HTML, CSS összefoglalás és tartalomkezelő rendszerek

Eddigi előadások

- Eddigi előadások letöltési linkjei
- https://dl.dropboxusercontent.com/u/93156461/!deleteme/!Tanfolyam/01.zip
- https://dl.dropboxusercontent.com/u/93156461/!deleteme/!Tanfolyam/02.zip
- https://dl.dropboxusercontent.com/u/93156461/!deleteme/!Tanfolyam/03.zip
- https://dl.dropboxusercontent.com/u/93156461/!deleteme/!Tanfolyam/04.zip

HTML

- Egy leíró nyelv minden előnyével (könnyen érthető kód) és hátrányával (sok tag/attribútum, megengedő szintaxis ezért nem mindig egyértelmű az eredmény) együtt
- Tagek között találhatóak az elemeink, szövegeink
- A tageket legtöbbször be kell zárni
- A HTML-t ne formázásra hanem tartalom leírásra használjuk
- Sok tagről volt már szó, talán az egyik legfontosabb a div tag, aminek adunk egy class (osztály) nevet, vagy ID-t (azonosítót)
- Ezek segítségével tudunk a CSS kódban hivatkozni az eleminkre

CSS

- A CSS-t használjuk formázásra
- Doboz modell alapján helyezzük el, pozicionáljuk az elemeinket
- Figyeljünk oda a méretezési arányokra
- Sok tulajdonság, érték párról volt már szó, nézzünk mindig utána a használatuknak, így időt spórolhatunk meg a hibakereséssel
- A CSS kód külső fájlban legyen így több oldalunkhoz is könnyen belinkelhetjük ugyanazokat a CSS stílus tulajdonságokat

CSS mértékegységek

- A legegyszerűbb mértékegységek:
  px – pixelben megadott mérték
  cm – centiméterben
  % - százalékban megadott mérték az ősosztály (legutóbb volt ősosztály<->származtatott osztály) méreteihez képest
- Speciális mértékegységek (böngésző-támogatottságuk változó):
  em – az alapértelmezett szövegméretéhez képesti szorzó
  vw – (viewport width) a böngésző szélességéhez képesti százalékos mérték
  vh – (viewport height) a böngésző magasságához képesti százalékos mérték
- Fontos tudni, hogy ha a padding vagy margin tulajdonságok értékét százalékban (%) adjuk meg, akkor az a böngésző szélességéhez képesti százalékot jelent, tehát ha pl a böngésző magasságához képest szeretnénk megadni, akkor a vh mértékegységet kell használnunk
- Referencia: http://www.w3schools.com/cssref/css_units.asp

Böngésző kompatibilitás

- Honlapkészítésnél a böngésző kompatibilitás komoly problémákat jelent
- A felhasználók számtalan típusú és méretű böngészőkből szeretnék az oldalunkat megnyitni, ezért minél több méretre és típusra tervezni kell (pl legutóbbi példában mivel fix px méreteket adtam a dobozaimnak, ezért otthon nekem jól jelentek meg, a tanári gépen meg összecsúsztak, mert kisseb a felbontása)
- Ezért jó hozzáállás az, hogyha minél többször (de nem mindig!) százalékos arányméreteket adunk meg fix px (pixel) értékek helyett
- Korlátozzuk a speciális CSS3 függvényhívások számát
- Ha minél több verziószámú, típusú böngészőt szeretnénk támogatni, akkor ugyanahhoz a tulajdonsághoz sorban egymás után úgy adjunk értékeket, hogy a legtámogatottabb érték legyen elöl és a legspeciálisabb érték a végén (pl CSS3 függvényhívás, vagy speciális mértékegység)
  Pl: margin-top:150px;
  margin-top:20vh;
- Kompatibilitási táblák referencia: http://caniuse.com/

Responsive web desing

- A responsive web design célja az, hogy az oldalunk minden képernyőméreten (akár telefonokon), felbontáson jól nézzen ki
- A böngészőméretre érzékeny tulajdonságokat megadhatunk CSS-ben, ekkor megadhatjuk feltételben a böngészőnk minimális/maximális szélességét/magasságát:
- @media only screen and (min-width: 1500px) and (min-height: 600px) {
  .home {
  font-size: 3em;
  }
  }
- Ez a tulajdonság akkor lesz beállítva, ha a böngészőnk szélessége legalább 1500px, magassága legalább 600px
- Az alapértelmezett tulajdonságokat ez előtt adjuk meg
- Referencia: http://www.w3schools.com/html/html_responsive.asp

Elérés és kommunikáció

- A webszerverek alapértelmezetten az index.html, index.htm, index.php kezdőoldalt keresik a szervergépünk főkönyvtárában (WAMP pl: www mappában), ha a felhasználó (kliens) nem ad meg elérési útvonalat az URL címben (tehát a felhasználó csak a domainnevet, vagy csak a külső IP címet írja be a böngészőjébe)
- Ekkor a kommunikáció (http) porjta a 80-as lesz, tehát ha szervert szeretnénk futtatni a saját gépünkön, akkor a routerünkben a 80-as portot kell továbbítani (a szervergépünk lokális IP címéhez rendeljük a beállítást, így külső csatlakozó el tudja érni a weblapunkat)

Elérési útvonal

- A szervergépünk főkönyvtárában (pl www mappa) elhelyezzük a kezdőoldalt (pl index.html) és pl ugyanebbe a mappába kerülhetnek az aloldalak is más névvel (pl aloldal.htm)
- Weboldalak általános tulajdonsága, hogy linkek segítségével navigálhatunk, böngészhetjük az oldalt, ezt a HTML kódunkban a link az aloldalra segítségével tehetjük meg, a href attribútumnak itt most relatív elérési útvonalat adtunk meg, tehát akkor találja meg az aloldalunkat, ha ugyanabban a mappában van mint ahonnan navigálnák oda
- Ha a főkönyvtáron belül egy mappában lenne az aloldalunk, akkor így tudjuk elérni: link az aloldalra, ez megint egy relatív elérés
- Ha a keresett html fájl egy mappával feljebb (kijjebb) van, akkor elérése: link az aloldalra
- A számítógép mappastruktúra nyelvén a ‘.’ a jelenlegi mappát, a ‘..’ egy feljebb levő mappát jelent

URL elérési útvonal

- Ha egyből egy aloldalra szeretnénk jutni, akkor az URL címben megadjuk a protokollt (http vagy https), a domain nevet és utána írjuk az elérni kívánt aloldal elérési útvonalát
- Pl: https://www.oldalam.hu/almappa/aloldal.htm
- Ez azért is jó, mert ezt könyvjelzőzhetjük egyből, nem kell mindig a főoldalról az aloldalakra átnavigálni, böngészni
- Ha a html kódunkban egy elemünknek, szövegrészünknek ID azonosítót adunk, akkor a böngészőnk képes arra, hogy egyből erre az elemünkre görgessen, ezt is az URL címben kell megadni:

  https://www.oldalam.hu/almappa/aloldal.htm#azonosito

  Ezt is könnyen könyvjelzőzhetjük, pl a wikipédia főrovatai egyéni azonosítókat kaptak, ezért a böngészőnk könnyen ezekre tud ugrani

URL php változóval

- Ha php szkriptet használ a weboldalunk, akkor az URL cím végén kérdőjel után megadhatunk változó nevet és értéket, ezeket & (and) jellel választjuk el egymástól, a szóközök helyett + jel van
- Pl:
  https://www.oldalam.hu/almappa/aloldal.php?name=Bela&foglalkozas=tanulo
- Ekkor a php szkriptben lekérhetjük a változóértékeket és felhasználhatjuk őket, ezt így tehetjük meg a szkriptünkben:
  $_GET[„valtozonev”]
  (itt pl: $_GET[„name”] és máshol meg $_GET[„foglalkozas”])
- Ezek a $_GET tömbök a megfelelő változónévhez tartozó változóértéket adják vissza, amit felhasználhatunk a kódunkban bárhol
  (pl.: weblap elején: echo „Üdv”. $_GET[„name”]
  ez jelenik meg a kimeneten: Üdv Bela)
- Ez a változóátadás azért jó, mert a felhasználó is könnyen átírhatja az értékeket a böngésző URL címében (tehát pl ha listázási elemek száma van így megadva, akkor manuálisan beírhatunk bármennyit igényeink szerint), viszont azért rossz, mert minden változóértéket látna így minden felhasználó, tehát fontos adatokat ne így adjunk át a szkriptünknek!!!

Űrlap adatátadás

- A HTML kódunkban megadhatjuk hogyan szeretnénk az űrlapba beírt adatokat átadni a feldolgozó szkriptünknek
- HTML-ben <form></form> taggel jelezzük az űrlaprészt
- Attribútumként megadjuk a feldolgozó szkript elérési útvonalát, vagy ha nem adjuk meg akkor önmagának fogja átadni az adatokat és még megadjuk az adatátadás típusát
- Két típus van: get, post
- A get típus úgy működik ahogy az előző dián láthattuk, tehát ez az adat az URL címben is megjelenik
- A post típust minden fontos információ (pl jelszó) átadására, ez nem jeleik meg az URL címben
- Pl: <form method=„post”>
  <input type=„text” name=„firstname” value=„Keresztnév Ide”>
  … </input>
  </form>
- A változónevet az input tag attribútumában kell megadni: name=„valtozonev” és az alapértelmezett értékét is: value=„valtozoertek”

Webszerver telepítése

- Webszerver csomagot kell telepítenünk ami tartalmaz: Apache HTTP szervert, MYSQL adatbázis kezelőt, PHP értelmezőt
- Ilyen csomag pl.: WAMP, LAMP, XAMPP
- Mi most a WAMP szervert telepítjük: http://www.wampserver.com/en/
- Telepítés után jobb alsó sarokban láthatjuk az ikonját
  Jobb klikk rá: átállíthatjuk a nyelvet magyarra ha akarjuk
  Bal klikk rá: kezelhetjük a szerverünket
- A HTML, CSS fájlainkat rakjuk be a 'c:\wamp\www' mappába (töröljük ki nyugodtan az ottani fájlokat)
- Az Apache HTTP szerver beállításait a c:\wamp\bin\apache\apache2.4.9\conf\httpd.conf fájlban találhatjuk, ezt nyissuk meg notepad++-al, itt láthatjuk, hogy az első oldalunknak a fájl neve lehet index.html/index.htm/index.php… is
- A konfigurációs fájl is követi a leíró nyelvek sémáját (nyitó, záró tagek vannak benne)

A httpd.conf

- Az Apache HTTP szerver beállításait a c:\wamp\bin\apache\apache2.4.9\conf\httpd.conf fájlban találhatjuk, ezt nyissuk meg notepad++-al
- Itt keressük meg a <Directory "c:/wamp/www/"> részt, és itt írjuk át a '*Require local granted*' részt '*Require all granted*'-ra.
- Így a weboldalunkat meg tudja nyitni az internetről bárki, ha tudja a külső IP címünket, és a routerünkben továbbítva van a HTTP (80-as) port
- A weboldalunkat gyorsan megnyithatjuk a WAMP ikon bal klikk után->localhost segítségével (index.php,index.html nyitódik meg)

WAMP PHP értelmező

- A WAMP webszerver-csomag PHP értelmezőt is tartalmaz, tehát megpróbálhatjuk újra az űrlapok (Form) kezelését. A „10 - Form” mappából másoljuk ki az index.php-t a ‘c:\wamp\www’ mappába
- A php szkriptet is tartalmazó html fájlunk kiterjesztése: .php
- Ebben minden pont ugyanúgy működik mint eddig (HTML CSS szabályok), de ebbe akár php szkriptet is beírhatunk <?php …php szkript… ?> közé
- Egyelőre nem tanuljuk meg a php nyelvet, csak néhány alapdolgot
- Adatküldés az űrlapban (formban) php szkriptnek kétféleképpen történhet: vagy a böngészőben az URL cím után megadjuk az adatot (method=„get"): localhost/?nev=ertek
  Vagy method="post” segítségével, személyes információátadásnál mindig az utóbbit használjuk!!!
- A „10 – Form” példában az index.php fájlban az elején GET módszerrel az URL cím végéből megkapjuk a nevet, ha ezt megadjuk (URLcim?name=AzEnNevem)
- Formok (űrlapok) készítésénél megadjuk a feldolgozó php szkript nevét (itt most saját maga dolgozza fel) és az információküldés típusát (itt: method="post")

PHP alapok

- A form (űrlap) definícióján belül megadtuk a beviteli típusokat (input type=„…”) majd ezeknek adtunk name, value attribútumokat. A PHP így tud adatnevekhez értéket rendelni.
- Az adatküldés típusától függően (method=„get” vagy method=„post”) egy-egy tömbből olvashatjuk ki és dolgozhatjuk fel az adatnevek értékeit. A két tömb: $_GET[„adatnév”] és $_POST[„adatnév”]. Ezek a tömbök a megfelelő elküldött adatértéket adják vissza.
- Majd az elküldött adatokat az echo függvény segítségével kiírattam az index.php fájl végén

A HTML egy leíró nyelv…

- Ezért:
  - Nagyon sok tag van benne rengeteg attribútum párosítással, ezeket képtelenség fejben tartani
  - Annak ellenére, hogy megengedő a szintaxisa, számunkra természetes definíciókat nem biztos, hogy megért (pl: <div=„classnev”> hibás!, helyette:)
  - Hibakeresés nehézkes, sokszor nehezen találjuk meg, vagy nem értjük a hibákat
  - Kódmódosítgatással nagyon lassan lehet egy normálisan kinézők honlapot készíteni
  - Kompatibilitási problémák
  - A mellékhatásokkal számolni kell, nem mindig egyértelmű, hogy a definíciók milyen sorrendben kell hogy következzenek
- Megoldás: Tartalomkezelő rendszerek

Tartalomkezelő rendszerek

- Tartalomkezelő rendszerek (CMS) segítségével profibban kinéző oldalakat csinálhatunk sablontémák alapján rövidebb idő alatt
- Ilyen rendszer pl.: WordPress, Joomla
- Mi most a WordPress-t fogjuk használni, mert ezt a legkönnyebb telepíteni, konfigurálni, legtöbb téma ehhez készült
- Letöltés: https://wordpress.org/download/
- Bontsuk ki és a mappán belüli (tehát wordpress mappán belül) fájlokat (és mappákat) másoljuk át a ‘c:\wamp\www’ mappánkba.

WordPress előkészületek

- WAMP szerver ikonra bal klikk->válasszuk a phpMyAdmin-t, itt találhatjuk meg az adatbázisainkat
- Válasszuk ki a Databases fület
- Itt adjunk meg egy új nevet: wordpress, a Collation-ból válasszuk ki '*utf8_hungarian_ci*', ->aztán Create
- Ha sajátgépről WAMP szervert használunk akkor ezeket nem kell megcsinálni:
- ((Menjünk vissza a kezdő phpMyAdmin oldalra, most válasszuk ki a Users fület
- Itt Add user
- Username: wordpress
  Password: (itt adjunk meg egy saját jelszavat, nagyon fontos, hogyha saját szervert akarunk csinálni, akkor hosszú biztonságos jelszó legyen (akár generálni is lehet jelszavat))
- Klikkeljünk a Go gombra
- A most létrehozott wordpress felhasználónevünknél klikk Edit Privileges -> Database-specific privileges-nél válasszuk ki a wordpress adatbázist, majd frissül az oldal és itt jelöljük be a *Check All* pipát -> aztán Go gomb))

WordPress telepítés

- Nyissuk meg az oldalunkat és itt el kell induljon a telepítő lépésről lépésre
- Válasszunk ki egy nyelvet és menjünk tovább
- Itt adjuk meg az adatbázisunk nevét, amit majd használni fog a WordPress: wordpress
  Felhasználónév: root (Ha WAMP szervert használunk)
  Jelszó: hagyjuk üresen
  Többit ne bántsuk
- Tovább: itt írjuk be a címét az oldalunknak, adjunk meg egy felhasználónevet, majd az ahhoz generált jelszavat jegyezzük fel valahova (ennek nagyon biztonságos jelszónak kell lennie) és adjuk meg az e-mail címünket
- És voila kész is a telepítés
- Jelentkezzünk be és kezdődhet az átalakítás!

WordPress átalakítás

- WordPress egy előre elkészített téma alapján készíti el az oldalainkat. Ezek a témák teljesen átalakíthatóak vizuális szerkesztésen át egészen a forráskódig (Appearance->Editor).
- Bármilyen mások által elkészített témát kiválaszthatunk, anélkül hogy az egész oldalunkat újra kéne szerkeszteni. (Appearance->Themes)
- Könnyen használhatjuk blogként is az oldalunkat, egyből létrehozhatunk posztokat.
- Könnyen átalakíthatjuk professzionálisan kinéző honlapra
- Hozzáadhatunk Pluginokat is (vannak pl lejátszók, effektek…)
- Hozzáadhatunk, kezelhetünk felhasználókat is, hogy mások is szerkeszthessék az oldalunkat pl.