Webszerverek és tartalomkezelő rendszerek (CMS)

CSS pozicionálás

- CSS-ben a position tulajdonság 4 értéket kaphat: static, relative, fixed, absolute
- Minden szövegrész (elem) alapértelmezetten a positon:static értéket kapja (úgy következnek egymás után az elemek, ahogy a HTML kódban írtuk)(ezt nem lehet elmozdítani a helyéről)
- position:relative; segítségével az eredeti helyéhez képest ahova kerülne az elemünk megadhatjuk, hogy melyik irányba (left,right,top,bottom) hány pixellel (px) mozgassa el
- position:absolute; segítségével az ősosztályon belül adhatjuk meg, hogy melyik irányba mennyivel helyezze el (ősosztály: egy olyan nagyobb szövegrész amiben a mi szövegünk (elemünk) elhelyezkedik, az osztályok hierarchiába rendezhetőek (ősosztályon belül származtatott osztályok),
- tehát pl.: Sok szöveg Speciális szövegünk Sok szöveg
- Itt az általános tulajdonságokat az ososztaly classban adjuk meg, az egyéni speciális tulajdonságokat a szarmaztatottosztaly classban adjuk meg, a szarmaztatottosztaly örökli az ososztaly tulajdonságait

CSS pozicionálás

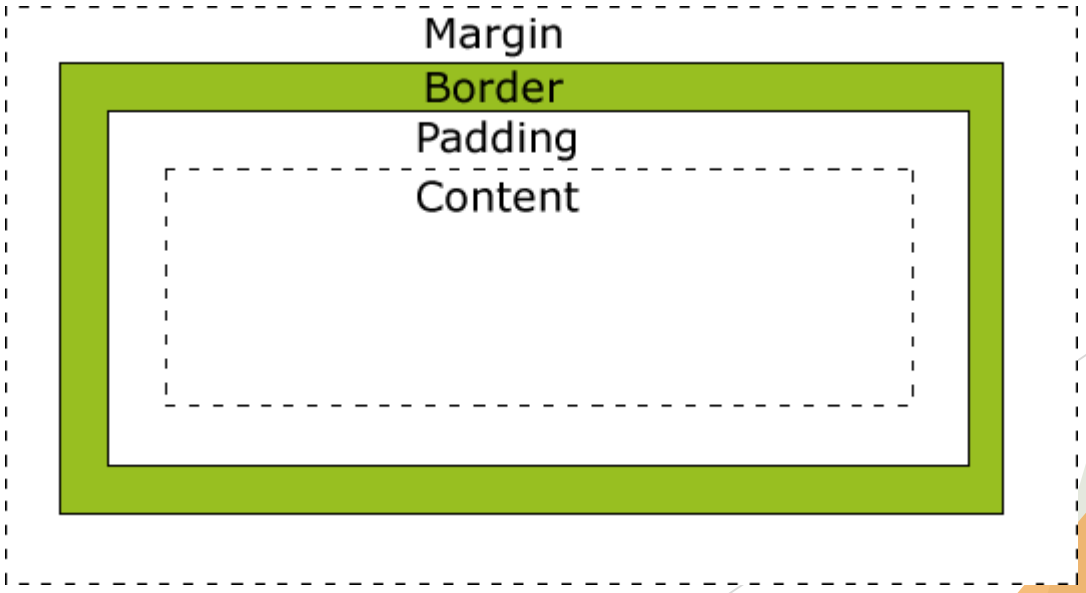
- Viszont sok tulajdonság nem öröklődik!! (legtöbbször újra ki kell írni a tulajdonságdefiníciókat (pl.: pozicionálás, méretezés, elhelyezés))
- Fontos, hogy megértsük a leszármaztatott (öröklött), ősosztály kapcsolatát, ugyanis ez az Objektum Orientált Programozás alapja (lásd később Java prognyelvben)
- Tehát ha pl egy div-nek nincs megadva saját ősosztály, akkor ősosztálya a body lesz (ezért a body alaptulajdonságait minden elem és szövegrész örökli), ilyenkor a div-ünk pozicionálása (position:absolute;) az egész honlapoldalunkra kiterjedő pozicionálást jelent
- positon:fixed; azt jelenti, hogy a böngészőablakunkhoz képest pozicionálunk (és így a böngészőbe rögzített rész egérgörgetés ellenére se mozdul el)
- Ha megadtuk a pozicionálás típusát, akkor a left:?px right:?px top:?px bottom:?px segítségével értelemszerűen balról, jobbról, felülről és lentről számított pixeleltolást jelent

CSS méretezés

- Alapértelmezetten a méreteket a szövegek és az elhelyezések alapján próbálja beállítani, úgy hogy igényesen nézzen ki
- Egy div-nek nem csak az elhelyezését, hanem a méretét is megadhatjuk (lásd doboz (box) modell), így létrehozhatunk dobozelemeket (téglalapokat amiben a szövegünk van pl.)
- Ezeknek a téglalapoknak megadhatjuk a méretüket pixel (px), centiméter (cm) vagy % formában (a százalék mindig az ősosztály méreteihez képest értendő)
- width:?px; vagy width:?%; vagy width:?cm;
- height:?px; vagy height:?%; vagy height:?cm;

CSS doboz modell

- Ha létrehoztunk egy dobozelemet, akkor annak megadhatjuk a külső távolságát más elemektől (margin:?px;), a keret vastagságát (border-width:?px), a szövegünk (tartalom, content) kerettől való belső távolságát (padding:?px;)
- Ezeket alapértelmezetten nem veszi bele a doboz méretébe (tehát nagyobb lesz a dobozunk mint amit megadtunk)(ha bele szeretnénk számíttatni, akkor használjuk a: box-sizing: border-box; tulajdonságot)



CSS Layout

- A float:left;/right; tulajdonság segítségével egy dobozt, vagy képet automatikusan körbefuttathatjuk szöveggel
- clear:left;/right; tulajdonsággal megakadályozhatjuk a float érvényesülését
- Ezek helyett jobban járunk, ha a ‘display: inline-block;’ tulajdonságot használjuk (nem kell clear-t használni hozzá)
- overflow tulajdonság segítségével beállíthatjuk, hogy a szövegünk „kifolyhat”-e a dobozunkból, értékek: hidden (ami kifolyik az eltűnik), scroll (ami kifolyna, azt egérgörgővel felhozhatjuk), auto (automatikusan eldönti, hogy nézne ki jól)
- z-index tulajdonság segítségével indexszel megadhatjuk, melyik elem melyik felé kerüljön (ehhez a position tulajdonságot meg kell adni! (static nem lehet))
- margin:auto tulajdonság segítségével középre lehet zárni a dobozunkat az ősosztályán belül (automatikusan úgy állítja be a margin-left és margin-right –ot, hogy középre kerüljön, ekkor megadjuk pl.: width:60%;)

HTML iframe tag

- Segítségével beágyazhatunk más weblapokat, akár youtube videókat az oldalunkra
- Használata:
  - `<iframe width="560" height="315" src="https://www.youtube.com/embed/FrZRIW87eWI" frameborder="0" allowfullscreen>A böngésződ nem támogatja az iframe beágyazást</iframe>`
- Youtube videók alatt: megosztás (Share) -> beágyazás (Embed) , innen másolhatjuk ki a videó HTML beágyazási kódját
- http://www.w3schools.com/tags/tag_iframe.asp

Tipp:
- Sok weblapot nem lehet így beágyazni
- Néha nem elég frissíteni az oldalunkat amikor változtattunk valamit rajta, hanem be kell zárni és meg kell nyitni újra

Kommunikáció az interneten

- Minden internetre kötött eszköznek van egy külső (external) IP címe, ez a cím egyedi mindenkinek, ezért ezzel címmel lehet gépeket elérni az interneten
- Az otthoni hálózatunk általában úgy néz ki, hogy az eszköz ami a külső IP címünket kapja, az a routerünk (hálózatelosztónk, amire wifi-vel vagy lan kábellel csatlakozunk). A router szerepe az, hogy a rácsatlakozott gépeknek internet hozzáférést biztosítson és így létrehoz egy lokális (local) hálózatot (network). A router minden rácsatlakozott gépnek kioszt egy lokális IP címet (local IP address), azért, hogy az otthoni hálózatunk gépei is meg legyenek címezve és lehessen rájuk hivatkozni.
- A routerünk beállításainak segítségével egy lokális gépen (pl. a saját számítógépünkön) futtatott bármilyen szerver elérhető lesz az interneten
- Szervert futtathatunk saját gépünkön, vagy rendelhetünk 0-24-es szervergépet (webhosting) általában havidíjért

Portok az interneten

- Ha egy szervergépet el akarunk érni, akkor mindig meg kell adni a kommunikáció (protokoll) típusát, azért hogy egyértelmű legyen a kapcsolat a kliens és szerver között
- Ha egy weboldalt meglátogatunk, akkor a protokoll típusa http/https (HyperText Transfer Protocol) lesz és így a böngészőnk egyértelműen fogja tudni, hogy HTML nyelvet kell értelmeznie
- Felmerül a kérdés, hogy mi van akkor, ha többféle szervert is szeretnénk futtatni, különböző protokollokkal (pl. FTP szervert, Minecraft szervert… stb.). Mindegyik szervertípushoz rendelünk egy egyedi csatorna azonosítót, portot, azért hogy a különböző kommunikációk egyedik legyenek és ne keveredjenek össze. Így ha egy konkrét szervertípusra akarunk csatlakozni, akkor nem csak a külső IP címre van szükségünk, hanem a kommunikáció csatorna portjára is.
- Minden szervertípushoz tartozik egy értelmező kliens, amit a felhasználó használ (pl.: böngésző, játékkliens…), amiben megadjuk az elérési címet és a kommunikáció (protokoll) típusát (portját, csatornáját)

A port

- A portot megadhatjuk külső IP cím után kettősponttal (pl.: 12.34.567.890:40 , itt a port 40)
- Vagy ha megadtuk a protokollt pl a böngészőnkben (pl.: http://...), akkor ez határozza meg a portot
  - pl.: HTTP protokoll portja: 80
  - FTP protokoll portja: 20,21
  - POP3 email protokoll portja: 110
- A router feladata az, hogy a külső internetről érkező kéréseket a megfelelő routerre lokálisan csatlakozott szervergépre irányítsa és létrehozza a kapcsolatot szerver és kliens között
- Tehát, ha egy routeres lokális hálózaton belül futtatunk egy szervert, akkor az ahhoz tartozó kommunikációs portokat engedélyezni kell („ki kell engedni”) a router beállításában (port továbbításra van szükség (port forwarding)

Hamachi helyett…

- Ha bármilyen szervert futtatunk a saját gépünkön, akkor tudjuk meg a kommunikáció port számát (port number) és nézzünk utána, hogy a saját routerünk beállításaiban, hogy lehet továbbítani ezt a portot (port forwarding)
- Az irányelv az szokott lenni, hogy megtudjuk a lokális szerverünk (ez lehet pl. a saját gépünk) lokális IP címét (amit a router osztott ki)(cmd-ben ipconfig)
  http://www.helpfulpctools.com/HowToCheckYourLocalIP.php
  Majd beírjuk a böngészőnkbe a routerünk lokális IP címét, ami általában 192.168.0.1 vagy 192.168.1.1 szokott lenni. Ott beírjuk az alapértelmezett felhasználónevet/jelszavat, ami általában admin/admin. Így elérhetjük a routerünk beállításait, itt meg kell keresni a „port forwarding” részt és ott megadjuk a szerver lokális IP-jét, majd a portot.
  Így mások el tudják érni majd a szerverünket az internetről Hamachi nélkül is.
- Segítség: http://portforward.com/
- Tehát ha pl. azt szeretnénk, hogy a webszerverünket elérjék az interneten, akkor a 80-as portot kell továbbítani ellenőrzés: http://www.canyouseeme.org/

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.