

Debreceni Egyetem

Informatikai Kar

Weblapkészítés ingyenes eszközök felhasználásával

Témavezető:

Pánovics János

egyetemi tanársegéd

Készítette:

Czagány Ádám

programozó matematikus

Debrecen

2009

Tartalomjegyzék

1. Bevezetés.....	4
2. Felhasznált technológiák	6
2.1 HTML, XHTML.....	6
2.2 CSS.....	8
2.3 JavaScript.....	10
2.4 PHP.....	11
2.5 MySQL	13
3. Felhasznált eszközök	15
3.1 Notepad++	15
3.2 GIMP.....	17
3.3 IrfanView	19
3.4 MySQL Workbench.....	21
3.5 Mozilla Firefox.....	23
3.6 Windows Internet Explorer	25
3.7 Google Chrome.....	28
3.8 W3C Validator, mint munkaeszköz	29
4. Munkafolyamat.....	32
4.1 Mit várunk el az oldaltól?	32
4.1.1 Design és használhatóság ellentéte	33
4.2 Az oldal megtervezése.....	34
4.2.1 Adatbázis tervezés	34
4.2.2 Az oldal struktúrájának megtervezése.....	37
4.2.3 Grafikai tervezés	38
4.3 A megvalósítás.....	41
4.3.1 Grafika és design	41
4.3.2 Oldalfelépítés – HTML és CSS.....	43
4.3.3 Az adatbázisváz	46
4.3.4 Dinamikus tartalmak.....	48

4.4 Tesztelés	50
4.4.1 Megjelenés	50
4.4.2 Navigáció.....	51
4.4.3 Funkció	52
4.5 Végeredmény	52
4.5.1 A weblap rövid bemutatása.....	53
4.5.2 A weblap adminisztrációs felülete.....	59
5. Konklúzió	60
6. Köszönetnyilvánítás.....	61
7. Irodalomjegyzék.....	62

1. Bevezetés

A mai világban, amikor minden irányból milliónyi adat és információ áramlik hozzánk, a mindennapjaink természetes részévé vált, hogy használjuk az internet nyújtotta szolgáltatásokat. Az internetes médiumok, portálok, híroldalak száma folyamatosan növekszik, amik egyre komplexebb és többféle szolgáltatást nyújtanak nekünk. Lehetőségünk van, hogy szinte minden információt megtaláljunk online tartalom formájában. Egészen a buszmenetrendtől, a friss híreken át, a boltok nyitvatartási idején keresztül a TV műsorig, és lehetne még sorolni, amíg csak szeretnénk. Az információkon túl akár már pénzügyeinket is intézhetjük ilyen módon, vagy akár be is vásárolhatunk. Nem túlzás azt állítani, hogy az internet életünk szerves része lett, és ez a folyamat egyre erősödik.

Ami viszont kevés embernek jut eszébe, az az, hogy minden egyes oldal amit meglátogat böngészés közben, az kisebb vagy nagyobb programozói munkát igényelt valakitől. Mindegyiket el kellett készítenie valakinek vagy valakiknek. Az egyszerűen, szinte bárki által elkészíthető apró kis bemutatkozó lapoktól az óriás cégek által készített, technológiában bővelkedő lapokig mind-mind megszületett valahogyan. Egy kezdő ötlet után, egy bizonyos folyamaton keresztül elérkezett ahhoz a ponthoz, amikor publikálható formát öltött, feltöltődött tartalommal, és használható lett. Ha ezt a folyamatot elég távolról, és elég absztrakt módon szemléljük, akkor tulajdonképpen kijelenthető, hogy mind ugyanazon az elven – legalábbis nagyon hasonló elven, vagy elv mentén – jutott el a végső felhasználható formájába. A munkamódszerek lehetnek igen eltérőek, a projektek mérete úgyszintén, de azok a folyamatok ahogyan ezek a termékek készülnek rengeteg helyen találkoznak, és vannak általános szabályok, amiket minden oldal készültkor felfedezhetünk. Nevezhetjük akár irányelveknek is őket.

A dolgozat témája a mellékelt programhoz használt technológiákon és eszközökön keresztül általánosan bemutatni a fent vázolt folyamatot, még inkább a

folyamatokból összeálló egész képet úgy, hogy azok külön-külön is értelmezhetőek legyenek, mégis logikusan kapcsolódjanak egymáshoz. Kihhasználva, hogy a moduláris felépítés könnyebb és jobb megértést eredményez minden téren, főleg az összetett, több rétegből álló dolgok esetén, ezen módszerrel haladva kerül bemutatásra és elemzésre a választott téma.

2. Felhasznált technológiák

A dolgozatban tárgyalt technológiák mellett több más technológia is elterjedt, és széleskörűen alkalmazzák is, azonban a területi korlátoltság miatt nem törekszem a teljességre, mivel egy-egy kiválasztott témában nyomtatott és online média formájában is több mint elégséges referencia áll rendelkezésre. Inkább szeretnék egyfajta útmutatást nyújtani a felhasznált technológiákban arról, hogy milyen előnyös és praktikus alkalmazhatósági körök van, esetlegesen milyen hiányosságokkal rendelkeznek, vagy mik azok a felhasználási területek, ahova kevésbé ajánlott választások lehetnek.

2.1 HTML, XHTML

A HTML betűszó az angol Hyper Text Markup Language szavakból áll össze, ami magyarra így fordítható: hiperszöveges jelölőnyelv. Egy leíró nyelv, amit célzottan weboldalak készítéséhez fejlesztettek ki és mára már internetes szabvánnyá nőtte ki magát. Nem túlzás azt állítani, hogy a web alapjáról beszélünk, amikor a HTML-t emlegetjük. Aki bármennyire is foglalkozott már az internettel az elsők között – ha nem a legelső helyen – találkozott a szabvánnyal, és az általa támasztott kritériumokkal.

Jelenleg a 4.01^[1] verzióval tart, de a W3C (azaz a World Wide Web Consortium)^[2] ajánlása 2001. május 31. óta az XHTML 1.1^[3] (Extensible Hyper Text Markup Language) szabvány már. Az XHTML és a HTML között a legjelentősebb különbség, hogy míg a HTML SGML^[4] (Standard Generalized Markup Language) szabvány szerint íródott, így maga is tulajdonképpen egy SGML dokumentum, addig

^[1] HTML 4.01 szabvány teljes specifikációja: <http://www.w3.org/TR/html4/>

^[2] további információ: <http://www.w3.org/>

^[3] XHTML 1.1 szabvány teljes specifikációja: <http://www.w3.org/TR/xhtml11/>

^[4] SGML további információk: <http://www.w3.org/MarkUp/SGML/>

az XHTML már az XML^[5] (Extensible Markup Language) szabvány alapján definiált leíró nyelv, ami szintén SGML szabvány szerinti, de szigorúbb követelményeknek kell, hogy megfeleljen. Egy XHTML dokumentumnak jól formázottnak kell lennie, szigorúbb formai szabványokkal, viszont cserébe egy szabványos XML elemző (angolul: parser) is fel tudja dolgozni a dokumentumot a HTML-el ellentétben, amihez egy sokkal inkább engedékeny, specifikusan a HTML-hez készített elemző szükséges. Az oldalak szabványosításához a W3C készített egy ellenőrző eszközt, amivel tesztelhetjük lapjainkat, hogy maradéktalanul megfelelnek-e a W3C ajánlásának: <http://validator.w3.org/>. A validátorról a későbbiekben még bővebben esik majd szó.

```
<!doctype html>
<html>
  <head>
    <title>Page title</title>
  </head>
  <body>
    <p>This is a basic HTML page.</p>
  </body>
</html>
```

Egy alapvető HTML kód felépítése.

A HTML önmagában igazából csak alapfokú funkciókat kínál, mind a megjelenést, mind a tartalmat alapul véve. Inkább hasonlítható egy kellemesen formázott szöveghez, amit az alap eszközökkel egyszerűen létrehozhatunk, mint a

^[5] a W3C 1998 február 10-i ajánlása az XML-ről: <http://www.xml.com/axml/testaxml.htm>; valamint a jelenlegi verzió <http://www.w3.org/TR/xml/>

ma megszokott dinamikusan változó és változtatható, erősen felhasználóbarát weblapokhoz. Persze a HTML mély ismeretével és kellő gyakorlattal, lehetséges sokkal összetettebb és jobban formázott megjelenést létrehozni, de az egyszerűbb használhatóság és könnyebb szerkeszthetőség miatt inkább más technológiákkal együtt szokás már használni.

2.2 CSS

A CSS az angol Cascading Style Sheets szavak rövidítése, magyarul kb. annyit tesz, hogy átfedő stílus lapok. Kizárólag stílus leíró nyelv, tehát csak a weblapok megjelenését befolyásolja, azok tartalmára semmilyen hatással nincs.

Maguk a stíluslapok már a korai 1990-es évektől jelen vannak, de eleinte csak a böngészők beépített stíluslapjairól beszélhetünk, amik definiálták, hogy egy adott oldal hogyan jelenjen meg. Ekkor még a HTML szabvány is meglehetősen szegényes eszköztárral bírt az oldalak megjelenítésének formázásában. Később ezt felismerve és a webfejlesztők, valamint az internet felhasználói igényeinek növekedése miatt került kifejlesztésre több stílus leíró nyelv is, azonban a CSS az egyetlen, ami végül megkapta a W3C ajánlását, 1996 decemberében. A jelenlegi ajánlás a CSS2^[6], emellett a CSS2.1^[7] is készülőben van, ami most még csak ajánlásra jelölt fázisnál tart, valamint ezekkel párhuzamosan folyik a CSS3 fejlesztése is.

Ahogy a HTML esetében is, a CSS-hez is rendelkezésre áll a megfelelő validáló szolgáltatás a W3C jóvoltából, ami nagyban segíthet kiküszöbölni a megjelenésbeli problémákat, vagy a nem megfelelően, azaz nem szabványosan elkészített kódban lévő hibákat: <http://jigsaw.w3.org/css-validator/>.

^[6] a CSS2 szabvány teljes specifikációja: <http://www.w3.org/TR/2008/REC-CSS2-20080411/>

^[7] a CSS2.1 szabvány tervezett specifikációja: <http://www.w3.org/TR/CSS21/>

A stíluslapok használatával lehetőség nyílik arra, hogy teljesen külön kezeljük a dokumentumok tartalmát, és megjelenését. Míg kizárólag HTML használatával egy dokumentumban kell definiálnunk a tartalmat és annak formáját, addig a CSS-el képesek vagyunk arra, hogy vagy egy szeparált részen definiáljuk a stílusokat az egyes elemekhez, vagy külön fájlban tegyük ezt és azt csatoljuk a HTML forráskódjába. Ezek egyenes következménye, hogy több lap stílusát, vagy akár egy teljes webhelyét is definiálhatjuk egyetlen CSS fájlban, ami így jelentősen leegyszerűsíti és átláthatóbbá teszi a webhely szerkezetét és forráskódját is. Egyszerűbbé válik a megjelenítési stílusok kezelése, utólagos frissítése, valamint lehetőségünk lesz különböző stílusokat tárolni és adott esetben felhasználni azt a webhely különböző funkciók megvalósítására. Gondoljunk például az olvasást elősegítő nagyított, magas kontrasztú megjelenítésre, vagy akár egy nyomtatóbarát formára, ahol a weblaphoz tartozó képek, és formai elemek nem szükségesek ahhoz, hogy a tartalmat használható módon papírra vessük.

```
p {  
  font-family: "Garamond", serif;  
}  
  
h2 {  
  font-size: 110%;  
  color: red;  
  background-color: white;  
}  
  
.megjegyzes {  
  color: red;  
  background-color: yellow;  
  font-weight: bold;  
}  
  
p#bevezetes {  
  border: 3px dashed black;  
}  
  
a:hover {  
  color: yellow;  
  background-color: navy;  
}
```

Egy példa CSS file felépítése.

2.3 JavaScript

A JavaScript hovatartozása körül vannak nézet különbségek a különböző felhasználók körében. Egyesek úgy tekintenek rá, mint egy programozási nyelvre, akár csak például a nagyon hasonló nevű Javára, viszont a fő felhasználási területét alapul véve igazából egy szkript nyelv. A nyelv objektum-orientált és dinamikus típuskezeléssel rendelkezik. Leginkább kliens oldali alkalmazások megírására használják weblapok készítésekor. A jelenlegi legfrissebb verziószáma a JavaScript 1.8.1, viszont az érvényes szabvány az ECMA^[8]-262 Edition 3, ami a JavaScript 1.5-nek felel meg.

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

<html>

  <head>
    <title>simple page</title>
  </head>

  <body>

    <script type="application/javascript">
      document.write('Hello World!');
    </script>

    <noscript>
      <p>Your browser either does not support JavaScript,
or you have JavaScript turned off.</p>
    </noscript>

  </body>

</html>
```

Egy egyszerű JavaScript példa program egy HTML oldalba ágyazva.

^[8] ECMA: Európai Számítógépgyártók Szövetsége (European Computer Manufacturers Association)
<http://www.ecma-international.org/>

A JavaScript megjelenésekor igen közkedvelt eszköz volt és sokan használták, de egy idő után úgy tűnt, hogy kezd feledésbe merülni. Azonban az AJAX^[9] elterjedésével és egyre inkább divatossá válásával újra fókuszba került, és újabb rajongókat szerzett magának a kiszélesedett felhasználhatósági területével.

Hasonlóan a CSS-hez ez is beágyazható az oldal HTML forrásába, de akár hivatkozásként is csatolhatjuk a lapunkhoz, mint egy különálló fájlt, ekkor ***.js** kiterjesztéssel szokás ellátni. A nyelv képes a lap forrásából létrejövő DOM^[10] teljes manipulálására, ami maga is objektumokból épül fel, így egy igen hatékony eszközről beszélünk, amikor a weblapokkal kapcsolatban van szó a JavaScriptről.

2.4 PHP

A PHP: Hypertext Preprocessor^[11] nyílt forráskódú szkript nyelv, de a JavaScripttel ellentétben ez nem kliens oldali alkalmazás, hanem szerver oldali. Az utasítások HTML-be szintén tetszőlegesen beágyazhatók, de magát a HTML-t a PHP csak, mint keretet használja. Az oldal elérésekor a PHP utasítások kerülnek először feldolgozásra a kiszolgáló által és a kliens oldalon csak a végeredményt látjuk már, tehát a forráskód rejtve marad.

A PHP jelenlegi legfrissebb verziója az 5.3.0, amit 2009. június 30-án adtak ki. Maga a nyelv sok hasonlóságot mutat a C programozási nyelvvel, és szintaktikáját tekintve is ezt, illetve más magas szintű programozási nyelveket követ. Az átláthatóságot, valamint a HTML kódba való beágyazást segíti, hogy a PHP elemzője csak a határoló jelek közti kódot dolgozza fel, a többit egész egyszerűen figyelmen kívül hagyja. A szabványos határoló jelek a következők: **<?php ?>**, valamint ennek rövidített változatai, amik azonban a **php.ini** fájlból, azaz a PHP

^[9] AJAX: Asynchronous JavaScript And XML, ami tulajdonképpen egy XML alapú HTTP kérést valósít meg, anélkül, hogy a böngészőkben oldalt kellene frissíteni

^[10] DOM: Document Object Model; <http://www.w3.org/DOM/>

^[11] a PHP hivatalos weboldala: <http://php.net/>

konfigurációs fájljából letilthatók, ezért nem minden szerveren működnek. Ilyenek pl: `<? ?>;` `<?= ?>.`

```
<html>

  <head>
    <title>PHP test page</title>
  </head>

  <body>

    <?php
      echo "<p>";
      echo "Hello World!";
      echo "</p>";
    ?>

  </body>

</html>
```

Egy egyszerű példa beágyazott PHP utasításokra.

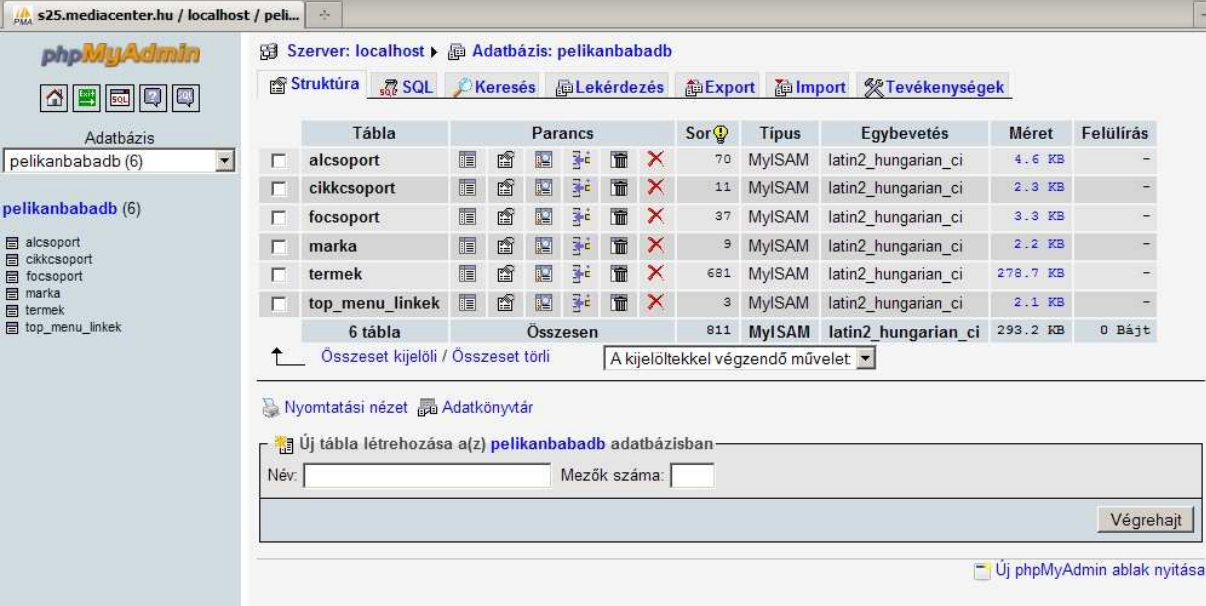
A nyelv kiválóan használható űrlapok feldolgozására, adatbázis műveletek végrehajtására, vagy ismétlődő feladatokhoz tartozó szöveg generálására. Ilyen lehet akár egy fényképalbum, ahol egyszerre több képet kell a képernyőn megjeleníteni, és egy egyszerű ciklusutasítással megkíméljük magunkat a hosszú és unalmas kód gépelésétől; mindezt elvégzi helyettünk a PHP elemzője, majd a végeredményt átadva a böngészőnek megjelenik a képernyőn.

A nyelvnek van objektum-orientált része is, azonban használható pusztán funkcionális nyelvként is. A különböző oldalakon több nyílt kódú, ingyenes kiegészítést találhatunk a PHP-hoz, de az alap disztribúció is tartalmaz kellő számú könyvtárat, amik megkönnyíthetik a munkánkat. A teljes függvénykönyvtár online elérhető – egy része már magyarul is – a következő oldalon: <http://hu.php.net/manual/en/funcref.php>.

2.5 MySQL

A MySQL egy SQL^[12] alapú relációs adatbázis kezelő. A hivatalos honlap^[13] szerint „A világ legnépszerűbb nyílt forrású adatbázisa”. Ez valószínűleg nagyban köszönhető annak, hogy a nagy népszerűségnek örvendő LAMP^[14] csomag része a MySQL, amivel egyszerűen, gyorsan, és költség hatékonyan lehet megoldani dinamikus szerverszolgáltatásokat.

Jelenleg az 5.1.40 verziószámánál tart a stabil kiadás az adatbázis kezelőből, amit éppen nemrégiben adtak ki, 2009. október 23-án. A rendszer több felhasználós, és rengeteg különböző platformon működik. Mivel SQL alapú a nyelv, így a használata nem okoz problémát senkinek, aki valaha foglalkozott már adatbázis kezeléssel. Nem csak parancssori üzemmódban működik, vannak hozzá grafikus kezelőfelületek is, mint például Linuxon a MySQL Administrator, vagy a sokak által kedvelt webes felületű phpMyAdmin.



The screenshot shows the phpMyAdmin interface in a web browser. The left sidebar displays the database 'pelikanbabadb' with its tables: alcsoport, cikksoport, focsoport, marka, termék, and top_menu_linkek. The main panel shows the 'Struktúra' (Structure) tab for the 'pelikanbabadb' database. It lists 6 tables with their respective columns, data types, and sizes. Below the table list, there are options to 'Összeset kijelölő' (Select all) and 'Összeset törli' (Delete all), and a dropdown for 'A kijelöltekkel végzendő művelet' (Operation to be performed on the selected records). At the bottom, there is a section for 'Új tábla létrehozása a(z) pelikanbabadb adatbázisban' (Create new table in the pelikanbabadb database) with fields for 'Név' (Name) and 'Mezők száma' (Number of fields), and a 'Végrehajt' (Execute) button.

Tábla	Parancs	Sor	Típus	Egybevetés	Méret	Felülírás
☐ alcsoport		70	MyISAM	latin2_hungarian_ci	4.6 KB	-
☐ cikksoport		11	MyISAM	latin2_hungarian_ci	2.3 KB	-
☐ focsoport		37	MyISAM	latin2_hungarian_ci	3.3 KB	-
☐ marka		9	MyISAM	latin2_hungarian_ci	2.2 KB	-
☐ termék		681	MyISAM	latin2_hungarian_ci	278.7 KB	-
☐ top_menu_linkek		3	MyISAM	latin2_hungarian_ci	2.1 KB	-
6 tábla		Összesen	811	MyISAM	latin2_hungarian_ci	293.2 KB
						0 Bájtt

↑ Összeset kijelölő / Összeset törli A kijelöltekkel végzendő művelet

Nyomatási nézet Adatkönyvtár

Új tábla létrehozása a(z) pelikanbabadb adatbázisban

Név: Mezők száma:

Végrehajt

Új phpMyAdmin ablak nyitása

A phpMyAdmin felhasználói felülete.

[12] SQL: Structured Query Language, azaz strukturált lekérdező nyelv

[13] MySQL hivatalos weblapja: <http://www.mysql.com/>

[14] LAMP: Linux, Apache, MySQL, PHP

Felhasználási területe teljesen megegyezik más adatbázis kezelő rendszerekkel, de néhány konkrét példa lehet akár: látogatói adatok nyilvántartása, felhasználó menedzsment, webáruházak árukészlet nyilvántartása, hírportálok adatbázisa, vagy bármely egyéb dinamikus oldal háttérében dolgozhat MySQL adatbázis motor.

3. Felhasznált eszközök

Az oldal készítése során nem csak a felhasznált technológiák, hanem a fejlesztő eszközök között is úgy válogattam, hogy gyakorlatilag mind ingyenes, esetleg nyílt forráskódú legyen. Ez nagy segítséget nyújthat a diákoknak, akik szeretnék kipróbálni magukat az online tartalmak készítésében, de nem akarnak – esetlegesen lehetőségük sincsen – drága, liszenszelt szoftverekkel dolgozni. Nem elhanyagolható azonban az a tényező sem, amikor akár egy nagyobb vállalat is amellet dönt, hogy nyílt forrású szoftverekre állnak át ahol csak lehetséges és ezzel rengeteg költséget tudnak megtakarítani. Manapság már nem jellemző ami a kezdeti nyílt forrású szoftverek ellen szólt, vagyis a kiforratlanság, esetleges problémás működés, vagy az alacsonyabb megbízhatóság. Mostanra nyugodtan kijelenthetjük, hogy a nyílt forrású szoftverek is elérték a kívánt színvonalat, sőt van ahol többlétszolgáltatásokat is tudnak nyújtani, a zárt kódú programokkal szemben.

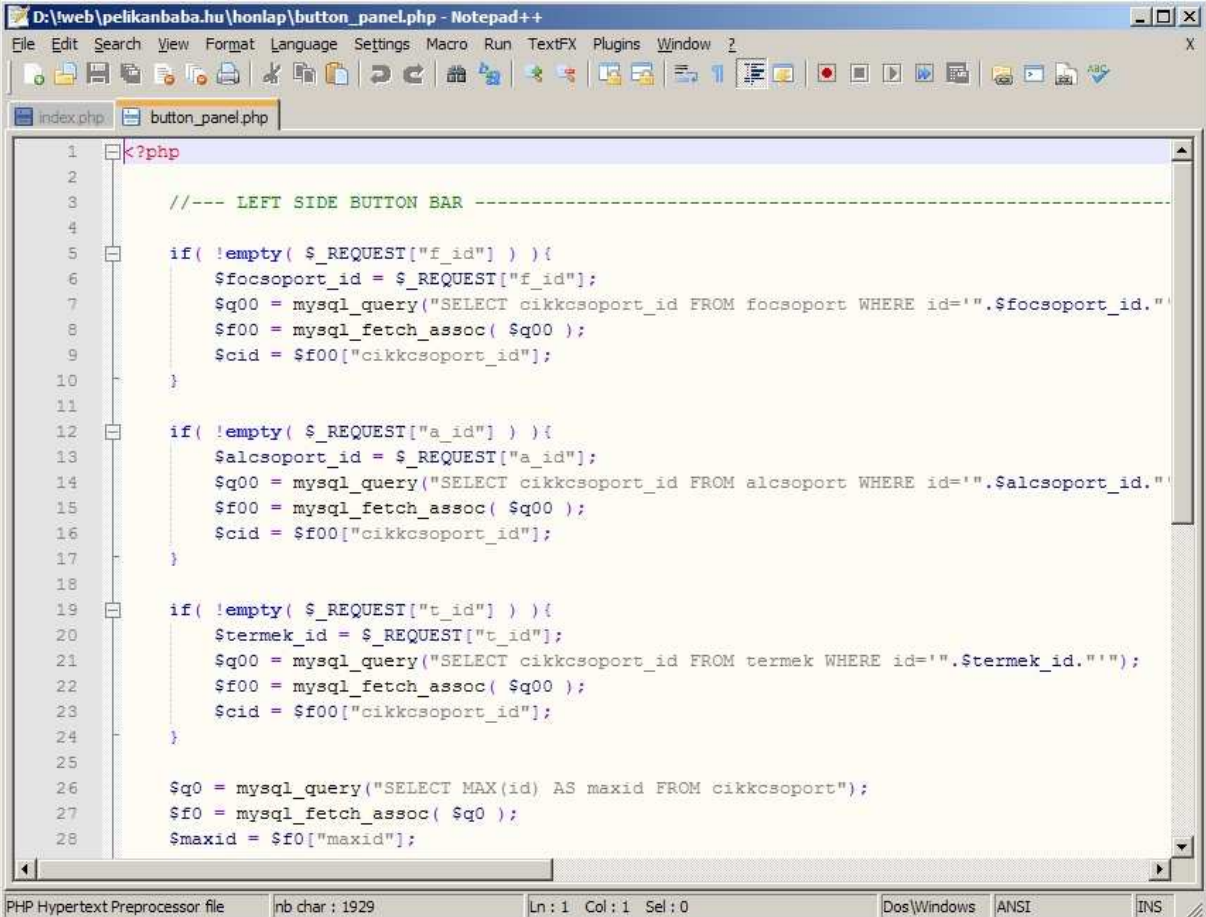
3.1 Notepad++

A Notepad++^[15] egy ingyenes forráskód szerkesztő program. Arra a kérdésre pedig, hogy miért pont ezt használom több nyomós indokot is fel tudok sorolni. Az én elvárásaimnak szinte maradéktalanul megfelel, amit nagyon kevés program mondhat el magáról. Általában minden szoftverrel meg kell tanulni bánni és rá kell jönni, hogy a hiányzó, vagy nem tetsző részeket hogyan hidaljuk át, vagyis hogyan használjuk úgy, hogy az nekünk kézre essen. Aki programozott már valaha az tudja, hogy egy megfelelő keretrendszer igazán meg tudja könnyíteni az ember munkáját, de ugyanúgy megnehezíti egy nem megfelelően választott, vagy nem kézre álló szoftver.

A legfontosabb tulajdonságai közt muszáj szót ejtenem arról, hogy rengeteg programozási, és szkript nyelvet ismer, de ha valami még így is hiányozna a

^[15] a program hivatalos honlapja: <http://notepad-plus.sourceforge.net>

felhasználónak, akkor lehetőséget nyújt arra is, hogy saját nyelvet definiáljunk, amikhez többféle beállítási lehetőséget is kínál. Rendelkezik a kód összezárásának lehetőségével, pl. függvényeket, eljárásokat, vagy blokkokat is összezárhatunk, így csak a fejlécük látszik és a kódunk sokkal átláthatóbb lesz. Színezi a kódot nekünk, így egyszerűen észrevehetjük az elütéseket, ha nem megfelelő függvény nevet használunk, vagy elfelejtjük lezárni egy string végét. Automatikus kódkiegészítéssel is rendelkezik, ami szintén hasznos lehet ha még csak ismerkedünk egy programozási nyelvvel, vagy egyszerűen csak kevesebbet akarunk gépelni.



```
1 <?php
2
3 //--- LEFT SIDE BUTTON BAR -----
4
5 if( !empty( $_REQUEST["f_id"] ) ){
6     $focszoport_id = $_REQUEST["f_id"];
7     $q00 = mysql_query("SELECT cikkszoport_id FROM focszoport WHERE id='".$focszoport_id."");
8     $f00 = mysql_fetch_assoc( $q00 );
9     $scid = $f00["cikkszoport_id"];
10 }
11
12 if( !empty( $_REQUEST["a_id"] ) ){
13     $alcszoport_id = $_REQUEST["a_id"];
14     $q00 = mysql_query("SELECT cikkszoport_id FROM alcszoport WHERE id='".$alcszoport_id."");
15     $f00 = mysql_fetch_assoc( $q00 );
16     $scid = $f00["cikkszoport_id"];
17 }
18
19 if( !empty( $_REQUEST["t_id"] ) ){
20     $termek_id = $_REQUEST["t_id"];
21     $q00 = mysql_query("SELECT cikkszoport_id FROM termek WHERE id='".$termek_id."'");
22     $f00 = mysql_fetch_assoc( $q00 );
23     $scid = $f00["cikkszoport_id"];
24 }
25
26 $q0 = mysql_query("SELECT MAX(id) AS maxid FROM cikkszoport");
27 $f0 = mysql_fetch_assoc( $q0 );
28 $maxid = $f0["maxid"];
```

A Notepad++ felhasználói felülete, egy PHP kódrészlettel.

Többféle karakterkódolást támogat; többek közt a szokásos ANSI-t, és UTF8-at is, valamint tud konvertálni Windows, Unix, és Mac formátumra is. Fejlett keresés és csere parancsok vannak benne, amik támogatják a reguláris kifejezéseket^[16] is. Képes makrókat felvenni, tárolni, és futtatni akár egymás után többször is, amikkel automatizált folyamatokat hozhatunk létre, megtakarítva sok-sok időt és energiát gépiesen megoldható feladatok elvégzésénél. Valamint tartalmaz olyan beépített kényelmi funkciókat, mint például a speciális karakterek escape karakterré való alakítása (pl: „tab” -> „\t”), vagy az aposztrófok cseréje idézőjelekké és fordítva. Ami számomra a legfontosabb tényezők közt szerepelnek azok a hordozhatóság, mivel nem feltétlenül igényel telepítést; a méret, ugyanis az egész pár megabyte-ot foglal; és a gyorsaság, mivel szinte azonnal indul, mert minimális az erőforrás igénye és egyszerű a kezelőfelülete.

Léteznek egyéb közkedvelt eszközök, de ennyire jól összerakott, ilyen összetett szoftvert nagyon sok esetben egyetlen, esetleg néhány programozási nyelvre hegyeznek ki, plusz sokkal nagyobb méretben képesek csak megvalósítani, ami együtt jár a program lassulásával is, valamint ezek többségében fizetős fejlesztő eszközök.

3.2 GIMP

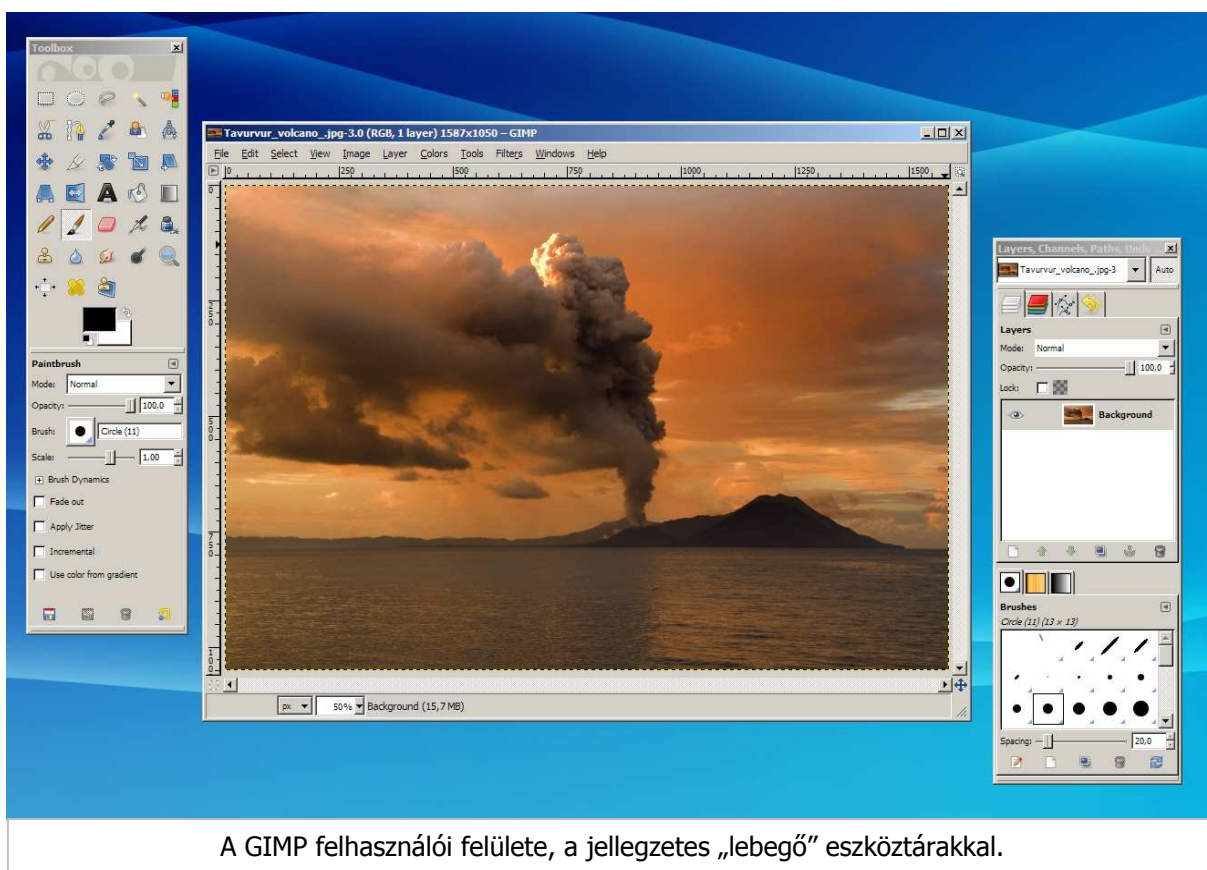
A GIMP^[17], azaz a GNU Image Manipulation Program, egy szabadon használható, és szabadon terjeszthető képszerkesztő program. Több platformra is elérhető, azaz Linuxra (néhányik disztribúció eleve tartalmazza a GIMP csomagot), Windowsra, és Mac-re egyaránt. Sokak által közkedvelt képszerkesztő és manipuláló program. Tulajdonságaiban sokban hasonlít az előbb tárgyalt Notepad++-ra, azaz ez is kellően kisméretű, és létezik belőle hibátlanul működő hordozható verzió is.

^[16] további információt itt található a reguláris kifejezésről:

http://hu.wikipedia.org/wiki/Reguláris_kifejezés

^[17] a szoftver hivatalos oldala: <http://www.gimp.org/>

Rendelkezik a hasonló programok alapvető eszközeivel, amikkel akár egy meglévő képet, fotót módosíthatunk vagy szabhatunk át kedvünk szerint, de készíthetünk üres vászonra egy teljesen új képet is. Fontos, hogy a nagyobb és profibbnak nevezett eszközökhöz hasonlóan szintén támogatja a rétegek használatát, ami nagyon sok lehetőséget nyújt a felhasználóknak, és sok esetben képes jelentősen megkönnyíteni a munkát. Az alapvető funkciókon túl – mint az átméretezés, levágás, torzítás, stb. – rengeteg szűrővel és egyéb eszközzel rendelkezik. Hogy egy konkrét dolgot említsek ami sokak személyes kedvence, az az, hogy bármilyen képből egyetlen kattintással képes olyan képet létrehozni, ami mozaik elrendezést használva teljesen jól illeszkedő mintát ad ki. Ez nagyon hasznos lehet akár weboldalak háttereinek készítésekor is például.



A GIMP felhasználói felülete, a jellegzetes „lebegő” eszköztárakkal.

Ha esetleg nem elégednénk meg azzal, ami a programban eleve megtalálható, akkor az interneten rengeteg ingyenesen elérhető kiegészítőt találhatunk – egészen az ecsetmintáktól, az egyedi szűrőig, és egyéb eszközökig.

A legtöbb gyakran használt grafikus fájl formátumot ismeri és képes kezelni. Használja az alfa csatornát, azaz képes átlátszóságot kezelni amit leggyakrabban PNG kiterjesztésű képekben láthatunk. Egy kevés vektorgrafikus támogatás is van benne, de igazán a szokványos ún. bittérképes grafikai szerkesztésre van optimalizálva. A profi eszközökhöz képest (pl: Adobe Photoshop) egy hátránya van ami nagy képek szerkesztésekor jelentkezik, az pedig a sebessége. Egy bizonyos pont után érezhetően lassabb a munkavégzés vele ami esetleg zavaró lehet, de ne feledjük, hogy egy ingyenes szoftverről van szó, ráadásul weblapok készítésénél csak elvétve fordul elő, hogy igazán nagyméretű képeket kellene szerkesztenünk, tehát ahhoz tulajdonképpen ideális partner lehet.

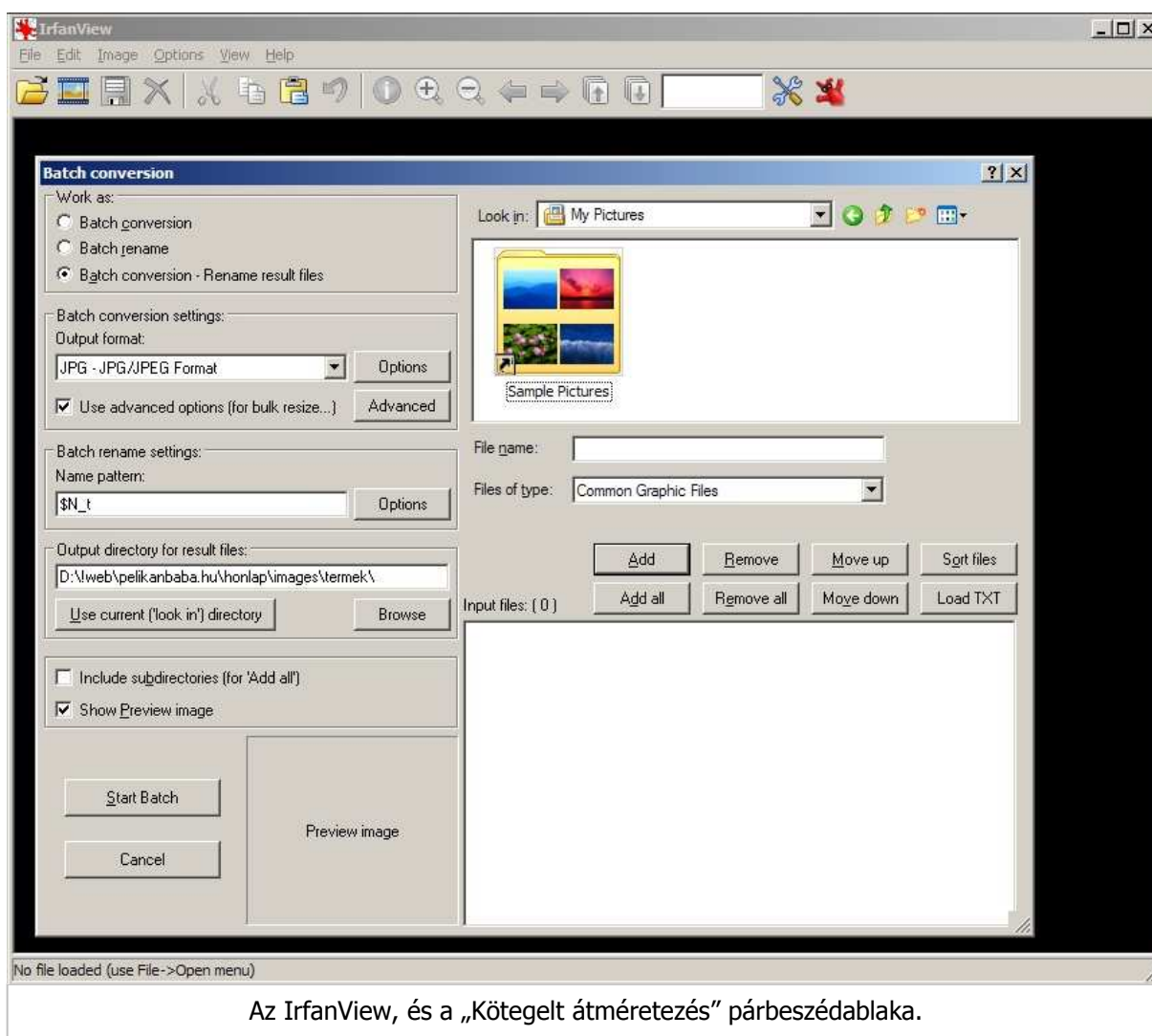
3.3 IrfanView

Az IrfanView^[18], vagy sokak által csak Iview-nak nevezett program elsősorban képnézegető, de rendelkezik képmanipuláló funkciókkal is. Funkcionalitásában messze elmarad ugyan pl. a GIMP-től, de ami miatt mégis szeretnék szót ejteni róla, az egy nagyon hasznos eszköze. Amikor több esetleg több száz képet, vagy fényképet kellene rendszerezni, átnevezni, esetleg még méretezni is, akkor tud igazán hasznos lenni ez a szintén aprócska program.

Az a bizonyos említésre méltó eszköze pedig a „Batch conversion / rename” nevet kapta, azaz köteget konvertálás / átnevezés. Egészen mélyen testre szabható, hogy milyen kimeneti fájlokat szeretnék kapni, milyen névvel. A formátumot, a minőséget, a méretet, esetleges képfeliratot, és még sok minden mást

^[18] az IrfanView hivatalos oldala: <http://www.irfanview.com/>

beállíthatunk, hogy valóban olyan eredményt kapjunk amelyet szeretnénk. Rengeteg kép formátumot ismer, és szinte mindre tud konvertálni. Igazán sok gépies munkát tud levenni a vállunkról, amikor egy csokor képből - akár egy weblaphoz is – kisebb, miniatűr képeket – ún. thumbnail-eket – kell gyártanunk. Emellett van egy másik érdekes lehetőség a programban, mégpedig a panoráma kép készítése. Itt több kép egymás mellé, vagy egymás alá helyezésével készít egy nagy képet a program.



3.4 MySQL Workbench

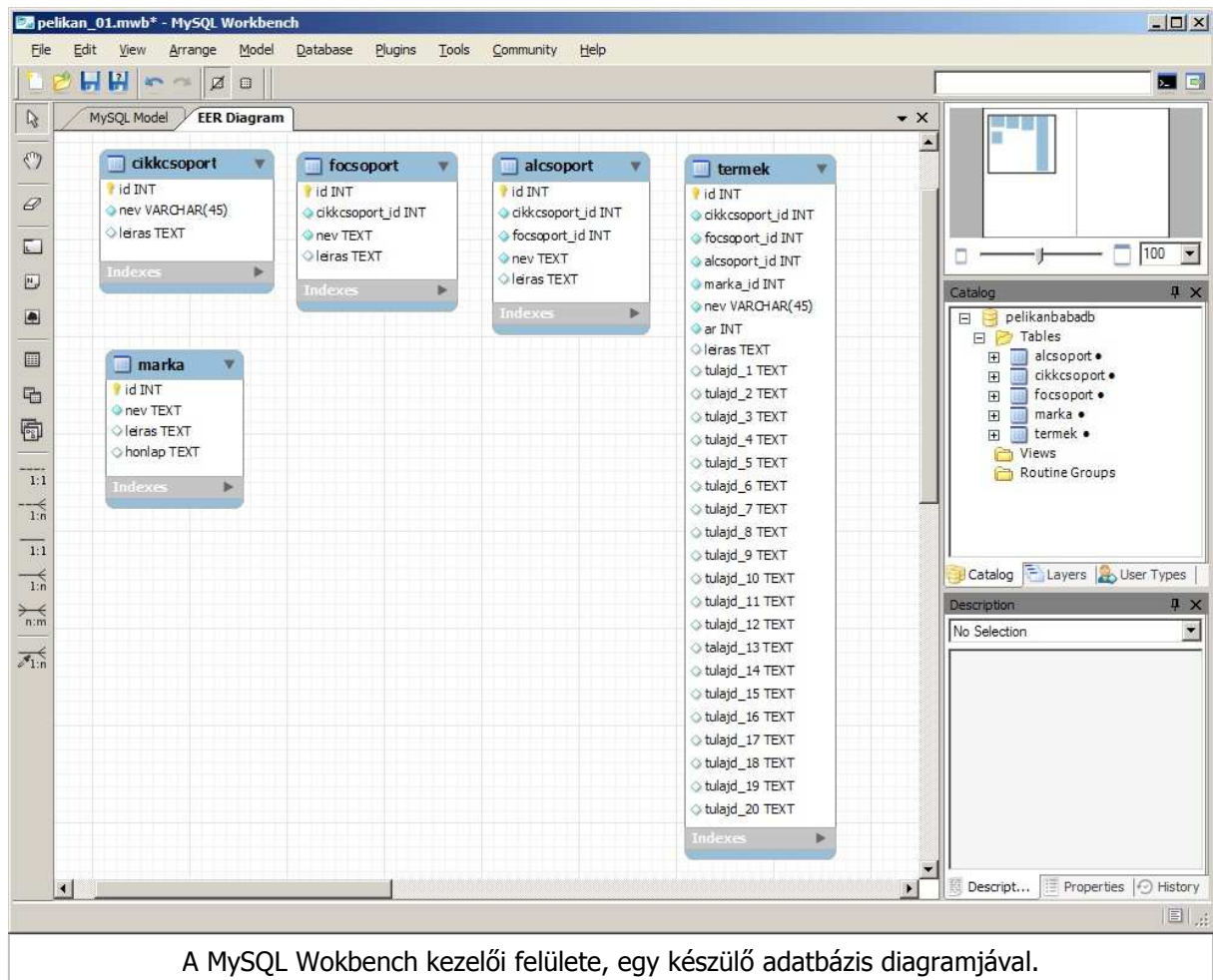
A MySQL Workbench^[19] egy grafikus adatbázis modellező eszköz – kifejezetten a MySQL adatbázis rendszerhez fejlesztve, a saját fejlesztőik által – ami egy felületen egyesíti az adatbázis tervezés, modellezés, létrehozás, illetve karbantartás minden műveletét egy egyszerűen használható és ingyenes elérhető szoftverben. Jelenleg kétféle kiadása van a programnak, amiből az egyik az OSS (Open Source Software, azaz nyílt forráskódú szoftver amiből következik, hogy ingyenesen használható), a másik pedig a Standard Edition, ami már kereskedelmi verziót takar. A két kiadás funkciói összehasonlításának elősegítésére elérhető egy egyszerű táblázat itt: http://wb.mysql.com/?page_id=11.

Ahogy az előző szoftverek esetében is, erre is igaz, hogy sikerült egy igazán kompakt programot készíteni a fejlesztőknek, mert a letölthető (tömörített) változatok el sem érik a 20MB-ot, és ebből is létezik hordozható verzió, ami nem igényel telepítést. Sokan nem tudják, de ez igazán nagy előnyt tud jelenteni, amikor más helyen kellene elővennünk valamilyen indokból a munkát. Ekkor egyszerűen csak felmásolunk egy kevés adatot a számítógépre, és máris ott van előttünk a használható program az összes beállításunkkal, módosítással, ahogyan mi megszoktuk.

Az eszköz használata egyáltalán nem bonyolult, aki már kezelt hasonló grafikus adatbázis modellezőt annak ismerős lesz a program legnagyobb része. Új modell készítésekor lehetőségünk van új diagramot létrehozni, új sémát, új privilégiumokat, új szkriptet, vagy új megjegyzést felvenni. A sémán belül a szokásos dolgokra nyílik lehetőség: táblákat, nézeteket, eljárásokat, vagy eljárás csoportokat hozhatunk létre. Amennyiben eleve úgy kezdjük a modellt, hogy egy új diagramot nyitunk meg, akkor az egész struktúrát tervezhetjük interaktívan és folyamatosan nyomon tudjuk követni, hogy éppen hol állunk. A diagramban parancsikonok, és

^[19] a szoftver hivatalos honlapja: <http://wb.mysql.com/>

menük segítségével tudunk létrehozni tartalmat, majd a tartalmakat „összeköthetjük”, hogy azonnal megjelenjen vizuálisan is amit terveztünk.



Egy ilyen vizuális szerkesztő sokszor könnyebbé teszi a munkát, ugyanis nem kell elképzelnünk, hogy éppen hogyan is néz ki az adatbázisunk, hanem meg tudjuk nézni. Láthatjuk az egyes táblákat, nézeteket, azok adatmezőit, tulajdonságait. Láthatóak az elemek közti kapcsolatok, és azok tulajdonságai. Sokkal könnyebb ezekkel a szinte megfogható dolgokkal manipulálni, mint folyamatosan

megpróbálni fejben tartani a jelenlegi állapotokat, vagy papírra rajzolni hogy éppen mikor, mit módosítunk és ez hogyan hat ki a teljes sémára.

Amikor úgy gondoljuk, hogy elkészültünk a tervezéssel, akkor a program lehetőséget ad arra, hogy az egészből generáljon nekünk egy szkriptet ahol beállíthatjuk, hogy milyen adat definíciós utasítást^[20] kívánunk használni (ALTER, CREATE). Ha csak szeretnénk bemutatni, hogy hogyan fog felépülni az adatbázisunk, akkor pedig akár PNG^[21], vagy SVG^[22] kiterjesztésű képként, esetleg PDF^[23], vagy PostScript^[24] dokumentumként is lementhetjük a struktúra grafikus megjelenítését.

3.5 Mozilla Firefox

A Firefox^[25] az egyik legelterjedtebb böngésző szoftverek egyike, amit a Mozilla alapítvány fejleszt. Nyílt forráskódú, így ingyenesen használható. Számos internetes szabványt támogat – ami már alap elvárás egy modern böngészővel kapcsolatban – és néhány olyat is, ami várhatóan szabvány lesz a jövőben. A jelenlegi legfrissebb verziója a 3.5.5-ös.

Támogatja a böngésző füleket, beépített – de kikapcsolható, és bővíthető – helyesírás ellenőrzővel rendelkezik, képes a címsorból azonnal keresni, de tartalmaz beépített direkt keresőt is, ahol a felhasználó szabhatja meg, hogy melyik internetes keresőmotort használja a program. Támogatja az RSS^[26]-t, és van egy ún. privát böngészés funkciója is, melynek használatakor a számítógépen semmilyen fájlt nem

^[20] további információ az adat definíciós utasításokról: <http://dev.mysql.com/doc/refman/5.4/en/sql-syntax-data-definition.html>

^[21] további információ a PNG formátummal kapcsolatban: <http://hu.wikipedia.org/wiki/PNG>, valamint a formátum specifikációja: <http://www.w3.org/TR/PNG/>

^[22] további információ az SVG formátummal kapcsolatban: <http://hu.wikipedia.org/wiki/SVG>, valamint a formátum specifikációja: <http://www.w3.org/TR/SVG/>

^[23] további információ a PDF formátummal kapcsolatban: <http://hu.wikipedia.org/wiki/PDF>

^[24] további információ a PostScript formátummal kapcsolatban: <http://hu.wikipedia.org/wiki/Postscript>

^[25] a szoftver hivatalos honlapja: <http://www.mozilla-europe.org/hu/firefox/>

^[26] további információk az RSS-ről: <http://hu.wikipedia.org/wiki/Rss>

tárol el az alkalmazás. Több platformon használható, pl. Microsoft Windows alapú rendszereken, Mac OS X-en, Linuxokon, és egyéb Unix alapú operációs rendszerekre is létezik belőle kiadás.



A Firefox 3.5.5 kezelőfelülete, több böngésző füllel.

Sok felhasználó szerint a Firefox legnagyobb erőssége a számtalan kiegészítő^[27] amivel elláthatjuk, és saját tetszésünk szerint szabhatjuk a böngészőnket. Mivel nyílt forrású, ezért rengeteg felhasználói kiegészítés és mini-al alkalmazás érhető el, amikkel számos további funkciót tudunk bele integrálni a

^[27] a Firefox kiegészítői itt találhatóak: <https://addons.mozilla.org/hu/firefox/>

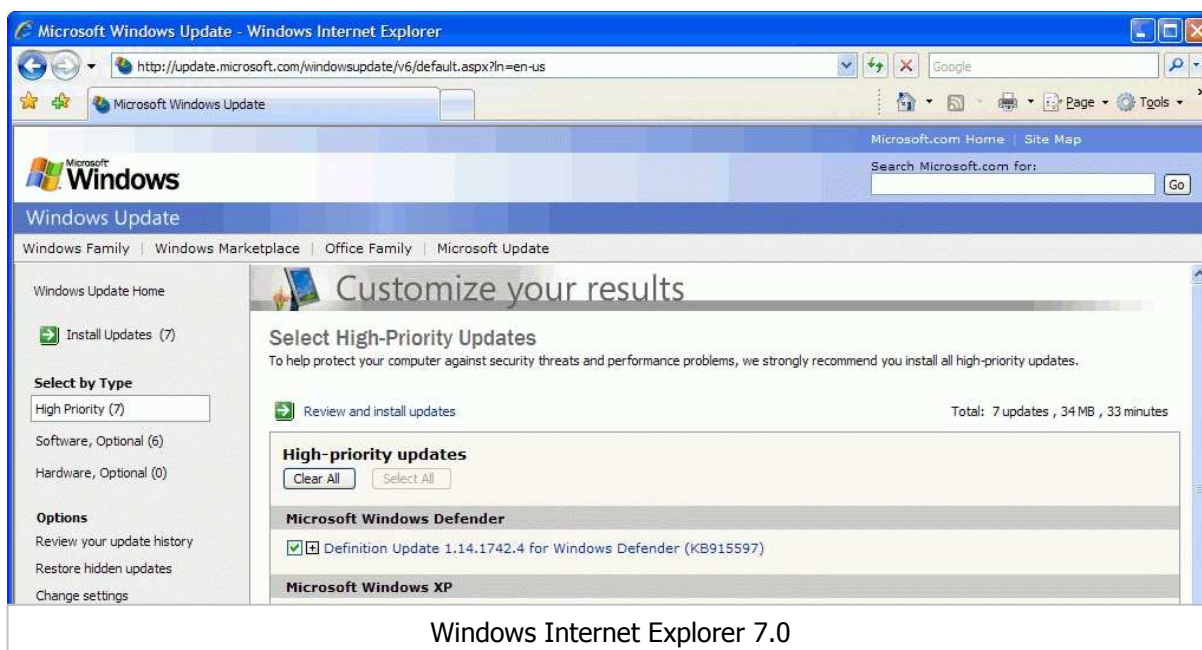
böngészőbe, vagy éppen le tudunk tiltani egyes folyamatokat. (sokak nagy kedvence pl. az Adblock Plus^[28] amivel le tudjuk tiltani a megjelenő reklám hirdetéseket, és így akár gyorsabban is böngészhetünk)

3.6 Windows Internet Explorer

A Windows Internet Explorer (a 7-es verzió előtt még Microsoft Internet Explorer) a jelenlegi legelterjedtebb böngésző. Ez nagyban köszönhető annak, hogy a böngésző a Windows operációs rendszerek részét képezi, így sokan használják azért, mert ez azonnal rendelkezésre áll, amikor a Windows feltelepült a számítógépre.



^[28] az Adblock Plus weboldala: <https://addons.mozilla.org/hu/firefox/addon/1865>



Sokan kifogásolják a szoftvert a nem megfelelő biztonság miatt, azonban mára már rengeteg biztonsági rést befoltozott a Microsoft. Sokszor egészen nagy technológiai lemaradásban is voltak (és néhol még vannak is) a konkurens böngészőkkel szemben. Például a Microsoft Internet Explorer 6-os verziója 2006-ban még csak 70%-os CSS2 támogatással futott – a W3C ajánlásait figyelembe véve –, amikor a konkurens termékek már szinte teljes támogatottsággal kerültek kiadásra. Részben ezért is vannak akkora megjelenítés béli különbségek a különböző szoftverek, és azok különböző verziói között. Az egyik legproblémásabb alkalmazás a fent említett 6.0-s Explorer verzió (nem beszélve az ez előtti verziókról), mert sok szempontból nem helyesek, vagy nem jól implementáltak a funkciók a W3C ajánlását figyelembe véve. A lapfüles böngészés lehetősége is csak a 7-es verzióban jelent meg 2006 novemberében, amikor a Firefox már sokkal korábban használta ezt az ötletet, ami sok felhasználó számára vált nagyon gyorsan szimpatikussá. Viszont ezzel a 7-es verzióval a Microsoft nagyon sok mindent kijavított, és sokkal inkább szabványos böngészővé tette az Internet Explorer-t, bár maradtak még hiányosságai. A 8-as verziónál már nagy hangsúlyt fektettek a szabványosításra, valamint

törekedtek a teljes, szabványos CSS2.1 támogatás megvalósítására. Továbbá sokat fejlesztettek az RSS és Ajax támogatáson is. Ezekkel a fejlesztésekkel az IE8 már sikeresen teljesíti az Acid2^[29] tesztet, de az Acid3^[30] teszten még nem megy át maradéktalanul.



Windows Internet Explorer 8.0

^[29] további információ az Acid2 tesztről: <http://en.wikipedia.org/wiki/Acid2>;

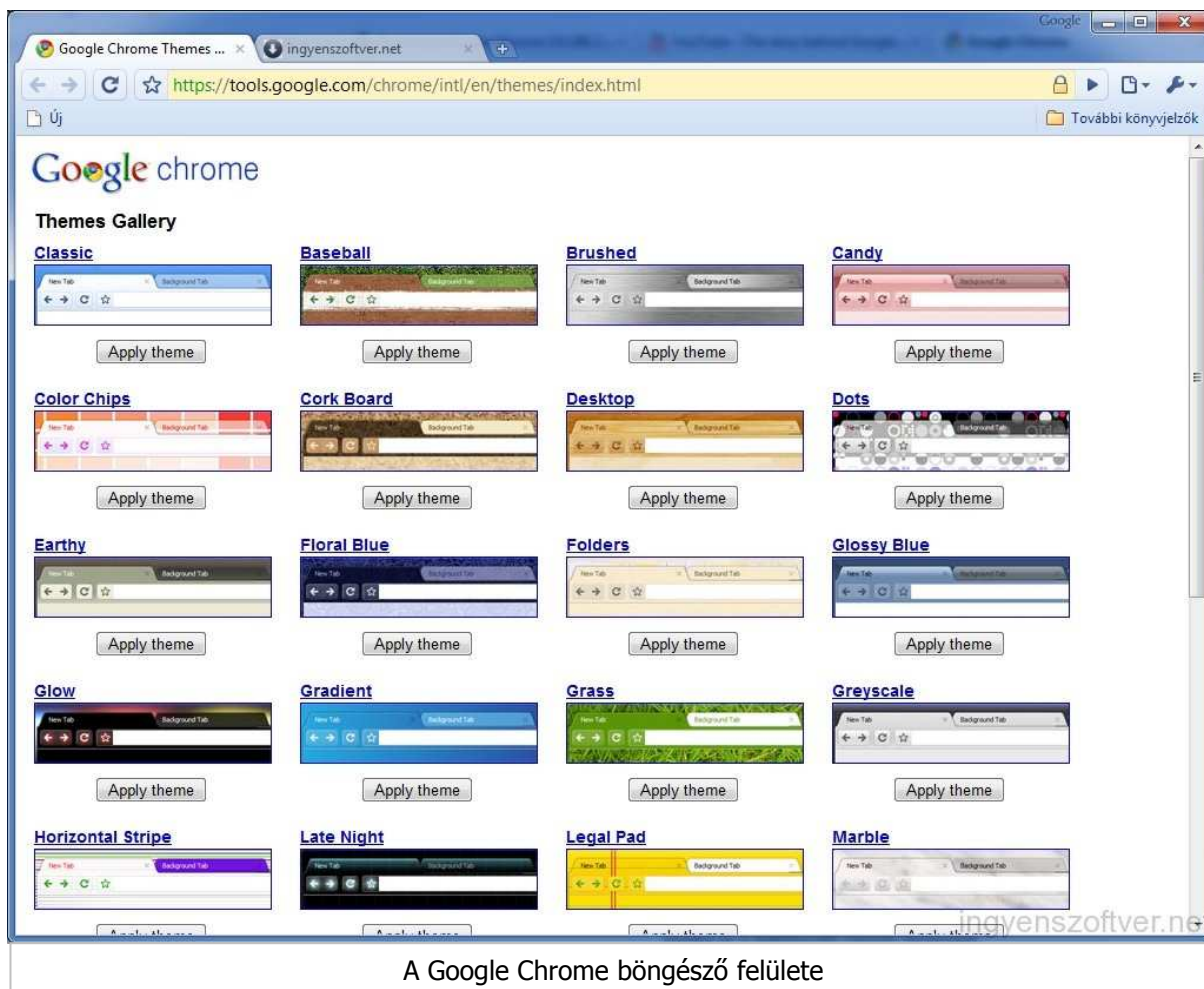
a teszt elérhetősége: <http://acid2.acidtests.org/>

^[30] további információ az Acid3 tesztről: <http://en.wikipedia.org/wiki/Acid3>;

a teszt elérhetősége: <http://acid3.acidtests.org/>

3.7 Google Chrome

A Google Chrome^[31] a böngészők piacán a legfiatalabb ugyan, de már a 3.0.195.32 verziónál jár a legfrissebb stabil változat, és közeledik a 4.0-s kiadása is. A nevéből is látszik, hogy a Google fejleszti, és a WebKit böngészőmotort használja. A szoftver teljesen ingyenes, valamint nyílt forráskódú.



A Google Chrome böngésző felülete

A szoftver említésre méltó előnye, hogy igen gyors és teljesen szabványos megoldásokat tartalmaz, aminek köszönhetően az Acid1, Acid2, illetve Acid3 teszteket is sikeresen teljesíti már a legfrissebb változat. A böngészőt úgy alakították

^[31] a böngésző hivatalos oldala: <http://www.google.com/chrome/>

ki, hogy a kezelőfelület minél kevesebb helyet foglaljon, így a böngészéshez rendelkezésre álló képernyőrész közel maximális. Támogatja a lapfüleket, ráadásul sikeresen megvalósították a multi-threading-et, azaz a több szálon való futtatást. Így a Google Chrome minden egyes lapfülhöz külön-külön processzt indít, ami jelentősen növeli a böngésző biztonságosságát illetve stabilitását. Ha valamilyen oknál fogva lefagyna egy böngésző fül, akkor ahhoz, hogy befejeztessük azt, nem kell bezárni az egész böngészőt, így a többi folyamat zavartalanul folytathatja a munkát, mi pedig böngészhetünk tovább.

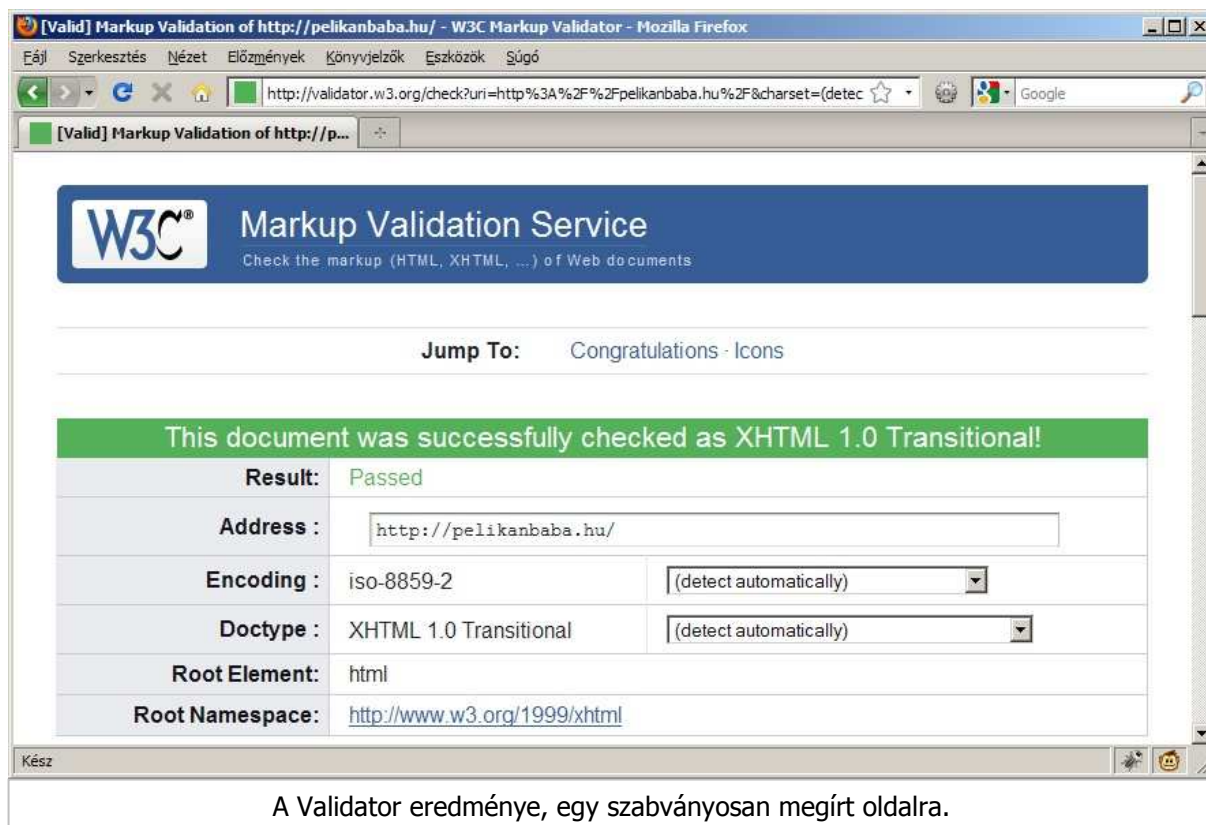
A program egyelőre csak Windows operációs rendszerek alatt érhető el, a Linuxos, valamint Mac OS X-es verziók jelenleg még fejlesztés alatt állnak. A Windows platformú verzió, azonban már megjelenésekor 43 nyelvváltozatot támogatott – ami 38 különböző nyelvet jelent – ami igen kiemelkedő teljesítmény. Másik hátránya például a Firefox-szal szemben, hogy nem lehet hozzá kiegészítéseket telepíteni, de ez a fejlesztők ígérete szerint a 4-es verzióval megváltozik. A Google böngészője eddig is szép eredményeket tudhat maga mögött, azonban ha hasonló mértékben tesztre szabható lesz, mint a Mozilla programja, akkor jó eséllyel a jövő egyik igen közkedvelt böngészőjévé válhat.

3.8 W3C Validator, mint munkaeszköz

A W3C Validator^[32] valójában egy weboldal, ami az elkészült – vagy készülőben lévő – lapok vizsgálatában és tesztelésében nyújt nagy segítséget. Nem csinál semmi mást, csak megvizsgálja az oldal forráskódját és megmondja, hogy az adott oldal vajon megfelel-e a W3C ajánlásának. Ha esetleg hibát talál az oldalon, akkor intelligensen azt is kiírja, hogy melyik sorban és hol találta a hibát, valamint azt, hogy milyen jellegű problémáról van szó. Igen nagy segítséget tud nyújtani az oldalak készítése közben, amikor valamiért nem úgy jelennek meg a lapon a dolgok,

^[32] a Validator itt található: <http://validator.w3.org/>

mint ahogy szerintünk kellene. Sok esetben előfordul, hogy lemarad egy záró tag vagy idézőjel, ami miatt az egész oldal struktúrája felborulhat.



A Validator képes többféle szabványt is ellenőrizni, a legegyszerűbb mód hagyni neki, hogy a lap fejlécében lévő információk alapján eldöntse, hogy melyik szabványra kell, hogy történjen a validáció. A fejlesztők nem csak a HTML oldalakra készítették el ezt a kellemes ellenőrző alkalmazást, hanem külön RSS^[33]-re, CSS^[34] fájlokra is, valamint képes felfedezni rosszul megadott linkeket, és további eszközök is rendelkezésre állnak az oldalon. További kényelmi funkció, hogy megadhatunk minden eszköznek egy linket, amit ellenőrizni szeretnénk, de lehetőségünk van kézzel bevitt adatokat is leellenőriztetni, vagy feltölteni egy fájlt, amit átnéz az oldal.

^[33] RSS Validator: <http://validator.w3.org/feed/>

^[34] CSS Validator: <http://jigsaw.w3.org/css-validator/>

Igazán örömteli, hogy mostanában egyre inkább kezd elterjedni az a nézet, hogy egy weblapnak már illik HTML/XHTML validnak lennie, komolyabb weboldalaknál pedig szinte már követelmény. Ami nem is csoda, hiszen ahhoz, hogy egy megfelelően megjelenő lapot adhassunk ki a kezünk alól, ez az első és legfontosabb lépés, hogy a szabványoknak megfelelően írjuk meg a forráskódot, és majd később lehetőségünk lesz egyéb böngésző specifikus beállításokat is elvégezni, amennyiben szükséges.

4. Munkafolyamat

A munkafolyamatoknak nem kell feltétlenül követnie ezt a sémát ami lejjebb lesz látható, mivel ezt csak egyféle ajánlásnak szánom, amitől tetszőlegesen el lehet térni. Sokan dolgoznak különféle módszerekkel, különböző sorrendben, és mindenkinek megvan a saját kialakult munkaszisztémája. Inkább szeretnék egy áttekintést nyújtani arról, hogy milyen lépésekből áll egy oldal kivitelezése, és az egyes lépésekben milyen teendőkkel kell, vagy ajánlatos foglalkozni. A sorrendet néhány esetben tetszőlegesen fel lehet cserélni, de vannak olyan lépések amiket nehezen lehetne elképzelni fordított sorrendben – gondoljunk csak arra, hogy hogyan állítanánk össze a lap vázát, ha nincsen kész legalább egy előzetes látványterv, ahonnan a design darabjait össze tudjuk válogatni, és tudjuk illeszteni.

4.1 Mit várunk el az oldaltól?

Ez az egyik legfontosabb kérdés amit minél előbb le kell, hogy tisztázzunk vagy magunkkal, vagy pedig az oldal megrendelőjével. Nagyon nem szerencsés dolog bármit is elkezdni anélkül, hogy legalább 80-90%-ig pontos víziónk ne lenne azzal kapcsolatban, hogy pontosan mit is kell majd felmutatnunk a munka végeztével. Sok esetben már itt elbukik egy-egy munka, mert nem megfelelően kommunikálták le a megrendelővel, hogy mik is az Ő igényei, és hogy azokból mit, és hogyan lehet megvalósítani. Amennyiben kellő energiát fordítunk arra, hogy egyeztessük az álláspontokat és megbeszéljük, hogy milyennek is kellene lenni annak a weblapnak amikor elkészül, akkor már sokkal-sokkal közelebb vagyunk ahhoz, hogy a végén egy ún. win-win szituációt teremtsünk, ahol a megrendelő elégedetten és mosolyogva távozik, mi pedig büszkén mutathatjuk fel legújabb elkészült munkánkat. Ez az ami megalapozhatja a jövőbeni munkáinkat is – ha szeretnénk ezen a területen dolgozni –, ugyanis abban szinte mindenki egyet ért, hogy az elégedett ügyfél a legjobb reklám.

Amennyiben saját magunknak fejlesztünk valamit, akkor se hagyjuk ki ezt a lépést – persze egy kicsit másképpen. A saját elképzeléseinket kell papírra vetnünk és összehasonlítani a technikai és gyakorlati tudásunkkal, valamint a rendelkezésre álló erőforrásokkal. Ha tisztában vagyunk a saját magunkkal szembeni elvárásainkkal, a rendelkezésre álló eszközeinkkel, és ezeket sikeresen összhangba hozzuk, akkor elkerülhetünk néhány görcsös órát – amikor nem akarnak sikerülni a dolgok –, vagy legrosszabb esetben a kudarc élményét, amire egyikőnk sem vágyik. Mindenki szeretne minél jobban megszerkesztett és minél tökéletesebb munkát kiadni a kezéből.

A konkrét példánál maradva a pelikanbaba.hu domain név alatt látható weblap megrendelőinek csak általános elvárásai voltak néhány apró kitéttel, mint az sok esetben lenni szokott. Egy olyan bemutatkozó jellegű weblap, amin a látogatók kényelmesen, és könnyen tudnak tájékozódni, de elegendő információval szolgál a cég által forgalmazott termékekről, és később egy hozzáértő szakember nélkül is könnyen lehessen módosítani legfőképp a termékpalettán. Nem volt cél, hogy az oldal minden része módosítható legyen, mert elképzelhető, hogy túlzott komplexitáshoz vezetett volna, ami így nehézkesen karbantarthatóvá tette volna azt egy átlag felhasználónak. A megrendelők kifejezett igénye volt, hogy egy modern és letisztult, de színes, kedves kinézetű oldaluk legyen, ami kellően passzol egy babaáruházhoz.

4.1.1 Design és használhatóság ellentéte

Ki ne látott volna olyan weboldalt, ami hihetetlenül szépen nézett ki, de valahogy olyan furcsán volt összerakva, és nem akart kézre állni az egész? Vagy az ellenkezőjét, amikor egy teljesen jól működő és egyébként kellemes honlapot a túlzott puritánság rontott el? Ezek azok a helyzetek, amiket el szeretnénk kerülni mindannyian. Valahol a kettő között van az a pont, ahol egy kellemes kinézetű oldal található egy helyesen megoldott navigációval, és a funkciók átgondolt

kivitelezésével. Manapság egyre inkább elvárás lesz, hogy egy honlap legyen szemet gyönyörködtető, még jobb, hogy ha animációk is vannak benne, és sok esetben pont ez lesz az a dolog, ami elriasztja a látogatót az oldalunkról. Elég aggasztó és unalmas tud lenni, amikor egy-egy gomb megnyomása után 1-2-3 másodpercet is kell várunk csak arra, hogy az animáció végigfusson, és végre megnézhezzük amire kíváncsiak vagyunk.

Ezért is fontos az, hogy a design terv elkészítésekor ne csak arra figyeljünk, hogy az oldal kinézete megfelelő legyen, hanem arra is, hogy a kivitelezés során megfelelő helyre kerüljenek a megfelelő elemek. Úgy kell kialakítani a grafikai elemeket, hogy kellő helyet adjunk mind az oldalhoz tartozó navigációnak, mind pedig a lényegi tartalom megjelenítési helyének úgy, hogy elváljon egymástól a két vagy több részterület, de szerves egységként jelenjenek meg, és megfelelő kiemelést kapjanak a lényeges dolgok.

4.2 Az oldal megtervezése

A tervezéskor a fentebb említett dolgokat figyelembe véve kell összeraknunk az oldalt, de már a konkrét elképzeléseket formába öntve, közelítve a végeleges megjelenéshez. Itt tárgyaljuk majd egy kicsit részletesebben a különböző rész folyamatokat, mint a háttérben működő adatbázisnak, az oldal strukturális felépítésének, valamint a grafikai megjelenésnek a tervezését. Konkrét tervek tárgyalása helyett inkább szeretném felhívni a figyelmet néhány általános irányelvre, valamint azokra a dolgokra, amikre érdemesebb nagyobb hangsúlyt fektetni ebben a szakaszban.

4.2.1 Adatbázis tervezés

Ahogy a 3.4-es szakaszban is kifejtettem, az adatbázis, vagy adatbázisok megtervezésekor érdemes valamilyen vizuális tervező programot

használni, mert nagyban megkönnyítheti a munkafolyamatot, főleg nagyobb és komplexebb adatbázisok tervezésekor, ahol nem olyan egyszerű fejben tartani akár több-tíz – esetleg több-száz – adatbázis tábla adatait, és kapcsolati rendszerét.

Ennél a pontnál már tisztában kell, hogy legyünk azzal, hogy a lapunknak mekkora mértékű dinamikusságot szánunk. A bevett szokás kb. úgy néz ki, hogy az oldalon belül nem minden lap töltődik fel dinamikus tartalommal, de ettől persze el lehet térni. Általában a bemutatkozó rész, vagy a kapcsolattartási rész tipikusan olyan elemek, amiknek az esetek túlnyomó többségében nincs szüksége dinamikusságra, mivel nem változnak rendszeresen. Így sokkal egyszerűbb egy statikus oldalt létrehozni hozzájuk, amivel kevesebb munka is van, valamint nem terheli feleslegesen például az adatbázis szerveret, mert nem kell minden oldalletöltéskor lekérni az adatbázisból azt a 2-3 telefonszámot, email címet, vagy földrajzi helyzetet, amik ritka esetekben változnak a honlap tulajdonosának élete során.

Amikor tudjuk, hogy milyen tartalmak lesznek dinamikusak az oldalon, azokhoz el kell készítenünk egy adatbázis sémát, hogy mik azok, amiket esetleg szeretnénk valamilyen módon összekapcsolni, és milyen adatokat, tulajdonságokat szeretnénk róluk eltárolni. Erre egy megfelelő példa, a pelikanbaba.hu oldal adatbázisában az, hogy a termékeknek van egy „marka_id” tulajdonsága, ami a „marka” tábla „id” oszlopára hivatkozik. *(megjegyzés: Sajnos a MySQL-ben egyelőre csak az InnoDB^[35] tároló motor esetében van lehetőségünk külső kulcs megszorításokat definiálnunk, amit az adatbázis motor figyelembe is vesz. Lehetőségünk van ugyan egy „REFERENCES” jelző használatára, ami viszont tényleg csak jelzi, hogy egy adott tulajdonság egy másikra hivatkozik, de semmilyen megszorítást nem hoz létre, és nem lehet hozzá pl. „ON DELETE” vagy „ON UPDATE” eseménykezelőket rendelni sem.)* Ezeket figyelembe véve egyszerűen összeállíthatjuk a MySQL Workbench-ben a táblák kinézetét, és az azok közti kapcsolatokat. Érdeemes

^[35] <http://dev.mysql.com/doc/refman/5.4/en/innodb.html>

az eltervezett összefüggések mentén dolgozni, azaz megtervezni egy táblát teljes egészében, majd az ehhez kapcsolódó első táblával megtenni ugyanezt, és így folytatni a további elemekkel. Ez körülbelül olyan folyamatot takar, mint egy rekurzív algoritmus, ahogyan folyamatosan megyünk bele egyre nagyobb mélységben a struktúrába, és amikor már nincsen további lépés, akkor egyet visszalépünk, és folytatjuk ott a munkát. Ezzel a módszerrel hatékonyan lehet csökkenteni a kezdeti hibák kockázatának előfordulását, mert mindig tudjuk, hogy az éppen tervezés alatt álló tábla hova fog kapcsolódni, vagyis honnan indult. Hibakeresés során szintén jól alkalmazható ez a módszer, mert sok időt, és fáradságot lehet vele megtakarítani, ha az összefüggéseket vizsgáljuk, és ezek mentén haladunk.

Ezek az elveken haladva született meg a pelikanbaba.hu oldal adatbázis váza is, ami végül eléggé minimális formát öltött. Mivel ebben a sémában nem a fent említett InnoDB adatbázis motort használtuk, ezért sajnos a táblák között nincsenek megszorítások létrehozva, csak hivatkozunk a struktúrában őt megelőző táblákra. Az elképzelés az volt, hogy a konkrét termékeket kategóriákba soroljuk, akár több mélységben is. Tehát a legalsó szint lesz maga a termék, és ezt kellene valahogyan csoportba foglalnunk. Az első változat egy „Cikkcsoport”, „Termék csoport”, „Termék” struktúra volt, de később praktikusabbnak láttuk egy plusz szint bevezetését is, mivel kiderült, hogy egy adott termékből a rendelkezésre álló többféle színváltozatot is szeretné a megrendelő kiemelni. A végső állapot így az lett, hogy a legfelső szint maradt a „Cikkcsoport” – pl. babakocsik. Ezt követi a „Főcsoport” – pl. sport babakocsik –, majd az „Alcsoport”, ami már majdnem a konkrét termék, de itt még csak a termék típusát jelöljük – pl. Espiro Jet. Végül pedig maga a „Termék”, amiből többféle színváltozat van az alcsoporton belül – pl. Espiro Jet kék. Ezek mellett kellett egy külön tábla, ahol tároljuk az egyes márkákat, amire hivatkozhat majd a termékeket tartalmazó táblánk, és így márkák között is tudnak majd könnyen böngészni a leendő látogatóink.

4.2.2 Az oldal struktúrájának megtervezése

A struktúra megtervezése alatt azt a folyamatot értjük, amikor eldől, hogy a teljes weblap hány oldalból fog állni, valamint hogy hogyan lehet majd az oldalak között navigálni.

Ez mindaddig nem okoz komolyabb fejtörést, amíg a lap legfelső szintjét nézzük, mert a sok esetben bevált és alkalmazott módszer teljesen megfelelő lehet – azaz vagy az oldal fejlécében, vagy valamelyik oldalon elhelyezünk egy menüsort, ahol megadjuk az összes elérhető al-oldalt, és hivatkozást, és ez minden oldalon meg is jelenik. Azonban amikor már olyan ötleteink támadnak – vagy már eleve az oldal összetettsége megköveteli –, hogy több szintű oldal struktúrát alakítsunk ki, akkor már lehet gondolkodni többféle megvalósítási módon. Nagyon elterjedt megoldások a lenyíló menük, és az al-oldalon megjelenő belső navigáció is. Ezeknek is több megoldása létezik, ezt mindenki szabadon eldöntheti maga, hogy melyik szimpatikus neki, vagy melyik illeszkedik legjobban az oldalához, amit jelenleg készít.

A navigáció tervezésénél az egyik legfontosabb szempont, az egyszerűség, átláthatóság, ami így egy felhasználóbarát felületet eredményez. Ide tartozik a logikai strukturáltság is, de ezzel a gondolattal már át is térhetünk az egész weboldal szerkezetére is.

Az oldal tartalmának megjelenítésére is több módszer kínálkozik, amiket figyelembe kell vennünk a navigáció témájánál is. Ugyanis ha például az oldalak tartalmát egy felbukkanó új ablakban szeretnénk megjeleníteni, akkor abban az új ablakban nem lesz jelen a navigációs menü – persze ott is megjeleníthetjük, de nem lenne szerencsés néhány kattintás után 10-15 felugró ablak között navigálni. Élhetünk azzal a módszerrel, amit az egyszerű HTML kód kínál, hogy minden egyes kattintással egy teljesen új oldal töltődik be, ahol bármilyen menüt építhetünk, viszont ha csak főoldali navigációnk van, akkor érdemes megtartani az eredeti

menüt, hogy ne zavarjuk össze a látogatóinkat. Egyéb technológiák használatával (például az Ajax-szal, amire ezen dolgozat keretein belül nem térek ki részletesen) megoldható az is, hogy egy linkre való kattintáskor ne kelljen az egész lapot újra betölteni, csak azt a részét, ahol az új tartalmat szeretnénk megjeleníteni. Ekkor a navigáció, az oldal fejléce, és sok minden más megmarad, ami sokkal kevesebb adatátvitelt jelent a szerver és a kliens gép között, így ezek a megoldások sokkal gyorsabbak lehetnek, és sok terheléstől mentesíthetik a webszervereket. Viszont itt különös figyelmet kell fordítanunk arra, hogy mivel a navigációs menük is megmaradhatnak, így azokat ennek megfelelően kell felkészítenünk, hogy minden tartalom elérhető legyen belőlük.

Mivel már az adatbázis terve készen volt az oldalunkhoz, így nagy segítséget tudott nyújtani, hogy hogyan is kellene megoldani annak kialakítását. A több szintű termékskála megjelenítéséhez mindenképpen kellett egy mindenhol látható és elérhető menü, a termékek böngészéséhez, mivel ez az oldal tulajdonképpen lényege. Valamint kellett még egy olyan menü, amivel bárhol elérhetőek az oldal általános funkciói – ilyenek lehetnek a visszalépés a főoldalra, kapcsolat menüpont, stb. Így az eredmény az lett, hogy kell néhány általános jellegű oldal ami mindenhol elérhető, valamint a termékek megtekintésére néhány strukturált jellegű, amik egymásból, vagy egymás után nyílhatnak meg. De hogy ne kelljen túl sok mélységgel dolgoznunk – plusz a könnyebb átláthatóság kedvéért is – a termék struktúra felső két szintjét – azaz a cikkszoportokat, és a főcsoportokat – kiemelten kezeljük.

4.2.3 Grafikai tervezés

A grafikai tervezés egy olyan pont, ahol valóban csak néhány dologra lehet felhívni a figyelmet, mert maga a kivitelezés és a tervezés folyamata nagyban függ attól, hogy milyen a tervezett design. Emellett sokszor fordul elő, hogy apróbb javításokra van szükség a grafika terén, amíg valóban megkapjuk az utolsó

használható változatot, és ezek sokszor vonnak maguk után egyéb dolgokat is. Előfordulhat többek közt, hogy 1-2 pixellel arrébb kell mozgatnunk valamit, és így az összes körülötte lévő elemet újra kell méretezni, és újra el kell rendezni, ami sokkal tovább is eltarthat, valamint ezek is újabb változtatásokat hozhatnak magukkal.

Amit már az elején érdemes eldönteni, hogy milyen felbontásra optimalizáljuk a lapot, vagy esetleg olyan lapot szeretnénk, ami mindig kitölti a rendelkezésre álló teljes területet. Persze ez utóbbi esetben is szükség van arra, hogy egy minimális képernyő felbontást definiáljunk, ahol az oldal már úgy néz ki, ahogy terveztük. Manapság ez a felbontás leggyakrabban 1024*768 pixel. Fontos dolog még, hogy akarunk-e háttér az oldalunknak, vagy egy egyszerű fehér háttérű arculatot tervezünk. Amennyiben lesz háttér, akkor tudnunk kell, hogy egy színt akarunk, vagy valamilyen összetettebbet, ami csak grafika használatával oldható meg. Ha grafikus háttér tervezünk, szeretnénk-e mozaikból összerakni azt vagy egy darabból fog állni, valamint fix háttérünk lesz, vagy olyan, ami együtt mozog az oldallal görgetés közben.

Nagyon fontos szempont, amit mindenképpen szem előtt kell tartanunk, hogy a weblap funkciója: ismeretek és információk eljuttatása a látogatók felé. Így az elsődleges szempont az, hogy a tartalmi rész kellő helyel rendelkezzen, és lehetőleg ennek legyen alárendelve az oldal többi része. Nem szabad hagyni, hogy az egyéb dolgok, mint a navigáció, menük, grafikai látvány elemek, fejléc, lábléc, stb. elvonja a figyelmet és túl sok helyet raboljon el a lényeges tartalomtól.

Ahogy a 4.1.1 pontban is említettem már, törekedni kell grafikailag az egységesség biztosítására, ugyanakkor megfelelően kell elválasztanunk a logikailag külön tartozó részegységeket. Ezen egyensúly eltalálásával lehet azt elérni, hogy egy modern, kellemes kinézet társuljon, egy könnyen használható informatív tartalommal.

A mi esetünkben ez önkényesen eldöntött fix hátteret jelentett, ami előtt az oldal tartalmi része mozog. A megfelelő egységesség kialakítására pedig azt a módszert választottuk, hogy az oldal tartalmi része egy közös részben lesz megjelenítve, ami így segít láttatni az összetartozást a külön részek között. A külön részeket pedig kisebb blokkokba csoportosítottuk, amiket a nagy közös panelen helyeztünk el. Ezzel az egyszerű húzással megoldottuk a fent említett problémák jelentős részét, már csak grafikailag kellett megvalósítani az egységet a megfelelő színek összhangjának alkalmazásával.



4.3 A megvalósítás

Maga a megvalósítás célszerűen legtöbbször azzal kezdődik, hogy elkészül az oldalról egy látványterv, akár egy egyszerű kép vagy csak vázlat formájában, ahol már a tervezésnél említett szempontokat kellően figyelembe vették, és azok szerint született meg az elképzelés. Ha ezzel elégedettek vagyunk, akkor kezdődhet az oldal statikus vázának a felépítése úgy, hogy a látványterv darabjait próbáljuk a helyükre igazítani. Amikor úgy néz ki, hogy minden a megfelelő helyre került, akkor pedig jön az oldalak, vagy csak egy részük dinamikussá tétele, majd feltöltésük tartalommal. Ezeket a lépéseket vesszük most sorba.

4.3.1 Grafika és design

A grafikai tervezést érdemes olyan szoftverrel végeznünk, ami támogatja a rétegeket. Ez egy kicsit hasonlít ahhoz, amikor a HTML kódban div-ek segítségével alakítjuk ki a felhasználói felületet, ahogyan egymásba ágyazzuk a tartalmazó objektumokat. A korábban tárgyalt GIMP, vagy esetleg az Adobe Photoshop is tökéletesen megfelel a célnak.

Érdemes a különböző logikai egységekhez tartozó részleteket külön rétegen megtervezni, mert így sokkal könnyebb lesz később módosítani őket, valamint nagyobb szabadságot biztosít a pozicionálás, és méretezés terén. Sok programban megtalálhatóak direkt webfejlesztőknek szánt eszközök, ahogyan az előbb említettekre is igaz. A design munka végeztével egyszerű segédvonalak használatával be tudjuk jelölni, hogy egy-egy elem mekkora kiterjedésű – például a menüsor, egy gomb, egy térelválasztó, stb... Amint beállítottuk a vonalakat – esetleg el is neveztük a részeket, ami erősen javasolt –, akkor a program automatikusan feldarabolja nekünk, és kimentí részenként a kész tervet. Nekünk nem kell a vágással, és méretezéssel foglalkoznunk minden egyes darab esetében.



A végleges grafikai megjelenés.

A pelikanbaba.hu oldal tervezésekor a fenti képen jól láthatóan elkülönülő paneleket természetesen külön-külön rétegekként terveztem. *(tulajdonképpen azokon belül is több al-réteg volt, de a szemléltetés szempontjából ez most lényegtelen)* Tehát logikailag a logó, és a felirat – a háttérrel, kerettel, stb. együtt – egy egységnek tekinthető, így érdemes őket külön rétegen kezelni. Hasonlóan a tartalmat keretbe foglaló sárga doboz is külön kezelendő, illetve az azon elhelyezkedő panelek, és részek is megérdemelnek legalább egy-egy saját réteget, a praktikusság szempontjából.

Amint elkészültünk ezekkel, máris kezdhethetünk foglalkozni az oldal HTML vázával, hogy hogyan fog ez megjelenni egy böngészőben, vagy hogyan fog megjelenni a különböző böngészőkben.

4.3.2 Oldalfelépítés – HTML és CSS

Minden oldal kódjának a megírását érdemes a kötelezően megadandó elemekkel kezdeni, hogy ezekkel később már ne legyen probléma. A kötelező elemeken kívül érdemes még néhány javasolt elemet is elhelyeznünk a kódban, mint például a karakterkódolást megadni, kereső szavakat, és kulcs szavakat definiálni az oldallal kapcsolatban, és hasonlókat.

Amikor ezekkel végeztünk, akkor eldönthetjük, hogy milyen oldalfelépítést szeretnénk választani. Régebben az egyszerű cím és bekezdés típusú formázások voltak főként, de ahogy az igények növekedtek az egyre több grafikát tartalmazó oldalak iránt, úgy kezdett előtérbe helyeződni a táblázatos formázás. Táblázatok segítségével tudták pontosan definiálni egy-egy képi vagy szöveges elem helyét az oldalon, viszont ez a módosíthatóságot, és a könnyű kezelhetőséget alaposan lecsökkentette. A manapság legelterjedtebb formázási módszer a div-ek használata, amik egyfajta konténerként szolgálnak, és tartalmazhatnak bármilyen elemet, beleértve újabb div-eket is.

Ezekhez már sokkal többféle formázási lehetőség kapcsolódik, a CSS-nek hála. A div-ekkel tulajdonképpen bármit megvalósíthatunk, amit csak elképzeltünk az oldalon, ráadásul úgy, hogy ezek könnyen kezelhetők maradnak, és könnyen módosíthatóak. Tetszés szerint változtathatjuk ezek elhelyezkedését az oldalon, akár abszolút pozícionálást használva, akár relatívan megadva a helyzetüket, egy másik objektumhoz képes, vagy éppen a tartalmazó div-et használva referenciaként. Egyszerűen megoldhatóak lettek az olyan problémák például, mint a szöveg, vagy bármely egyéb elem előtt és mögött való elhelyezés, amivel egyszerűen lehet felbukkanó ablakokat is létrehozni anélkül, hogy valóban egy új ablakot nyitnánk. Így sokkal leegyszerűsödött a fejlesztés folyamata, hogy egy modern kinézetű, könnyen kezelhető weblapot állítsunk össze. Ennek az egyik legkönnyebb, és leginkább felhasználóbarát módja manapság a div-ekkel való megvalósítás.

Az oldalam végleges felépítése elég jól szemlélteti a különböző div-ekkel formázott lapokat.

```
<!-- WORKSPACE ***** -->
<div class="workspace">

    <!-- TOP MENU ***** -->
    <div class="top_menu">
        <?php require_once("top_menu.php"); ?>
    </div>
    <!-- END OF TOP MENU ***** -->

    <!-- BUTTON PANEL ***** -->
    <div class="button_panel" id="button_panel">
        <?php require_once("button_panel.php"); ?>
    </div>
    <!-- END OF BUTTON PANEL ***** -->

    <!-- MAIN PANEL ***** -->
    <div class="main_panel">
        <?php require_once("main_index.php"); ?>
    </div>
    <!-- END OF MAIN PANEL ***** -->

    <!-- RIGHT PANEL ***** -->
    <div class="right_panel">
        <?php require_once("aktualis_panel.php"); ?>
    </div>

    <div class="right_panel">
        <?php require_once("marka_panel.php"); ?>
    </div>
    <!-- END OF RIGHT PANEL ***** -->

    <!-- BOTTOM MENU ***** -->
    <div class="bottom_menu">
        <?php require_once("bottom_menu.php"); ?>
    </div>
    <!-- END OF BOTTOM MENU ***** -->

</div>
<!-- END OF WORKSPACE ***** -->
```

Az index.php file egy részlete a pelikanbaba.hu weboldalról.

A kód részletesen jól látszik, hogy az egész tartalmi részt egy ún. workspace objektum tartalmazza. Ezen belül van a left panel, main panel, valamint right panel objektum. Ezek mindegyike egy külső fájlra hivatkozik, a könnyebb olvashatóság, szerkeszthetőség, kisebb méret, és újra felhasználhatóság miatt. Így pl. minden oldalon csak egy hivatkozást teszek a bal oldalon található gombok paneljára, és azt illesztem a kódba. Ha valamit módosítok a gombsoron, akkor az azonnal minden oldalon látható lesz, és nem kell minden egyes lapra külön beszúrnom az új kódot.

Ezek a panelek így egymástól elhatárolva könnyen és gyorsan kezelhetőek, tulajdonképpen önállóan testre szabhatóak. Ehhez is fejlett eszköztárat nyújt a CSS, hogy a különböző div-ek megjelenését manipulálni tudjuk.

```
.button_panel {  
    background-color: #ffffdd;  
    float: left;  
    width: 148px;  
    padding: 1px;  
    margin-top: 10px;  
    margin-right: 0px;  
    margin-bottom: 10px;  
    margin-left: 10px;  
    font-size: 9px;  
}  
  
.main_panel {  
    background-color: #ffffdd;  
    text-align: justify;  
    float: left;  
    width: 540px;  
    padding: 10px;  
    margin-top: 10px;  
    margin-right: 10px;  
    margin-bottom: 10px;  
    margin-left: 10px;  
}
```

A CSS file egy részlete a weblapról; a bal oldali panel, illetve a középső panel stílus definíciói.

4.3.3 Az adatbázisváz

Ez a fejezet tulajdonképpen valahol az adatbázis tervezése és a dinamikus tartalommal való feltöltés közti néhány lépést teszi ki, de mindenképpen érdemes megemlítenünk, hátha segítséget nyújt néhány kérdésben, vagy segít egy kicsit jobban átlátni az egész képet.

Ha az adatbázis tervezésével elkészültünk, akkor igazán sok dolgunk nincsen, mert a MySQL Workbench segítségével automatikusan elkészíthetjük az adatbázis létrehozásához való szkriptet. Ennek segítségével tulajdonképpen készen is van az adatbázisunk, amit már csak össze kell kötnünk valahogy a weblapunkkal. Az egyszerűség kedvéért egy rövid kódrészletben bemutatom, hogy ez hogyan is működik PHP segítségével.

```
<?php
    $dbHostname = "<adatbázis szerver host neve>";
    $dbUsername = "<adatbázis szerver felhasználói neve>";
    $dbPassword = "<adatbázis szerver jelszava>";
    $dbName      = "<használni kívánt adatbázis neve>";

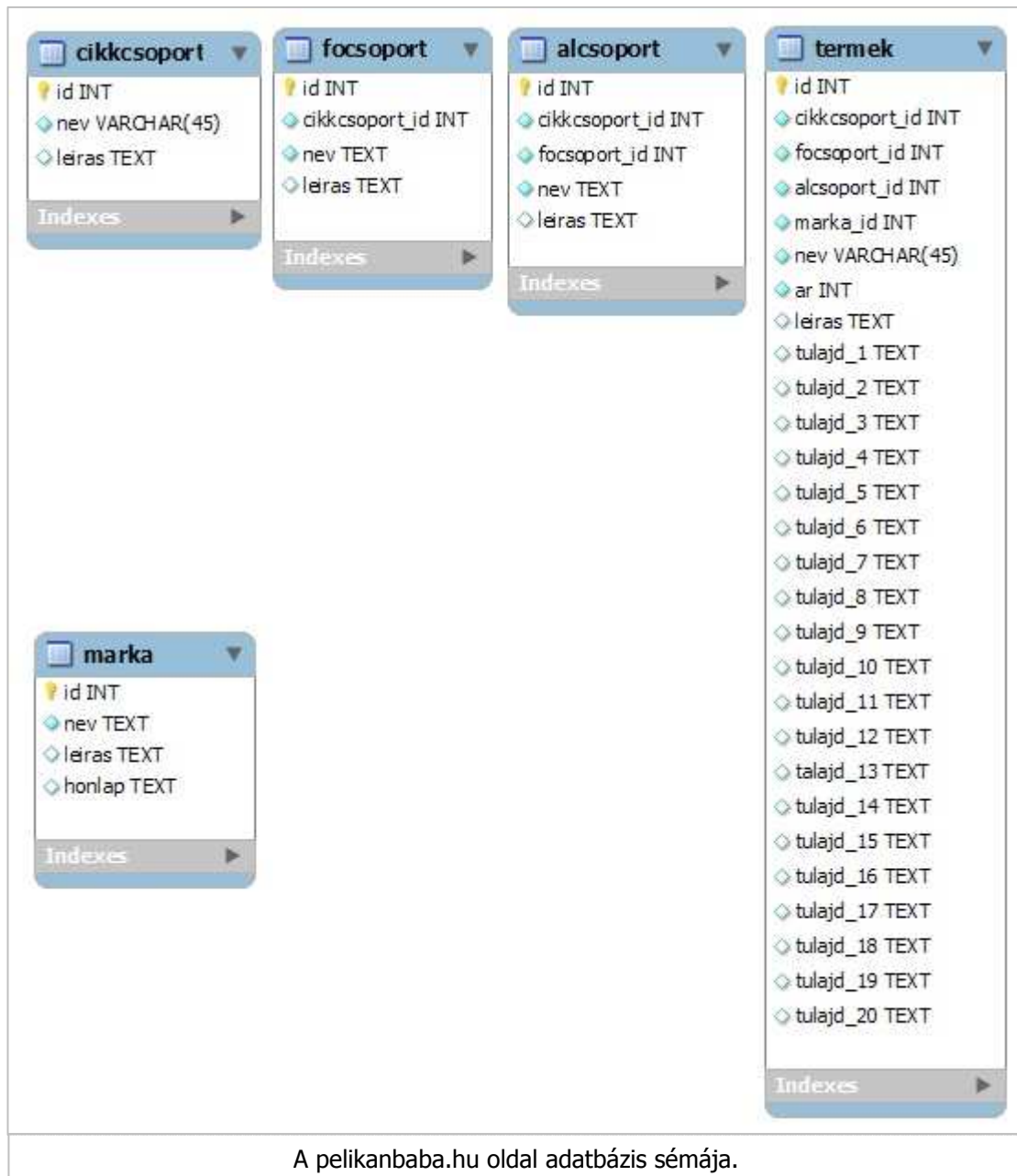
    $dblink = mysql_connect($dbHostname, $dbUsername, $dbPassword) OR
                die("Hibauzenet: nem tudok kapcsolódni az adatbázishoz!");

    mysql_select_db($dbName) OR
    die("Hibauzenet: nem tudom kiválasztani az adatbázist!");
?>
```

Kapcsolódás egy MySQL adatbázishoz PHP segítségével

Amint ezzel megvagyunk, már használhatunk is SQL lekérdezéseket a PHP kódunkban az adatbázis manipulálására.

A pelikanbaba.hu oldal adatbázis vázának képe már elég jól körvonalazódott a tervezéskor is, de azért egy gyors áttekintésként álljon itt egy kép az elkészült sémáról:



Mint az bizonyára szembetűnik az ábráról, van néhány adat amit redundánsan tárolunk le, és nem biztos, hogy szerencsés dolog. Ugyanis a „termék” táblában elég lenne az „alcsoport” tábla „id”-jára hivatkozni, és feleslegesen tároljuk

le mind a „focsoport_id”-t, mind pedig a „cikkcsoport_id”-t. Ugyanígy az „alcsoport” táblában is bőven elég lenne, ha azt tudnánk, hogy az melyik főcsoportoz tartozik. Ez az a pont, ahol egy kicsit előtérbe helyeztem a gyakorlati hasznót, az elegáns megoldásokkal szemben. Így egy esetlegesen hibás adatbázist elég könnyű előidézni, ha nem figyelünk kellően. A másik oldal ami miatt mégis ezt a megoldást választottam, hogy az oldal használata során így lehetséges kisebb terhelést róni a szerverre, és gyorsabb lekérdezéseket produkálni az adatbázisból. Ugyanis ha valami miatt meg szeretnénk tudni, hogy egy termék melyik cikkcsoportba tartozik, akkor nem kell mind a 4 táblát összekötnünk JOIN utasításokkal, hanem elég csak a „termek” táblát, és a „cikkcsoport” táblát összekapcsolnunk – esetleg nincs is szükségünk JOIN-ra, amennyiben csak a „cikkcsoport_id”-ra vagyunk kíváncsiak. Ez egy nagy adatbázis esetében igen sokat jelenthet sebesség és terhelés terén.

4.3.4 Dinamikus tartalmak

Készen vagyunk az oldal felépítésével, megjelenik a böngészőben a HTML kódból generált lap, sikeresen összekapcsoltuk az eddig még statikus lapunkat a használni kívánt adatbázissal, így nincs más dolgunk, mint az adatbázisból elővenni az adatokat, megfelelő formára hozni azokat, és beilleszteni a kódunkba a megfelelő helyre.

Ennek megvalósításában nagy segítségünkre lesz a PHP beépített MySQL függvénytára^[36]. Ezekkel a beépített PHP függvényekkel gyakorlatilag mindent meg lehet valósítani az adatbázisban, amire parancssorból, vagy más kezelőfelületről képesek lennénk. Az oldal publikus részén az esetek többségében másra nem lesz szükségünk, csak olyan lekérdezésekre, amik adatot olvasnak ki a táblákból valamilyen formában. Ezeket az adatokat ha szükséges akkor egyéb függvényekkel tovább formázhatjuk, vagy szétbonthatjuk, majd egyszerűen csak kiíratjuk őket a képernyőre – pontosabban a generált HTML kódba. Az adatok megjelenési formáját,

^[36] <http://hu.php.net/manual/en/ref.mysql.php>

az őket körbeágyazó HTML forrás fogja meghatározni, a PHP csak egyszerűen visszaadja az egyes változóban tárolt adatok értékét, mint string-et, vagy számot, és nem tartalmaz semmilyen formázást alapértelmezés szerint.

```
$q = mysql_query("SELECT id, nev FROM alcsoport WHERE focsoport_id='".$focsoport_id."' ORDER BY nev ASC");

$i = 1;
while ( $f = mysql_fetch_assoc ( $q ) ){
    if ( file_exists( $pic_dir."/alcsoport/".$f["id"].".jpg" ) )
        $kep = "<img src=\"images/alcsoport/".$f["id"].".jpg\"
style=\"border-color: #cccccc; border-style: dashed; border-width: 1px;\"
alt=\"".$f["nev"]."\" />";
    else
        $kep = "<img src=\"images/no_pic.jpg\" style=\"border-style:
none;\" alt=\"".$f["nev"]."\" />";

    if ( $i % 4 == 1 ){
        echo
            "\t\t<div class=\"main_panel_item_group\">\n";
    }

    echo
        "\t\t<div class=\"main_panel_items\"
onclick=\"javascript:window.location='alcsoport.php?a_id='".$f["id"]."'\">
\n".
        "\t\t\t<div style=\"height:110px;\">".$kep."</div><br />\n".
        "\t\t\t<a
href=\"alcsoport.php?a_id='".$f["id"]."'>".$f["nev"]."</a>\n".
        "\t\t</div>\n\n";

    if ( $i % 4 == 0 ){
        echo
            "\t\t</div>\n";
    }

    $i++;
}
```

Egy adatbázis lekérdezés, majd kiírása a pelikanbaba.hu oldalról. (részlet)

Ha sikerült beágyaznunk a forráskódba a szükséges lekérdezéseket, valamint vissza is kapjuk a kért adatokat, akkor más már nem marad az oldal készítéséből hátra, csak a tesztelés aprólékos és alapos lépései.

4.4 Tesztelés

A tesztelésről ugyan külön beszélünk, de mindenképpen meg kell, hogy jegyezzük, hogy a tesztelés nem választható el szervesen a megvalósítás bármelyik folyamatától. Ez a gyakorlatban úgy szokott kinézni, hogy kódolunk valamit, felöltjük a szerverre – kivéve, ha van feltelepítve a saját gépünkre egy szerver alkalmazás –, és teszteljük. Aztán kódolunk, másolunk, tesztelünk, és kezdjük előlről. Apránként haladva, így mindig tudjuk, hogy amit egyszer megírtunk és az elvártaknak megfelelően működött, azzal valószínűleg nem lesz probléma. Ezt külön nem emelem ki a következő fejezetekben, de vonatkozik az összes külön-külön is tárgyalt részre egyaránt.

4.4.1 Megjelenés

Az oldal megjelenésének a tesztelése egyszerűen annyiból áll, hogy a böngészőben, illetve a különböző böngészőkben megjelenő oldalt összehasonlítjuk azzal a látványtervvel, amit eredetileg elképzeltünk, hogy hogyan is kellene megjelennie az oldalnak, amikor elkészül. Sajnos ezzel szokott lenni a legtöbb probléma, ugyanis a különböző böngésző programok különböző elven megírt és felépített motorokat használnak. Manapság már jól kivehető irány, hogy minden egyes fejlesztő csapat arra törekszik, hogy a szabványos megoldások – a sokat emlegetett W3C ajánlások – valóban úgy jelenjenek meg, ahogyan azoknak kellene. Tehát az irány egyelőre jónak tűnik, de még közel sincsenek annyira jók ezek a szoftverek, hogy mindegyikben ugyanazt az eredményt láthassuk egy szabványos kód megírása után.

Azonban nem elhanyagolható a tény, hogy ezek fejlődésével is egyre inkább előtérbe fog helyeződni a szabványosan megírt oldalak jelentősége. Ezért is erősen ajánlott a korábban már említett W3C Validator, vagy Validator-ok használata. Amennyiben az oldalunk mégis problémásan jelenne meg valamely program

használatakor, akkor az interneten rákeresve a problémára biztosan találunk többé-kevésbé elegáns és működő megoldásokat. Vannak direkt ilyen ügyekkel foglalkozó gyűjtő oldalak^[37], portálok, és fórumok, de amennyiben mégsem találunk rá a megoldásra, ne féljünk kérdezni.

4.4.2 Navigáció

A navigáció tesztelése is igen egyszerűen elvégezhető. Sorba véve a navigációs menüket, végig kell lépkednünk azokon, hogy megbizonyosodhassunk arról, hogy mindegyik működik és mindegyik a megfelelő helyre visz minket. Itt ügyelni kell arra, hogy az új lapon – esetleg a dinamikus betöltődő tartalommal ellátott területen – szintén megfelelőek-e az esetlegesen felbukkanó újabb navigációs elemek, valamint biztosítanunk kell, hogy az újonnan megnyílt oldalról tudjunk majd tovább haladni, esetleg visszalépni az előző lapra, így biztosítva a látogatóink számára a folyamatos böngészés lehetőségét.

Amennyiben a tervezési és megvalósítási fázisokban kellő körültekintéssel jártunk el, folyamatosan szem előtt tartottuk a korábban vázolt szempontokat, valamint logikailag is jól van felépítve a navigációnk, akkor ezzel nem szabad, hogy különösebb probléma legyen. Esetleg apróbb hibák, mint elgépelés, helytelen link megadása persze előfordulhatnak, de ezeket aránylag könnyen, és gyorsan orvosolni tudjuk a legtöbb esetben. Komplexebb oldalak esetében, ahol több szintű navigációs menük vannak – esetleg az al-oldalak között külön is tudunk navigálni – már tovább tarthat mindent ellenőrizni, hogy jól működnek-e a tervezett hivatkozások. Ha lehetőségünk van arra, hogy segítséget kérjünk egy olyan embertől, aki még sosem látta az oldalt akkor éljünk vele, mert sokszor külső szemlélőként tud olyan hasznos tanáccsal szolgálni, ami nagyban megkönnyíti a vendégeink tájékozódását az oldalon.

^[37] például: <http://html.lap.hu/>, <http://css.lap.hu/>

4.4.3 Funkció

Az oldal funkcióinak tesztelése nem hagyható a munka végére, ahogyan korábban már ejtettünk erről szót, de muszáj itt is megemlítenünk. Már a fejlesztés közben folyamatos ellenőrzésre van szükség, de legalábbis jobban járunk, ha lépésről lépésre haladva folyamatosan teszteljük, hogy egy adott rész az elvártaknak megfelelően működik-e az oldalon. Ugyanis ha a végére hagyjuk, akkor több ezer sor forráskódból sokkal nehezebb kiszűrni a hibás részt, mintha folyamatosan teszteltünk volna egy-egy önállóan is működni képes kódrészletet annak elkészültekor.

A másik nem elhanyagolható dolog, hogy ha megírunk egy kódrészletet, amit le is tesztelünk, és jól működik, akkor még a munkamenet közben lehet, hogy rájövünk, hogy azt valamilyen módon ki lehet emelni a kódból, és más helyen is felhasználhatjuk. Ezt megtehetjük például egy kicsit általánosabban megírt függvények segítségével, ami nagyban lecsökkentheti a kód méretét, és segíti az olvashatóságot is, valamint sokkal szerencsésebb ún. moduláris felépítést használni az oldalak esetében. Ezzel elérkeztünk a tömören vázolt munkamenet végéhez, és nincs más hátra, mint hogy szemrevételezzük az elkészült oldalunkat, és értékeljük azt.

4.5 Végeredmény

A munkát a tesztelés elkészültével akár tekinthetnénk befejezettnek is, de van egy utolsó lépés, amit nem szabad kihagynunk a sorból semmi esetre sem. Mindenképpen kell egy végső áttekintést tennünk az oldalon – akár néhány nappal a munka befejezése után is –, hogy utólag hogyan látjuk az oldalt. Értékelnünk kell magunkban, hogy mi az amit sikerült elérni a célkitűzések közül, és mik azok amiket esetleg nem. Mindenképpen érdemes levonni a teljes munkafolyamatból a tanulságot, főleg akkor ha vannak olyan részek, amikkel nehezebben boldogultunk, vagy valami miatt problémát jelentettek. Ha ezeket letisztázzuk akár magunkban,

akár másokkal megbeszélve, akkor a jövőbeni munkáinknál nagy segítséget jelenthetnek majd ezek a tapasztalatok. A jó tapasztalatok azért, mert mindig tovább lehet gondolni egy ötletet, és lehet rajta csiszolni, a kevésbé jók pedig azért, mert többet fogunk arról tudni, hogy mire kell nagyobb figyelmet fordítani, és mik jelenthetik a problémás pontokat. Legyen szó akár technológiákról, akár a felhasznált szoftverekről, mindegyikkel érdemes foglalkoznunk, hacsak néhány gondolat erejéig is.

4.5.1 A weblap rövid bemutatása

Az oldal készítésekor próbáltam figyelembe venni az előző pontokban leírtakat, mind a tervezés folyamán, mind pedig a kivitelezéskor. A cél elég világos volt már a kezdetekkor is, és azt hiszem, hogy teljes mértékben sikerült azokat megvalósítani.



A weblap jelenlegi főoldala. Ez köszönt először, amikor ide látogatunk.

Az oldal felépítése igen egyszerűre sikerült. Ugyan elképzelhetőek további módosítások, és újabb al-oldalak hozzáadása, de ezek alapvetően a struktúrát nem fogják befolyásolni. A főoldal betöltődésekor kapunk egy egyszerű felső menüt, néhány fontosabb linkkel, valamint egy 3 részre osztott fő „munkaterületet”. Ennek a fő területnek is a nagy részét, a középső tartalmi rész tölti ki, mivel itt lesz látható az oldal lényegi része, a többi rész, csak navigációra, és egyéb kényelmi, valamint figyelemfelkeltő funkciókra van felhasználva. A bal oldalt található egy navigációs menü, ahol az oldalon jelenleg található termékeket böngészhetjük kategóriánként, a jobb oldalon pedig a jelenleg az oldalon megtalálható márkák listája található. Valamint pillanatnyilag egy kis felhívást tartalmazó doboz.

A termékek böngészése több szintűen lett megvalósítva – ahogyan az adatbázis tervezési résznél már említettem –, hogy a látogatók könnyebben megtalálják a keresett árut. Ez a következő struktúra szerint néz ki:



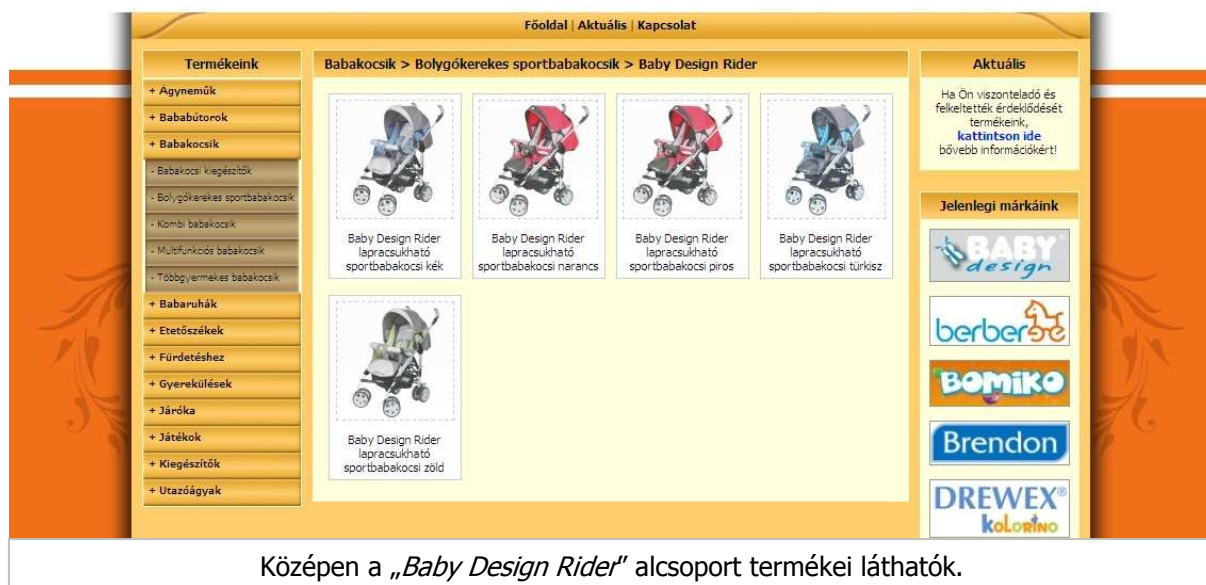
Alapértelmezés szerint a „Cikkszoportok” listája látható a bal oldalon, gombok formájában. Mikor egy cikkszoportot kiválasztunk – azaz rákattintunk – akkor megjelennek alatta a hozzá tartozó „Főcsoportok” lenyíló menü formájában.

A „Babakocsik” cikkszoport lenyitva; alatta a hozzá tartozó főcsoportok listája.

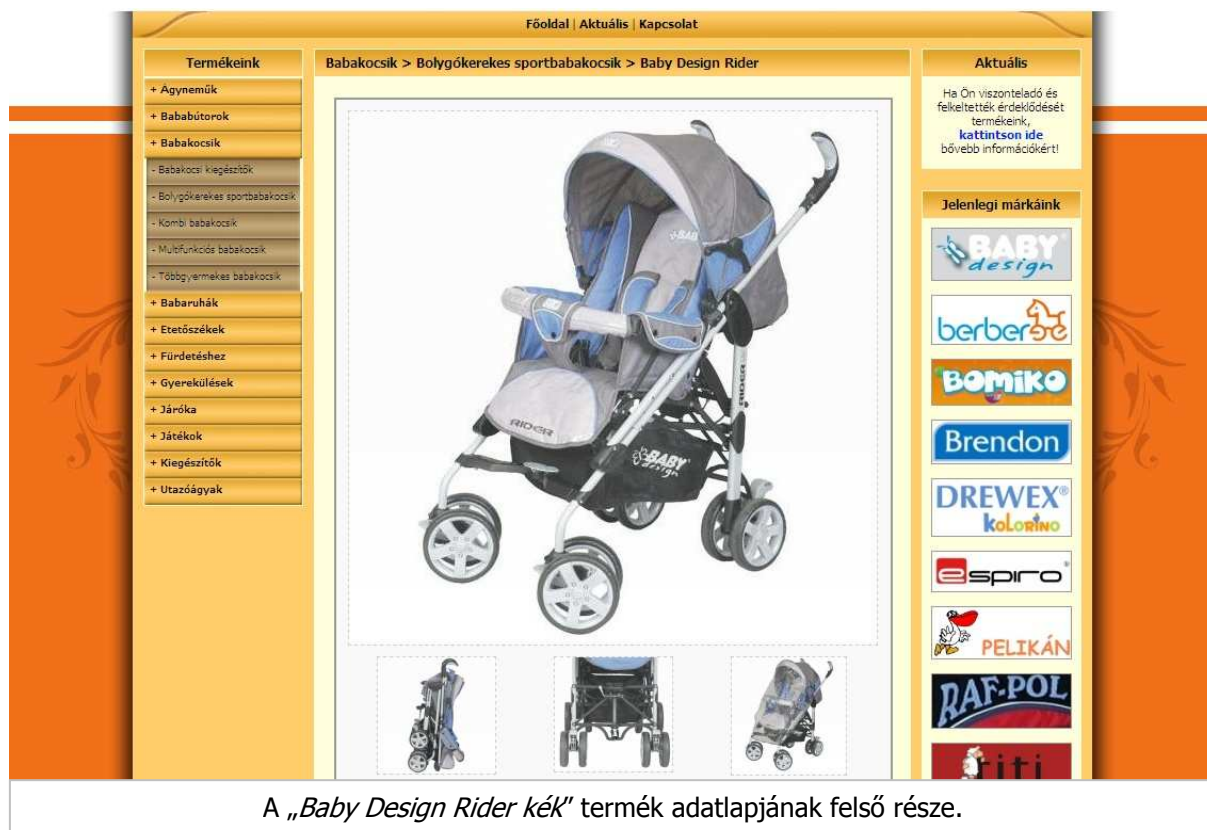
Ekkor még nem változik semmi egyéb az oldalon, csak újabb választási lehetőséget kapunk, hogy mit szeretnénk megtekinteni a főcsoportok közül. Amikor a főcsoportot kiválasztjuk, akkor már nem újabb menü jelenik meg, hanem az oldal fő részében felsorolja az adott főcsoporthoz tartozó alcsoportokat.



Ekkor is tudunk lejjebb menni még egy mélységet, mégpedig a nekünk tetsző alcsoport kiválasztásával, azaz rákattintva. Ekkor pedig betöltődik az alcsoporton belüli termékek listája, ahonnan tovább szemezgethetünk.



Ezután a konkrét termék kiválasztásakor, annak adatlapjára ugrunk, ahol – amennyiben rendelkezésre állnak – láthatunk egy nagyobb, illetve 3 kisebb képet, amikre ha ráhúzzuk az egér mutatóját, akkor megjelennek nagyobb méretben is. Lejjebb található még néhány tulajdonság felsorolás formájában, amik segíthetnek eldönteni az érdeklődőknek, hogy megfelel-e nekik az adott termék, vagy sem. Valamint itt említeném meg, hogy navigáció közben folyamatosan változik a középső rész fejlécének tartalma, ahogyan változik a középső részben megjelenített tartalom. Ez is a látogatók könnyebb tájékozódása céljából került az oldalra. Így mindig pontosan tudhatják, hogy hol állnak, valamint szükség esetén visszább tudnak lépkedni az oldal böngészése közben, a fejlécben szereplő szavakra kattintva.



A „Baby Design Rider ké” termék adatlapjának felső része.

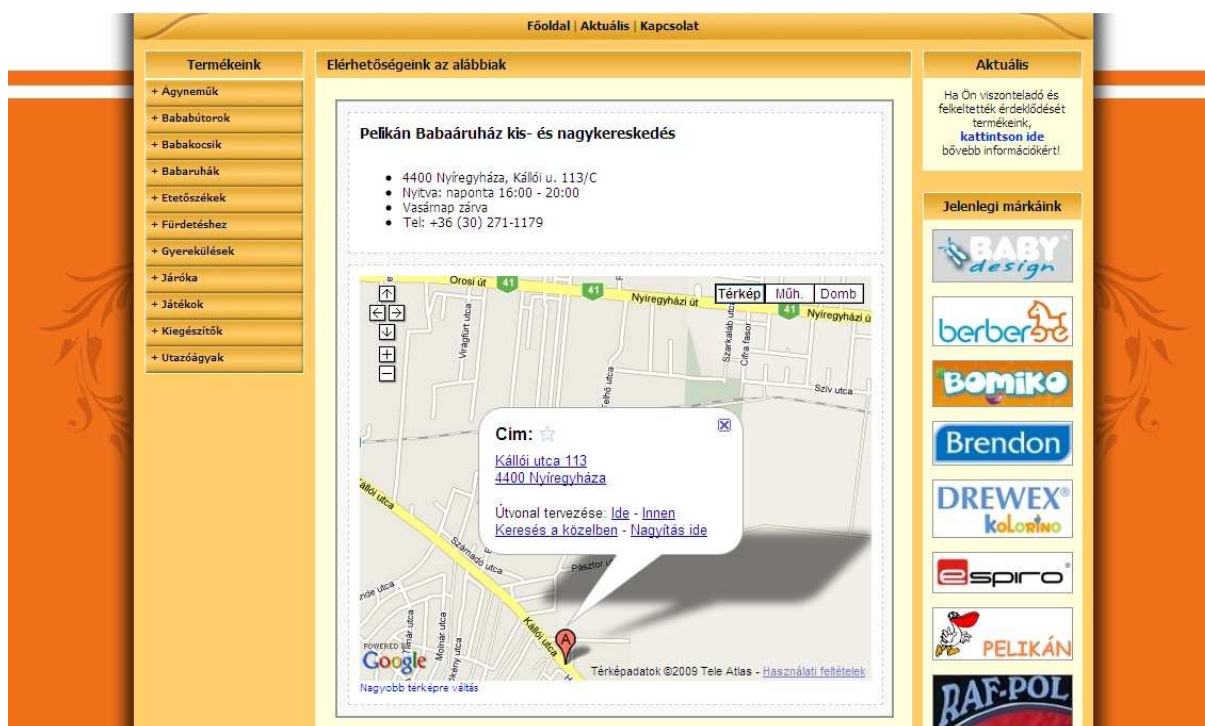
A termékek tallózásának további módja, a jobb oldalon lévő márka logókra való kattintás. Amikor kiválasztunk egy márkát, akkor a középső részen kilistázza nekünk az oldal, a választott márkához tartozó összes alcsoportot. Ez ugyanazt a

nézetet generálja, mintha a bal oldalról választottunk volna egy főcsoportot, csak itt nem főcsoportra, hanem márkára szűkítve kapjuk a listát.



A „Drewex” termékeket tartalmazó alcsoportok megjelenítése

Az oldalon található még továbbá egy a felső menüből elérhető „Kapcsolat” menüpont, ahol egy beágyazott Google Maps komponenst találhatunk. Igen hasznos lehet, mert így a látogatók azonnal tudnak tájékozódni, hogy hol is találják meg a céget, esetleg útvonalat is tervezhetnek a megadott ponthoz, és ehhez el sem kell hagyniuk az oldalt. A Google ezt az elérhető kiegészítést is ingyen adja, mint olyan sok más egyéb hasznos eszközét. Ilyen egyszerű kis apróságokkal sokkal könnyebbé, és barátságosabbá tehetjük az oldalunkat.



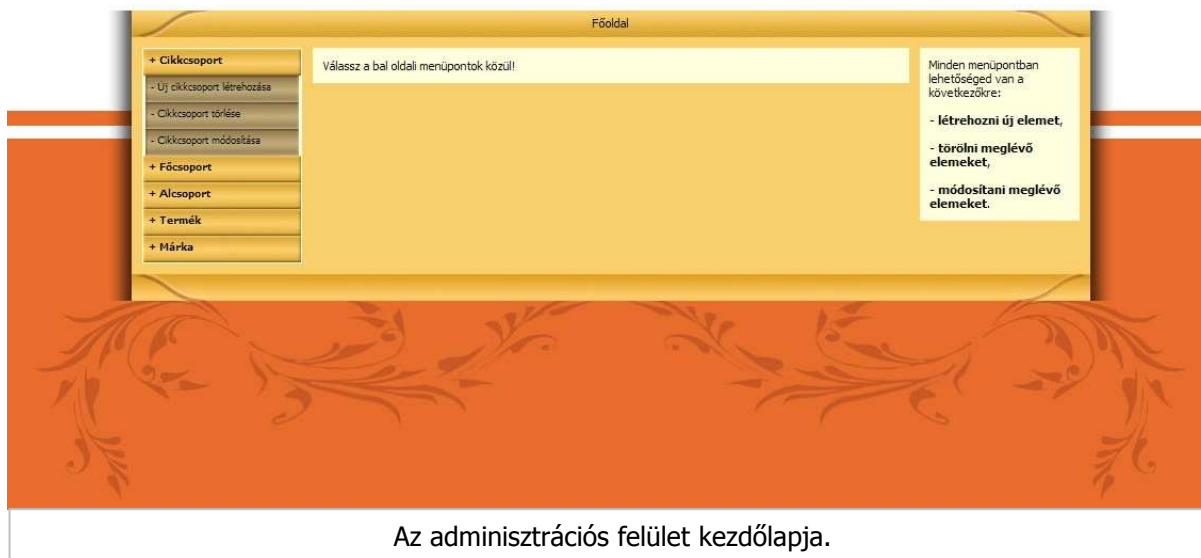
A beépülő Google Maps komponens a Kapcsolat menüpontban.

(tipp: az oldalon található Google Maps komponenst igen egyszerűen beágyazhatjuk. Nem kell hozzá mást tenni, csak ellátogatni a <http://maps.google.com> címre – esetleg a nem régóta működő <http://terkep.google.hu> címre –, és ott megkeresni az adott címet, amit szeretnénk majd megjeleníteni. Ha megvan, akkor a térkép jobb felső sarkánál lévő „Link”-re kattintani, ami egyből ad nekünk egy direkt linket, amit akár elküldhetünk e-mailben is, valamint egy HTML kódba ágyazható másik linket. Amennyiben komolyabban bele mélyednénk, és jobban testre szeretnénk szabni a beágyazott modult, akkor használjuk a Google Maps API-t^[38])

^[38] referenciák és dokumentáció: <http://code.google.com/intl/hu-HU/apis/maps/documentation/v3/>

4.5.2 A weblap adminisztrációs felülete

A weblaphoz egy kezdetleges adminisztrációs felület is kapcsolódik, ahol el lehet végezni az alapvető karbantartási funkciókat. Ezek bővítése, valamint további funkciók hozzáadása tervbe van véve, de további egyeztetésre lesz szükség a megrendelővel.



Az adminisztrációs felület kezdőlapja.

Az oldal felépítése szinte teljesen megegyezik a publikus oldalával. A bal oldali menüben megtalálható az összes termék csoport (cikksorozat, főcsoport, alcsoport, termék, márka), és mindegyikhez lehetőségünk van felvenni új elemet, törölni egy meglévőt, vagy módosítani egyet.



Új termék felvétele párbeszédpanel.

5. Konklúzió

A dolgozat írásakor nem volt céлом egy konkrét példán keresztül bemutatni a teljes weblapkészítési folyamatot, ugyanis az sokkal jobban beszűkítette volna a dolgozat esetleges felhasználási területét. Ugyanakkor igyekeztem a lehető legtöbb helyen apró és egyszerű gyakorlati példákkal alátámasztani a leírtakat az általam készített oldalon keresztül, mert sokban segítheti az anyag megértését. Sokkal inkább szerettem volna egy olyan általános útmutatást adni a weblapkészítés témájában, ahol a tárgyalt témákkal kapcsolatban egy kellően átfogó képet tud kialakítani az olvasó, és ennek fényében tudja eldönteni, hogy az adott dolog éppen megfelel-e az ő kívánalmainak és elvárásainak, vagy sem. Amennyiben jónak tűnik számára a tárgyalt téma (akár a szoftverek, akár a technológiák bármelyike), a további és alaposabb megértéshez kellő számú hivatkozást adtam meg a dolgozatban, hogy alaposabban utána lehessen nézni, és további segítséget lehessen szerezni.

Úgy gondolom, hogy sikerült a kitűzött célt elérnem, mind a dolgozattal kapcsolatban, mind pedig a készített weblappal kapcsolatban. A lap végül a leírtaknak megfelelően született meg, és teljesíti a vele szemben fennálló követelményeket. A dolgozat pedig tömören, kellő rendszerezettséggel, egy általános tudásanyagot biztosíthat a weblapfejlesztéssel ismerkedőknek a jövőben.

6. Köszönetnyilvánítás

Ezúton szeretném megköszönni Pánovics Jánosnak, a témavezetőmnek, hogy segített a dolgozat elkészítésében, és a rugalmas hozzáállását az egész szakdolgozattal kapcsolatban. Valamint a családomnak a támogatást, és a megértő hozzáállást az egyetemen töltött éveim alatt. Továbbá Dr. Bölcskei Andrásnak a példamutató megértést a legelső félévben tanúsított hozzáállásával, és Girhiny Róbertnek a szakmai segítséget az évek alatt a grafikai munkák terén.

7. Irodalomjegyzék

Virginia DeBolt - HTML és CSS: Webszerkesztés stílusosan

Matt Zandstra - Tanuljuk meg a PHP 5 használatát 24 óra alatt!

<http://www.w3.org/>

<http://hu.php.net/>

<http://www.javascriptkit.com/>

<http://www.lynda.com/>

<http://hu.wikipedia.org/>

<http://wikipedia.org/>

<http://phpstudio.hu/forum/index.php>

<http://www.w3schools.com/html>

<http://www.w3schools.com/xhtml>

<http://www.w3schools.com/css>