urlapmezo").value);
// Gyerekben:
szulo.window.document.getElementById(
  "bekezdes").textContent = eredmeny;
```

Ablakok közötti kommunikáció

Csak az azonos helyről származó ablakok kommunikálhatnak!

Pl. `http://uni.sze.hu`

Eltérőek a helyek, ha:

- a protokoll nem egyezik, pl. `https://uni.sze.hu`
- más aldomainen belülről származnak az oldalak, pl. `http://it.sze.hu`
- más portot használnak, stb.

IFrame és ablak közti kommunikáció

Hasonlóan zajlik a kommunikáció a felek között, mint az ablakok esetében.

Szülő ismeri a keretet:
`var keret = document.getElementById("keretId");` vagy
`var keret = window.frames[keretIndex];` vagy
`var keret = window.frames["keretNév"];`
(ez tehát a `name` attribútum, nem az `id`!) majd
`var ablak = keret.contentWindow || keret.contentDocument;`

Gyerek ismeri a szülőt: `var szulo = window.parent;`
Van szülője? `if(window.parent != window.self) { ...`

A HTML5-ben új lehetőségek jelentek meg.

Ugrás az előzmények között

A vissza/előre lépésre ad lehetőséget (ha megvalósítható) a `window.history`.

- `go()` ami adott számú oldallal vissza/előre lépteti a böngésző előzményeket (az aktuális oldal indexe 0)
- `back()` előző oldalra lép, azaz `go(-1)`
- `forward()` következő oldalra lép, azaz `go(1)`

A HTML5 új lehetőségeket biztosít az előzmények kezeléséhez (AJAX alkalmazások használatának megkönnyítése érdekében).

Teljesen nem valósítható meg a *fokozatos fejlődés* elve.

Átirányítás

Szerencsésebb a webszerverrel megoldani (nem kell letölteni egy oldalt a másik oldal letöltéséhez), de néha szükség lehet a böngésző átirányítására.

Lehetőségek:

- `window.location = "http://uni.sze.hu";`
- `window.location.href = "http://uni.sze.hu";`
  Mindkét esetben módosulnak az előzmények. Ha ezt el akarjuk kerülni:
- `window.location.replace("http://uni.sze.hu");`
- Kezdő oldal: `window.home();`
- Lekérdező karakterlánc: `window.location.search`
- Horgony: `window.location.hash`

Nyomtatás

Oldal kinyomtatása: `window.print();`
Nem minden HW támogatja (okostelefon?)

nyomtat.html (részlet)

```
<body>
  <a href="#" id="nyomtat">Nyomtass, az ördögit!</a>
  <script src="nyomtat.js"></script>
</body>
```

nyomtat.js

```
window.onload = function() {
  'use strict';
  if(typeof window.print == "function") {
    document.getElementById("nyomtat").onclick = function() {
      window.print();
    }
  }
};
```

A document objektum néhány tulajdonsága

A document objektum reprezentálja a böngészőbe töltött weboldalt. Néhány metódusa:

`getElementById()` lekér a DOM-ból egy id-vel azonosított objektumot

`write()` hozzáír a dokumentum forrásához egy sort

`writeln()` mint a `write()`, de sort is emel

Utóbbiak használata nem javasolt, mert

- XHTML dokumentumokkal nem használható
- véletlenül felülírhatjuk a már betöltött dokumentumot
- módosítja a DOM-ot, bár ha ez a szándékunk, arra vannak jobb megoldások

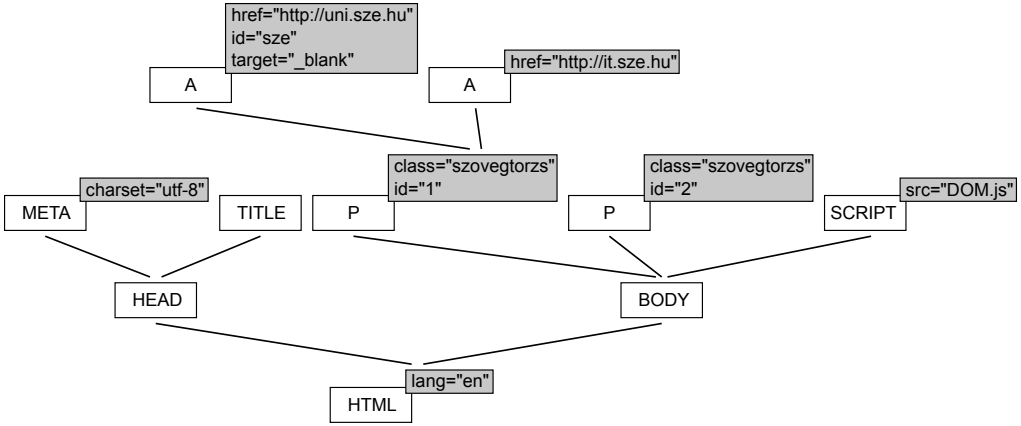
Adattagok:

`title` a böngésző fejlécében megjelenő cím

`compatMode` a böngésző kompatibilitási módjának állapota (DOCTYPE rendben?), értékek: *BackCompat*, *CSS1Compat*.

Document Object Model, DOM

DOM: W3C 1998-ban szabványosította. XML/XHTML/HTML adatok elérése, feldolgozása.



DOM.html

```
<!DOCTYPE html>
<html lang="hu">
  <head>
    <meta charset="utf-8">
    <title>DOM példa</title>
  </head>
  <body>
    <p class="szovegtorzs" id="1">Kattintson ide az
      <a href="http://uni.sze.hu" id="sze" target="_blank">egyetemi oldal</a>
      megnyitásához! Vagy nézegesse inkább a
      <a href="http://it.sze.hu">tanszéki</a> oldalakat!</p>
    <p class="szovegtorzs" id="2">További kellemes időtöltést!</p>
    <script src="DOM.js"></script>
  </body>
</html>
```

DOM objektumok elérése

Fa struktúra: csúcsok (node) és élek (az ábrán a text csúcsok nincsenek feltüntetve)
Gyökér: HTML
Minden csúcs egy objektum, adattagjaik:

`parentNode`	szülő csúcs
`childNodes`	gyerekek tömbje
`children`	csak a HTML elemek tömbje
`firstChild`	első gyerek
`lastChild`	utolsó gyerek
`nextSibling`	következő testvér
`previousSibling`	előző testvér
`nodeName`	az aktuális csúcs neve (pl. P, A, #text, #document)
`nodeValue`	csúcs értéke*
`nodeType`	csúcs típusa

DOM objektumok elérése

`nodeType` lehetséges értékei:

- 1 HTML elem
- 2 text
- 8 megjegyzések
- 9 document
- 10 a HTML gyökérelem*

`document.documentElement`: a HTML gyökérelemet adja meg

Tömbök az egyes elemtípusok gyorsabb eléréséhez: `document.`

- `forms` űrlapok
- `images` képek
- `links` hivatkozások

Indexelhetők számmal, vagy a `name` tulajdonság értékével

`document.head`, `document.body`

DOM objektumok elérése

Példák

```
document.body.parentNode;
  // <html lang="hu">
document.body.childNodes;
  // [<TextNode textContent="\n  ">, p#1.szovegtorzs, ...
document.body.children;
  // [p#1.szovegtorzs, p#2.szovegtorzs, script DOM.js]
document.body.firstChild;
  // <TextNode textContent="\n ">
document.body.nextSibling;
  // null
document.body.previousSibling;
  // <head>
document.body.nodeName;
  // "BODY"
document.body.nodeType;
  // 1
document.documentElement;
  // <html lang="hu">
document.links;
  // [a#sze, a]
document.links[0];
  // <a id="sze" target="_blank" href="http://uni.sze.hu">
```

DOM objektumok elérése

Elemek gyorsabb elérése:

- `document.getElementById()` adott id-jű elem objektumát adja, vagy `null`-t
- `getElementsByTagName()` bármilyen DOM objektumból hívható, a DOM fa ezen ágán lévő adott HTML elemek objektumaiból álló tömböt ad vissza
- `getElementsByClassName()` ilyen osztályú elemek tömbjével tér vissza (nem támogatott <IE9 esetén)

Példák

```
document.getElementById("1");
  // <p id="1" class="szovegtorzs">
document.body.getElementsByTagName("a");
  // [a#sze, a]
document.body.getElementsByClassName("szovegtorzs");
  // [p#1.szovegtorzs, p#2.szovegtorzs]
```

DOM objektumok elérése

CSS szelektorok használata is lehetséges (IE8+, IE8: CSS2.1)
Bármely DOM objektumon hívható metódusok további objektumok eléréséhez:

- `querySelector()` az első illeszkedő objektumot adja
- `querySelectorAll()` minden illeszkedő objektumot visszaad

Példák

```
document.body.querySelector("p.szovegtorzs");
  // <p id="1" class="szovegtorzs">
document.body.querySelectorAll("p.szovegtorzs");
  // [p#1.szovegtorzs, p#2.szovegtorzs]
document.body.querySelector(".szovegtorzs");
  // <p id="1" class="szovegtorzs">
document.body.querySelector("p a");
  // <a id="sze" target="_blank" href="http://uni.sze.hu">
document.body.querySelector("p > a");
  // <a id="sze" target="_blank" href="http://uni.sze.hu">
```

DOM objektumok módosítása

Az objektumok attribútumainak értékei módosíthatóak, pl.

Adattagok módosítása

```
document.getElementById("sze").href = "http://rs1.sze.hu";
document.getElementById("1").id = "3";
```

De a `class` és `for` szavak foglaltak a JS-ben, ezért

class, for

```
document.getElementById("1").className = "alap";
document.getElementById("cimke").htmlFor = "ujelem"; //*
```

DOM objektumok módosítása

Szöveges tartalom módosítása:

- `innerText` FireFox: nem megy, de ált. a többiben igen
- `textContent` W3C szabvány, IE: nem megy

textContent

```
document.getElementById("1").textContent;
// "Kattintson ide az egyetemi oldal megnyitásához!
// Vagy nézegesse inkább a tanszéki oldalakat!"
```

HTML tartalom módosítása:

- `innerHTML` gyak. minden böngészőben működik

innerHTML

```
document.getElementById("1").innerHTML;
  // "Kattintson ide az
  // <a href="http://rs1.sze.hu" id="sze" target="_blank">
  // egyetemi oldal</a>megnyitásához! Vagy nézegesse inkább a
  // <a href="http://it.sze.hu">tanszéki</a> oldalakat!"
```

DOM objektumok létrehozása

A módszer lassabb, mint az `innerHTML` alkalmazása, de néha elkerülhetetlen.

1. létrehozás: `document.createElement(típus)` új DOM objektumot hoz létre, ami nem része a fának
2. adattagok módosítása: `innerHTML`, `textContent`, ...
3. beszúrás a fába:
   `insertBefore(gyerek, testvér)` testvér elé helyezi
   `appendChild(gyerek)` az utolsó testvér lesz
   `replaceChild(gyerek, régi_gyerek)` testvér lecserélése

HTML elemek készítése

```
var bekE = document.createElement("p");
bekE.className = "szovegtorzs";
bekE.id = "legelso";
bekE.textContent = "Ez lesz a legelső bekezdés!";
document.body.insertBefore(bekE, document.getElementById("1"));

var bekK = document.createElement("p");
bekK.id = "kozepso";
bekK.textContent = "Ez lesz a középső bekezdés!";
document.body.replaceChild(bekK, document.getElementById("2"));

var bekU = document.createElement("p");
bekU.id = "legutolso";
bekU.textContent = "Ez lesz a legutolsó bekezdés!";
document.body.appendChild(bekU);
```

DOM objektumok létrehozása

További lehetőségek:

- szöveg csúcs létrehozása: `document.createTextNode("szöveg")` nem módosítja a többi csúcsot, mint egy `innerText/textContent` hívás
- csúcs másolása: `objektum.cloneNode()` nagy részfákat érdemesebb lemásolni, majd visszarakni a fába, mint minden részét külön-külön módosítgatni, mert elég 1x frissíteni a képernyőt
- gyerek eltávolítása: `szulo.removeChild(gyerek)`
- szülő lekérdezése: `gyerek.parentNode`

HTML elemek másolása, törlése

```
var duma1 = document.createTextNode("új szöveg");
var duma2 = duma1.cloneNode();
var egy = document.getElementById("1");
egy.insertBefore(duma1, document.getElementById("sze"));
egy.appendChild(duma2);

var ketto = document.getElementById("2");
ketto.parentNode.removeChild(ketto);
```

Stíluslapok használata

A DOM objektumok `style` objektumán keresztül módosítható a formázás, de

- mindig camelCase formázást kell használni
- mindig meg kell adni a mértékegységet
- soron belüli formázást eredményez (mint a `style` attribútum)

Hivatkozás méretének növelése

```
document.getElementById("sze").style.fontSize = "16pt";
```

Az alkalmazott stílusok kinyerhetőek:

- IE: `objektum.currentStyle`
- Hordozható: `window.getComputedStyle(objektum)`

Az eredmény csak olvasható.

Hivatkozás méretének lekérdezése

```
window.getComputedStyle(document.getElementById("sze")).fontSize;
  // "21.3333px"
```

Elemek eltüntetése, megjelenítése

Kétféleképpen lehetséges:

1. `visibility` stílussal: nem tördeli újra az oldalt. Lehetséges értékek: `hidden`, `visible`.
2. `display` stílussal: újratördeli az oldalt. Lehetséges értékek pl.: `inline`, `block`, `none`.

Láthatóság módosítása

```
var stilus = document.getElementById("1").style;
stilus.visibility = "hidden";
stilus.visibility = "visible";
stilus.display = "none";
stilus.display = "block";
```

Stíluslapok kezelése

stilus.html

```
<!DOCTYPE html>
<html lang="hu">
  <head>
    <meta charset="utf-8">
    <link rel="stylesheet" id="stilus" href="piros.css">
    <title>Stíluslap példa</title>
  </head>
  <body>
    <p>Milyen színű a szöveg?</p>
  </body>
</html>
```

piros.css

```
p {color: red}
```

kek.css

```
p {color: blue}
```

Stíluslapok kezelése

Stíluslap cseréje, törlése

```
var stilus = document.getElementById("stilus");
stilus.href = "kek.css"; // Most kék a szöveg
stilus.disabled = true; // Most fekete
```

Új szabály beszúrása

```
var elem = document.getElementById("stilus");
var stilus = elem.sheet ? elem.sheet : elem.styleSheet;
if (stilus.insertRule) { // legtöbb böngésző
  stilus.insertRule("p {font-size: 20pt}", 0);
} else { // <IE9
  if (stilus.addRule) {
    stilus.addRule("p", "font-size: 20pt", 0);
  }
}
```

Törlés hasonlóan: `stilus.deleteRule(index)` vagy `stilus.removeRule(index)`

Stíluslapok kezelése

Stílus készítése

```
var stilus = document.createElement("style");
if(stilus.textContent !== "undefined") {
    stilus.textContent = "p {font-size: 20pt;}";
} else {
    stilus.innerText = "p {font-size: 20pt;}";
}
document.body.appendChild(stilus);
```

Sütik

A NetScape fejlesztése. HTTP állapotmentes protokoll → állapotok tárolása lehetséges:

- munkamenetekkel a szerver oldalon (session)
- sütikkel kliens oldalon (cookie)

Sütik tulajdonságai:

- böngésző tárolja őket
- visszaküldi annak az oldalnak, ahonnét kapta
- kis méretűek (max. kb. 4kB)
- a felhasználó letilthatja használatukat
- a felhasználó olvashatja, alakíthatja tartalmát → érzékeny adatok tárolására alkalmatlan

Sütik által tárolt adatok:

- név
- érték
- érvényesség lejártának dátuma és ideje
- elérési út, amelyen érvényes (alapért.: aktuális út)
- tartomány (domain)

Sütik készítése

Egy süti kulcs-érték pár: `var suti = "color=red";`
Süti hozzárendelése az oldalhoz: `document.cookie = suti;`
Újabb hozzárendelés nem törli az eddigi sütiket!

```
document.cookie = "fontSize=20pt";
console.log(document.cookie);
 // color=red; fontSize=20pt
```

A sütik közé néhány böngésző szóközt is rak!
Süti tulajdonságait ;-k választják el egymástól:

```
document.cookie = "color=red;expires="+
    new Date(2012, 10, 30).toUTCString()+
    ";path=/mappa;domain=*.sze.hu";
```

A néven és értéken kívül más tulajdonság nem olvasható a `document.cookie` használatával!

Sütik használata

A sütik felhasználása a `document.cookie` változó feldolgozásával lehetséges.

Sütik olvasása

```
function suti() {
  var sutik = document.cookie.split(";");
  var so = {
    nev: [],
    ertek: []
  };
  for(var i=sutik.length-1; i>=0; i--) {
    var akt = sutik[i].split("=");
    so.nev.push(akt[0].trim());
    so.ertek.push(akt[1]);
  }
  return so;
}
console.log(suti());
```

Sütik törlése

A sütik azonos nevű, de érték nélküli, a múltban lejáró süti létrehozásával törölhetőek.

```
document.cookie =
    "fontSize=;expires=Thu, 01-Jan-1970 00:00:01 GMT";
```

Süti könyvtár készítése

Praktikus, ha sokat kell dolgozni sütikkel. Ld. ©Larry Ullman, `cookies.js`

Sütik használata

suti.html

```
<!doctype html>
<html lang="hu">
<head>
    <meta charset="utf-8">
    <link rel="stylesheet" id="szin" href="piros.css">
    <title>Süti példa</title>
</head>
<body>
    <p>Válasszon bekezdésszínt:
      <a href="#" id="piros">pirosat</a>
      vagy <a href="#" id="kek">kéket</a>.</p>
    <script src="cookies.js"></script>
    <script src="suti.js"></script>
</body>
</html>
```

Sütik használata

suti.js

```
function beallit(e) {
    'use strict';
    if (typeof e == "undefined") e = window.event;
    var elem = e.target || e.srcElement;
    var lejar = new Date();
    lejar.setDate(lejar.getDate() + 7);
    COOKIE.setCookie("szin", elem.id, lejar);
    document.getElementById("szin").href = elem.id + ".css";
    return false;
}

window.onload = function() {
    'use strict';
    document.getElementById("piros").onclick = beallit;
    document.getElementById("kek").onclick = beallit;
    var szin = COOKIE.getCookie("szin");
    if (szin) {
        document.getElementById("szin").href = szin + ".css";
    }
};
```

Időzítők

Nincs garancia a pontos időzítésre!

Egyszeri esemény: `var t = setTimeout(fv, kesleltetes_ms);`
Ismétlődő esemény: `var t = setInterval(fv, varakozas_ms);`
Időzítő törlése: `clearInterval(t);`

Feladat: pontos idő kijelzése, képernyő rendszeres frissítésével.

Időzítők

ido.html

```
<!DOCTYPE html>
<html lang="hu">
  <head>
    <meta charset="utf-8">
    <title>Pontos idő kijelzése</title>
  </head>
  <body>
    <p>A pontos idő:<span id="ido"></span></p>
    <script src="ido.js"></script>
  </body>
</html>
```

Időzítők

ido.js

```
function frissit() {
  'use strict';
  var cel = document.getElementById("ido");
  if(cel.textContent !== "undefined") {
    cel.textContent = new Date().toLocaleTimeString();
  } else {
    cel.innerText = new Date().toLocaleTimeString();
  }
}

window.onload = function() {
  'use strict';
  setInterval(frissit, 1000);
};
```

Bevezetés a JavaScript használatába

3. rész

Hatwágner F. Miklós

Széchenyi István Egyetem, Győr

2013. november 24.

Űrlapok

Általános tanácsok:

- kitöltött adatok ellenőrzése küldés előtt; hiba esetén alapértelmezett működés megakadályozása
- küldés gomb ki/bekapcsolása (`document.getElementById("kuldes").disabled = true;`)
- súgó:
  - billentyűzettel és egérrel is legyen használható (csak az aktuális mezőhöz tartozó látszódjék)
  - hibás adat esetén hibajelzés (pl. üres `span` elemben) hibaüzenet megjelenítése)

Űrlapok

Szöveges beviteli mezők:

- tárolt érték elérése a `value` attribútumon keresztül érhető el, és ezen keresztül módosítható is
  ```
  var adat = document.getElementById("szoveg").value;
  ```
- Használható `text`, `textarea`, `password` és `hidden` típusú űrlapmezőkkel
  ```
  <input id="nev" type="text" />
  ```
- HTML5: `maxlength` tulajdonsággal korlátozható a szöveg hossza
- input ellenőrzés: ne ellenőrizzük a helykitöltő, eredeti adatokat
  ```
  <input type="text" id="nev" value="&lt;Gipsz
  Jakab&gt;" required />
  if (nev.value != nev.defaultValue) { ...
  ```

Űrlapok

Legördülő lista (`select`)

- a `value` adattag csak lekérdezésre használható, módosításra nem!
- `selectedIndex`: elsőnek kiválasztott elem indexe, vagy -1
- `options` tömb: opció objektumok → `selected`, `value`, `text`.

Legördülő lista teszt #1

select1.html

```
<!DOCTYPE html>
<html lang="hu">
  <head>
    <meta charset="utf-8">
    <title>Legördülő lista teszt</title>
  </head>
  <body>
    <form>
      <div>
        <label for="legordulo">Többszörös választás</label><br />
        <select id="legordulo" multiple>
          <option value="a">Aladár</option>
          <option value="b">Béla</option>
          <option value="c">Cili</option>
        </select>
      </div>
      <div><p>Kiválasztott elemek: </p><span id="valaszt"></span></div>
    </form>
    <script src="select1.js"></script>
  </body>
</html>
```

Legördülő lista teszt #1

select1.js

```
function frissit() {
  'use strict';
  var legordulo = document.getElementById("legordulo");
  var kijelolve = "Elsőnek választott: ";
  if(legordulo.selectedIndex != -1) {
    kijelolve += legordulo.selectedIndex + "<br />";
  } else {
    kijelolve += "egyik sem";
  }
  for(var i=0, db=legordulo.options.length; i<db; i++) {
    var opcio = legordulo.options[i];
    if(opcio.selected) {
      kijelolve += opcio.text + " (" + i + ", " + opcio.value + ") ";
    }
  }
  document.getElementById("valaszt").innerHTML = kijelolve;
}
window.onload = function() {
  'use strict';
  document.getElementById("legordulo").onchange = frissit;
};
```

Űrlapok

Legördülő lista előállítása másikból történő választás alapján

- kényelmes: nem kell hosszú listából választani → AJAX
- sajnos az IE nem kezeli helyesen az `innerHTML`-t → DOM manipuláció
- fel kellene készülni arra az esetre is, ha nincs JS támogatás!

Legördülő lista teszt #2

select2.html

```
<!DOCTYPE html>
<html lang="hu">
  <head>
    <meta charset="utf-8">
    <title>Legördülő lista teszt</title>
  </head>
  <body>
    <form>
      <div>
        <label for="gyarto">Gyártó:</label>
        <select id="gyarto">
          <option>Válasszon!</option>
          <option value="opel">opel</option>
          <option value="vw">vw</option>
        </select>
      </div>
      <div>
        <label for="tipus">Típus:</label>
        <select id="tipus" disabled>
        </select>
      </div>
    </form>
    <script src="select2.js"></script>
  </body>
</html>
```

Legördülő lista teszt #2

select2.js

```
function frissit() {
  'use strict';
  var gyarto = document.getElementById("gyarto");
  var tipus = document.getElementById("tipus");
  while(tipus.firstChild) {
    tipus.removeChild(tipus.firstChild);
  }
  var lista = null;
  if(gyarto.value == "opel") {
    lista = ["astra", "movano"];
  } else if(gyarto.value == "vw") {
    lista = ["golf", "transporter"];
  }
  if(lista) {
    tipus.disabled = false;
    for(var i=0, db=lista.length; i<db; i++) {
      var elem = document.createElement("option");
      elem.value = elem.text = lista[i];
      tipus.appendChild(elem);
    }
  } else {
    tipus.disabled = true;
  }
}

window.onload = function() {
  'use strict';
  document.getElementById("gyarto").onchange = frissit;
};
```

Jelölőnégyzetek

- `value` adattag: a `value` attribútum lekérdezéséhez
- `checked` adattag: a jelölés állapotának lekérdezéséhez/módosításához
- feladat: csoport minden jelölőjének beállítása, törlése
- ötlet: csak a jelölők lekérdezése

```
var jelolok = document.querySelectorAll(
    "input[type='checkbox']");
```

Jelölőnégyzet teszt

check.html

```
<!DOCTYPE html>
<html lang="hu">
  <head>
    <meta charset="utf-8">
    <title>Jelölőnégyzet teszt</title>
  </head>
  <body>
    <form>
      <fieldset>
        <legend>Opel</legend>
        <label><input type="checkbox" id="astra" class="opel">Astra</label>
        <label><input type="checkbox" id="movano" class="opel">Movano</label>
        <a href="#" id="opel_j" class="opel">Mindet kijelöli</a> |
        <a href="#" id="opel_t" class="opel">Mindet törli</a>
      </fieldset>
      <fieldset>
        <legend>Volkswagen</legend>
        <label><input type="checkbox" id="golf" class="vw">Golf</label>
        <label><input type="checkbox" id="transporter"
                class="vw">Transporter</label>
        <a href="#" id="vw_j" class="vw">Mindet kijelöli</a> |
        <a href="#" id="vw_t" class="vw">Mindet törli</a>
      </fieldset>
    </form>
    <script src="check.js"></script>
  </body>
</html>
```

Jelölőnégyzet teszt

check.js

```
function jelol(e) {
  'use strict';
  if(typeof e == "undefined") {
    e = window.event;
  }
  var cel = e.target || e.srcElement;
  var kibe = (cel.id.slice(-2)=="_j") ? true : false;
  var gyarto = document.body.querySelectorAll("input." + cel.className);
  for(var i=gyarto.length-1; i>=0; i--) {
    gyarto[i].checked = kibe;
  }
  return false;
}

window.onload = function() {
  'use strict';
  var jelolo = document.body.querySelectorAll("a");
  for(var i=jelolo.length-1; i>=0; i--) {
    jelolo[i].onclick = jelol;
  }
}
```

Űrlapok

Rádiógombok

- sajnos nem tudjuk könnyen megmondani, melyik a kijelölt elem (ha van ilyen egyáltalán) → végig kell nézni az összeset
- ötlet: `document.getElementsByName(név);`
- `value` és `checked` tulajdonságok

Rádiógomb teszt

radio.html

```
<!DOCTYPE html>
<html lang="hu">
  <head>
    <meta charset="utf-8">
    <title>Rádiógomb teszt</title>
  </head>
  <body>
    <form action="#" id="jegyek">
      <fieldset><legend>Érdemjegy</legend>
        <div><label><input type="radio" value="1"
                     name="jegy">Elégtelen</label></div>
        <div><label><input type="radio" value="2"
                     name="jegy">Elégséges</label></div>
        <div><label><input type="radio" value="3"
                     name="jegy">Közepes</label></div>
        <div><label><input type="radio" value="4"
                     name="jegy">Jó</label></div>
        <div><label><input type="radio" value="5"
                     name="jegy">Jeles</label></div>
      </fieldset>
      <div><input type="submit" value="Nos?">
           <span id="eredmeny"></span></div>
    </form>
    <script src="radio.js"></script>
  </body>
</html>
```

Rádiógomb teszt

radio.js

```
function kiir() {
  'use strict';
  var rgt = document.getElementsByName("jegy");
  var eredmeny = "Nem kapott jegyet.";
  for(var i=0; i<rgt.length; i++) {
    var radio = rgt[i];
    if(radio.checked) {
      var cimke = radio.parentNode;
      var felirat = cimke.textContent || cimke.innerText;
      eredmeny = radio.value + " (" + felirat + ")";
      break;
    }
  }
  document.getElementById("eredmeny").innerHTML = eredmeny;
  return false;
}

window.onload = function() {
  'use strict';
  document.getElementById("jegyek").onsubmit = kiir;
}
```

Felhasznált irodalom

Larry Ullman
Modern JavaScript: Develop and Design
Peachpit Press (ISBN 0-321-81252-2), 2012.