SZAKDOLGOZAT

Hegedüs Péter

Debrecen

2010

Debreceni Egyetem

Informatikai kar

Portálkészítés
japán szótár és tananyagok számára
PHP nyelven

Témavezető:

Dr. Horváth Géza
egyetemi adjunktus

Készítette:

Hegedüs Péter
programtervező informatikus (BSc)

Orvos Gergő Attila
programtervező matematikus

Debrecen

2010

Tartalomjegyzék

I. Bevezetés

Tim Berners-Lee készítette el az első böngészőt 1990 októberében, és mutatta be az internetben rejlő lehetőséget. Kezdetben a weboldalak célja nem volt más, mint az egyirányú információközlés. Általában statikus oldalakból állt a hálózat, amelyeket ritkán frissítettek az üzemeltetői. Főleg kutatók használták, hogy dokumentumokat oszthassanak meg egymás között. A kiszolgáló webszervereknek annyi volt a feladata, hogy elérhetővé tegyék azokat a HTML fájlokat, amelyek között linkekkel lehetett navigálni. Az oldalak *frame*-ekből álltak, és az egész lap szerkezetét táblázatokból hozták létre. Az átlagos sávszélesség 64-128Kbs volt, így kevés képet használtak, amik ráadásul erősen tömörítve kerültek publikálásra. A web ezen változatát *Web 1.0* néven emlegetik [1].

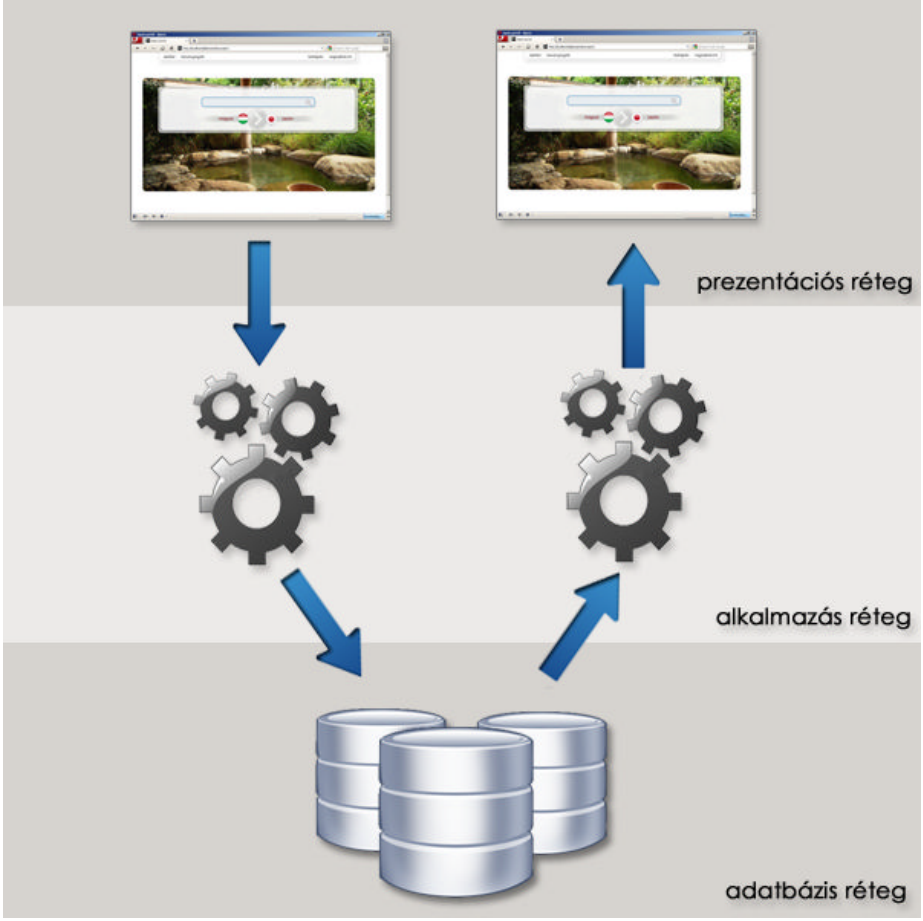
A *Web 1.5*-ként fémjelzett világban megjelentek az első blogok, fórumok, portálok, és egyéb dinamikus tartalommal rendelkező oldalak. Ezeknek az interaktív alkalmazások lehetővé tették, hogy akár minden oldaltöltésre új tartalmat generáljon a felhasználónak, és manipulálja az adatbázisban lévő adatokat. A megjelenés minősége is fejlődött a sávszélesség növekedésével.

A következő lépcsőfok *Web 2.0*-néven került a köztudatba [2] [3]. Azért jelentős lépés a világháló fejlődésében, mert nyit az átlagos felhasználó felé, és erre a közösségre épít. Nem kell informatikusnak lenni az embernek ahhoz, hogy egyszerűen megoszthassa gondolatait az interneten. A felhasználók közösen létrehozhatnak, megoszthatnak információkat egymással, a szolgáltató csak a keretrendszert biztosítja, a tartalmat maguk a felhasználók töltik fel, hozzák létre, osztják meg vagy véleményezik.

Minimális számítógépes ismeretekkel már bárki készíthet saját honlapot tartalomkezelő (*CMS*) rendszerek segítségével (Joomla, Drupal, WordPress, e107). Emellett indíthat blogot (blog.hu), vásárolhat (eBay, Amazon), levelezhet (Gmail, freemail), videót tölthet fel (YouTube), megoszthatja a fotóit (Flickr), zenét hallgathat (Last.fm) és közösségi életet élhet (iwiw, Facebook, Twitter).

Ezeket a lehetőségeket nyújtó oldalakat nevezhetjük webalkalmazásoknak. Úgy is felfoghatjuk, hogy a hagyományos alkalmazások a webre „költöztek”, így a világon bárhonnan, tetszőleges platformról elérhetjük őket.

Szeretném pár szóban összefoglalni a három rétegű alkalmazások lényegét, mert a szakdolgozathoz készített program, és a megvalósítása körüli munka felosztása ezt a felépítést követi.



1. ábra

A webalkalmazások általában három rétegből[4] épülnek fel, amelyek jól elkülöníthetők egymástól *(1. ábra)*.

Prezentációs réteg: Ez felel a megjelenítésért. A felhasználóval teremt közvetlen kapcsolatot, feladata az ügyviteli logika által küldött adatok formázása és megjelenítése. Biztosítja a felhasználó hozzáférését az üzleti logika szolgáltatásaihoz. Sohasem kerül közvetlen kapcsolatba az adatbázissal, és nem végez semmilyen műveletet az adatokon.

Alkalmazás réteg: Itt található az üzleti logika, ami meghatározza az alkalmazás működésének logikáját. Biztosítja az első, és a harmadik réteg között lévő kapcsolatot, de nem rendelkezik az adatok tárolásával, és megjelenítésével összefüggő formációkkal.

Adatbázis réteg: Az adatok tárolását végzi, nem rendelkezik információval azok feldolgozásával vagy megjelenítésével kapcsolatban.

Orvos Gergő Attilával csapatmunkában választottuk a portálkészítést dolgozatunk témájának. Az alkalmazást közösen fejlesztettük, megosztva egymás között a teendőket. Az adatbázis réteggel kapcsolatos tennivalók kivitelezés Orvos Gergő Attila munkájának eredménye, ami részletesebben az ő dolgozatában[5] olvasható. A felhasználói felület megalkotása, pedig az én feladatom volt. A portál motor, és az alkalmazás réteg megvalósítása közös munkát igényelt.

Az volt a szándékunk, hogy egy jól áttekinthető, egyszerűen használható keretrendszert alkossunk, japán szótár, és tananyagok számára. Lehetőséget adva a szavak keresésére, és a dokumentumok közötti böngészésre.

Elsődleges célom, hogy bemutassam egy internetes portál fejlesztésének lépéseit, és az ehhez felhasznált eszközöket. Mivel az oldal megjelenésének kialakításával foglalkoztam, így a felhasználói élmény felől közelítettem meg a feladatot.

II. Japán nyelv

A japán nyelv[6] a tíz legbeszéltebb nyelv közé tartozik, jelenleg is kb. 126 millió ember használja. Ezen szám nagy része egyértelműen Japánban található meg, máshol sem első, sem pedig második nyelvként nem használatos. Eredete egyelőre még nem tisztázott. Közelebbi rokonságot eddig az Okinaván beszélt rjúkjú nyelvvel igazoltak (a két nyelvet és dialektusaikat újabban a japán nyelvek családjának nevezik). A koreai nyelvvel távolabbi rokonságot feltételeznek (nyelvszerkezeti hasonlóságok, csekély számú rokon szó). Korábban gyakran az altaji nyelvcsaládba sorolták, de ez az elmélet nem egyértelműen bizonyítható és ezért nem általánosan elfogadott. Néha pedig az ajnu (Japán őslakosai) nyelvrokonság is felmerül, emellett azonban csak igen kevés érv szól.

Napjainkban egyre népszerűbbé válik mind a japán kultúra, mind pedig az ehhez kötődő japán nyelv is. A Debreceni Egyetem is otthont ad ezeknek a próbálkozásoknak. Az elmúlt évek során a japán napok, események, továbbá a japán nyelvi kurzusok megjelenése és fejlődése is teret engedett a japánt kedvelő emberek kibontakozásának.

Egyre több ember tanulja a nyelvet és fogják még egyre többen a térhódítással. A magyaroknak megkönnyíti a nyelvtanulást, hogy ragozó nyelvről beszélhetünk. Az alábbiakban három írásjel-típust fogok bemutatni, amely a nyelv alapját képezi. Ez a három típus a hiragana, katakana és a kanji[7][8]. Bár külön-külön is használhatjuk őket egy-egy szó leírására, egy anyanyelvű japán mindhárommal tisztában kell legyen, hiszen bárhol szétnézünk Japánban mind a három jelet használják keverve, azaz egyetlen szóban megtalálható mind. Ráadásul azt is figyelembe kell venni, hogy a hiraganákkal nem tudnánk mindent leírni. Ez amiatt van, mert a hiragana és katakana is szótagokból áll, az a,i,u,e,o magánhangzók - illetve valamely mássalhangzóból - összeállított szótagokból.

ts	c
ch	cs
z	dz
y	j
sh	s
s	sz
w	v

A baloldali táblázat egy kis magyarosítás a következőkben használatos táblázatokhoz. Ez csak annyit jelent, hogy az angol táblázatban található egyes mássalhangzók (a táblában a baloldalon) a magyar kiejtésben minek felelnek meg (a tábla jobb oldala). Azért nem kerestem magyar táblázatot, mert a teljes katakana lista csak angol verzióban volt megtalálható (hiányosan pedig nem lenne látható a számbeli különbség, illetve a plusz szótagok). Az angol kiejtést szóba hozva megemlítendő még a

romaji írásmód is. Ez a tulajdonképpeni latin betűs átírása a japán szavaknak. Igencsak hasznos tud lenni, ha az ember csupán szóban használja inkább a nyelvet. Mind a két szótagírás tartalmaz kiegészítő jeleket is. Ezek a jelek a kiejtésre vonatkoznak, így pl. a hosszú magánhangzó jelölésére katakanák esetén hosszú vízszintes vonal.

Szemléletesebben a hiraganák a *2. ábrán* láthatók.

	Monographs (gojūon)						Digraphs (yōon)		
	a	*i*	*u*	*e*	*o*		*ya*	*yu*	*yo*
ø	あ a [a]	い i [i]	う u [ɯ]	え e [e]	お o [o]				
K	か ka [ka]	き ki [ki]	く ku [kɯ]	け ke [ke]	こ ko [ko]	くわ kwa [ka]	きゃ kʲa [kʲa]	きゅ kʲu [kʲɯ]	きょ kʲo [kʲo]
S	さ sa [sa]	し shi [ɕi]	す su [sɯ]	せ se [se]	そ so [so]		しゃ ɕa [ɕa]	しゅ ɕu [ɕɯ]	しょ ɕo [ɕo]
T	た ta [ta]	ち chi [tɕi]	つ tsu [tsɯ]	て te [te]	と to [to]		ちゃ tɕa [tɕa]	ちゅ tɕu [tɕɯ]	ちょ tɕo [tɕo]
N	な na [na]	に ni [nʲi]	ぬ nu [nɯ]	ね ne [ne]	の no [no]		にゃ nʲa [nʲa]	にゅ nʲu [nʲɯ]	にょ nʲo [nʲo]
H	は ha [ha]	ひ hi [çi]	ふ fu [ɸɯ]	へ he [he]	ほ ho [ho]		ひゃ çʲa [çʲa]	ひゅ çʲu [çʲɯ]	ひょ çʲo [çʲo]
M	ま ma [ma]	み mi [mi]	む mu [mɯ]	め me [me]	も mo [mo]		みゃ mʲa [mʲa]	みゅ mʲu [mʲɯ]	みょ mʲo [mʲo]
Y	や ya [ja]		ゆ yu [jɯ]		よ yo [jo]				
R	ら ra [ɾa]	り ri [ɾi]	る ru [ɾɯ]	れ re [ɾe]	ろ ro [ɾo]		りゃ ɾʲa [ɾʲa]	りゅ ɾʲu [ɾʲɯ]	りょ ɾʲo [ɾʲo]
W	わ wa [wa]	ゐ wi [i]		ゑ we [e]	を wo [o]				
˙	ん n [n] [m] [ŋ] before stop consonants; [ɴ] [ɰ̃] [ĩ] elsewhere						っ (indicates a geminate consonant)	ゃ (reduplicates syllable)	ぃ (reduplicates and voices syllable)
	Diacritics (gojūon with (han)dakuten)						Digraphs with diacritics (yōon with (han)dakuten)		
	a	*i*	*u*	*e*	*o*		*ya*	*yu*	*yo*
G	が ga [ga]	ぎ gi [gi]	ぐ gu [gɯ]	げ ge [ge]	ご go [go]	ぐわ gwa [ga]	ぎゃ gʲa [gʲa]	ぎゅ gʲu [gʲɯ]	ぎょ gʲo [gʲo]
Z	ざ za [za]	じ ji [dʑi]	ず zu [zɯ]	ぜ ze [ze]	ぞ zo [zo]		じゃ ja [dʑa]	じゅ ju [dʑɯ]	じょ jo [dʑo]
D	だ da [da]	ぢ (ji) [dʑi]	づ (zu) [zɯ]	で de [de]	ど do [do]		ぢゃ (ja) [dʑa]	ぢゅ (ju) [dʑɯ]	ぢょ (jo) [dʑo]
B	ば ba [ba]	び bi [bi]	ぶ bu [bɯ]	べ be [be]	ぼ bo [bo]		びゃ bʲa [bʲa]	びゅ bʲu [bʲɯ]	びょ bʲo [bʲo]
P	ぱ pa [pa]	ぴ pi [pi]	ぷ pu [pɯ]	ぺ pe [pe]	ぽ po [po]		ぴゃ pʲa [pʲa]	ぴゅ pʲu [pʲɯ]	ぴょ pʲo [pʲo]
V			ゔ vu [v(ɯ)]						

2. ábra

A japán nyelvet ismerve látható, hogy ennyi szótag nem elég minden egyes szó esetén, hiszen pl. a „ti” szótagot tartalmazó szavak már nem írhatóak le. A hiraganákat toldalékok és határozószavak leírására használják. A nyelv alapjainak elsajátításához azonban nagy segítséget nyújt, illetve a kanjik átírása esetén is használják, az adott kanji olvasatának

segítéséhez, erről később még bővebben. A katakanák már egy bővítettebb jelrendszert nyújtanak *(3. ábra)*

	Monographs (gojūon)					Digraphs (yōon)		
	a	*i*	*u*	*e*	*o*	*ya*	*yu*	*yo*
ø	ア a [a]	イ i [i]	ウ u [ɯ]	エ e [e]	オ o [o]			
K	カ ka [ka]	キ ki [ki]	ク ku [kɯ]	ケ ke [ke]	コ ko [ko]	キャ kya [kʲa]	キュ kyu [kʲɯ]	キョ kyo [kʲo]
S	サ sa [sa]	シ shi [ɕi]	ス su [sɯ]	セ se [se]	ソ so [so]	シャ sha [ɕa]	シュ shu [ɕɯ]	ショ sho [ɕo]
T	タ ta [ta]	チ chi [t͡ɕi]	ツ tsu [t͡sɯ]	テ te [te]	ト to [to]	チャ cha [t͡ɕa]	チュ chu [t͡ɕɯ]	チョ cho [t͡ɕo]
N	ナ na [na]	ニ ni [nʲi]	ヌ nu [nɯ]	ネ ne [ne]	ノ no [no]	ニャ nya [ɲa]	ニュ nyu [ɲɯ]	ニョ nyo [ɲo]
H	ハ ha [ha]	ヒ hi [ç i]	フ fu [ɸɯ]	ヘ he [he]	ホ ho [ho]	ヒャ hya [ç ʲa]	ヒュ hyu [ç ʲɯ]	ヒョ hyo [ç ʲo]
M	マ ma [ma]	ミ mi [mi]	ム mu [mɯ]	メ me [me]	モ mo [mo]	ミャ mya [mʲa]	ミュ myu [mʲɯ]	ミョ myo [mʲo]
Y	ヤ ya [ja]	ヰ yi[note 1] []	ユ yu [jɯ]	ヱ ye[note 1] []	ヨ yo [jo]			
R	ラ ra [ɾa]	リ ri [ɾi]	ル ru [ɾɯ]	レ re [ɾe]	ロ ro [ɾo]	リャ rya [ɾʲa]	リュ ryu [ɾʲɯ]	リョ ryo [ɾʲo]
W	ワ wa [wa]	ヰ wi [i]	ヲ wu[note 1] []	ヱ we [e]	ヲ wo[note 2] [o]	ヰャ wya []	ヰュ wyu []	ヰョ wyo []
'	ン n [n] [m] [ŋ] before stop consonants; n [ɴ] [ɰ̃] [ĩ] elsewhere					ー (exemplifies a long vowel)	ヽ (reduplicates syllable)	ゞ (reduplicates and voices syllable)

	Diacritics (gojūon with (han)dakuten)					Digraphs with diacritics (yōon with (han)dakuten)		
	a	*i*	*u*	*e*	*o*	*ya*	*yu*	*yo*
G	ガ ga [ga]	ギ gi [gi]	グ gu [gɯ]	ゲ ge [ge]	ゴ go [go]	ギャ gya [gʲa]	ギュ gyu [gʲɯ]	ギョ gyo [gʲo]
Z	ザ za [za]	ジ ji [d͡ʑi]	ズ zu [zɯ]	ゼ ze [ze]	ゾ zo [zo]	ジャ ja [d͡ʑa]	ジュ ju [d͡ʑɯ]	ジョ jo [d͡ʑo]
D	ダ da [da]	ヂ ji[note 3] [d͡ʑi]	ヅ zu[note 3] [zɯ]	デ de [de]	ド do [do]	ヂャ ja[note 3] [d͡ʑa]	ヂュ ju[note 3] [d͡ʑɯ]	ヂョ jo[note 3] [d͡ʑo]
B	バ ba [ba]	ビ bi [bi]	ブ bu [bɯ]	ベ be [be]	ボ bo [bo]	ビャ bya [bʲa]	ビュ byu [bʲɯ]	ビョ byo [bʲo]
P	パ pa [pa]	ピ pi [pi]	プ pu [pɯ]	ペ pe [pe]	ポ po [po]	ピャ pya [pʲa]	ピュ pyu [pʲɯ]	ピョ pyo [pʲo]

3. ábra

A katakanás jelek lehetővé teszik, hogy akár ezzel a két jeltípussal mindent kifejezzünk, amit akarunk. A katakanák egyik fő szerepe azonban abban rejlik, hogy az egyes külföldi szavakat írjuk le velük. Ez annyit tesz, hogy például a Japánon kívüli országok, külföldi nevek (pl. egy magyar vagy angol személy neve) stb. katakanával kerülnek lejegyzésre. Ebben megtalálhatóak azon szótagok, amelyek a hiraganák esetében, ugyanazon magán- és mássalhangzókkal, valamint új szótagokat is láthatunk. Az újabb „modern” jelek bevezetésére a hirganáknál említett okok miatt volt szükség *(4. ábra)*.

	Modern digraph additions with diacritics (yōon with (han)dakuten)							
	a	*i*	*u*	*e*	*o*	*ya*	*yu*	*yo*
Y		(ユィ)イィ yi []		(ユェ)イェ ye []				
V	(ヷ)ヴァ va []	(ヸ)ヴィ vi []	(ヹ)ヴ vu [vju]	(ヺ)ヴェ ve []	(ヲ゛)ヴォ vo []	ヴャ vya]	ヴュ vyu []	ヴョ vyc []
Sh				シェ she []				
Je				ジェ je]				
Ch				チェ che []				
Sw	(スゥ)スァ swa]	スィ si []	スゥ swu []	スェ swe []	スォ swo []	スャ sya []	スュ syu]	スョ syo []
Zw	(ズゥ)ズァ zwa []	ズィ zi []	ズゥ zwu []	ズェ zwe []	ズォ zwo []	ズャ zya]	ズュ zyu []	ズョ zyc []
Ts	ツァ tsa [	ツィ tsi [		ツェ tse []	ツォ tso []			
Th	テァ tha [	ティ ti []	テゥ thu []	テェ tye []	テォ tho []	テャ tya []	テュ tyu []	テョ tyo []
Dh	デァ dha []	ディ di []	デゥ dhu []	デェ dye []	デォ dho []	デャ dya []	デュ dyu]	デョ dyo []
Tw	(トゥ)トァ twa []	トィ twi []	トゥ tu]	トェ twe []	トォ two]			
Dw	(ドゥ)ドァ dwa]	ドィ dwi]	ドゥ du []	ドェ dwe []	ドォ dwo []			
Hw	(ホゥ)ホァ hwa]	ホィ hwi]	ホゥ hu []	ホェ hwe []	ホォ hwo []			
F	ファ fa []	フィ fi []		フェ fe]	フォ fo []	フャ fya []	フュ fyu []	フョ fyo []
Ry		リィ ryi [		リェ rye []				
W	ゥァ wa []	ウィ wi [	(ゥゥ)ウー wu	ウェ we []	ウォ wo []	ウャ wya []	ワュ wyu []	ウョ wyo []
Kw	(クゥ)クァ kwa []	クィ kwi]	クゥ kwu []	クェ kwe []	クォ kwo []			
Gw	(グゥ)グァ gwa]	グィ gwi]	グゥ gwu []	グェ gwe []	グォ gwo []			
Mw	(ムゥ)ムァ mwa []	ムィ mwi []	ムゥ mwu []	ムェ mwe []	ムォ mwo []			

4. ábra

Az egyszerűbb szótagírás után lássuk a bonyolultabb részt. A kanjikkal a szavak fogalmi részét, például a főneveket, az igetőt írjuk le. Ezek az írásjelek nem japán, hanem kínai eredetűek. Még i.e. a negyedik század körül kerültek be a japán világba, és mivel a japánok akkor még nem rendelkeztek saját írásjelekkel, ezeket a jeleket kezdték használni. Azonban félreértések elkerülése végett, a két nemzet írásjeleinek hasonlósága ellenére, - bár vannak azonos kanjik is - legtöbbször ezeknek más olvasata, továbbá más jelentése is van.

A kanjik már nem szótagokat, hanem teljes szavakat jelölnek. Egyes kanjik esetén felfedezhető valamiféle kialakulási forma, a mostani formához. Így például egy egyszerűsített autó rajzából tovább egyszerűsítve a „kuruma” azaz az autó japán kanjiját kapjuk. Ezek

ugyanakkor csak a tanulást segítik, és ez csak pár kanji esetén használható mivel a legtöbb kanji összetettebb annál, mintsem valamire visszavezethető legyen.

Az egyes kanjiknál kétfajta olvasatról tudunk beszélni. Van kínai olvasatuk (On olvasat) és japán olvasatuk is (Kun olvasat). Van olyan is, amelyik csak az egyik olvasattal rendelkezik, ezeknek többsége a japán kun olvasatú. Ezek értelemszerűen a japánok saját jelei. Az olvasáskor nem könnyű eldönteni, hogy éppen melyik olvasatot kell használnunk. Ehhez van pár szabály, ami megkönnyíti a döntést:

- Ahol csak egy írásjelből áll a szó (egyszerű kanjik esetén) ott a japán olvasatot fogjuk használni
- Toldalékolás esetén szintén kun olvasat lesz szükséges
- Viszont ahol így nincs ragozás, ott általában a kínai on olvasatot használjuk
- Végezetül személynevek esetén a kun olvasat gyakoribb

Előfordulhat az is, hogy ugyanazon kanjival írt szónak több jelentése is van, attól függően, hogy épp melyik olvasatát nézzük. Így pl.: 白鳥 Ez a szó kun olvasattal: shiratori, ami fehér tollú madárt jelent, azonban on olvasatban: hakuchou, ami pedig a hattyú.

Meg kell még említeni, hogy a japán írás esetén, míg számítógépen, vagy nyomtatott változatban nézzük, ott a jelek többféle formában szerepelnek, de nyilván a vonássorrend a kanjik és a kanák esetén nem lényeges. A való életben azonban a szép külalak érdekében külön vonássorrendet kell megtanulni minden egyes kana és kanji esetében *(5. ábra)*. Különösen szép írásjeleket a kalligráfia elsajátításával lehet papírra vetni.

5. ábra

Röviden ennyivel összefoglalható a japán írástudomány pár alapköve. Azonban ennyiből is látható, hogy mindennek az elsajátításához rengeteg gyakorlás és kitartás kell.

III. Fejlesztői eszközrendszer

PHP

A PHP[9][10] egyike a legelterjedtebb webes programozási nyelveknek. Dinamikus, interaktív webhelyek építésére használják a leggyakrabban. Fejlődését mi sem bizonyítja jobban, mint az objektumorientált programozási elvek teljes körűen újjáalakított, és továbbfejlesztett támogatása, valamint a nyelv magasabb szintű XML-támogatásának megvalósítása.

A PHP oldalak elkészítésénél a HTML-t gyakorlatilag csak formázásra használják, ugyanis a teljes funkcionalitása a PHP-re épül. Amikor egy oldalt akarunk elérni, a kiszolgáló először feldolgozza a PHP utasításokat, és csak a kész kimenetet küldi el a böngészőnek, miközben egy ún. interpretert (értelmezőt) használ, amely általában egy külső modulja a web-szervernek.

Olyan webalkalmazások programozását oldhatjuk meg a PHP segítségével, amelyek:

- adatokat jelenítenek meg a legkülönbözőbb forrásokból, például adatbázisból, állományokból, vagy esetleg más weboldalakról;
- interaktív elemeket – például keresési lehetőséget, üzenőfalat – tartalmaznak;
- lehetővé teszik a felhasználók számára különböző műveletek végrehajtását, így például email küldést vagy egy vásárlási tranzakció lebonyolítását.

A PHP script kódot HTML oldalba lehet ágyazni „**<?php**" és a „**?>**" határoló jelek közé. A szintakszisa a C, Java és Perl nyelvekre épül, könnyen megtanulható. A programozók körében elterjedt, *'Helló, világ!'* program az alábbi módon valósítható meg:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
      <head>
         <title>PHP teszt</title>
      </head>

      <body>

      <?php
            echo 'Helló, világ!';
      ?>

      </body>
</html>
```

Népszerűségét növeli, hogy ingyenes, és könnyen tanulható. Tapasztalt programozóknak is hatékonyt eszköz nyújt, ezért érdemes vele megismerkedni.

JavaScript

A JavaScript[11] egy objektum alapú programozási szkript nyelv, ami eredetileg LiveScript néven volt ismert. Főleg weblapokon használnak. Eredetileg Brendan Eich, a Netscape Communications mérnöke fejlesztette ki; neve először Mocha, majd LiveScript volt, később „JavaScript" nevet kapott, és szintaxisa közelebb került a Sun Microsystems Java programozási nyelvéhez. A JavaScriptet először 1997–99 között szabványosította az ECMA „ECMAScript" néven. A jelenleg is érvényes szabvány az ECMA-262 Edition 3 (1999. december), ami a JavaScript 1.5-nek felel meg. Ez a szabvány egyben ISO szabvány is.

Uniform Server

Az Unifom Servert[12] elsőként Taras Slobodskyy fejlesztette alkalmazásként az ügyfelei számára, de később egy nyílt forráskódú és teljesen ingyenes projekt lett belőle. Ez a tökéletes rendszer mind a kezdők, mind pedig a haladó fejlesztők számára, egy teljesen felszerelt webszerver, amit akár pendrive-on is magával vihet az ember, vagy akár egy kamera flash meghajtóján is tárolhatjuk. A tökéletes prezentációs eszköz.

Az Uniform Server egy WAMP csomag, amely lehetővé teszi, hogy egy szervert futassunk bármely MS Windows operációs rendszer alatt. Kisméretű és hordozható, és feltelepíthető, mint eladási/élő szerver is. A fejlesztők arra is használhatják, hogy teszteljék a PHP, MySQL, Perl vagy akár Apache HTTPd szerver alatt készített alkalmazásaikat.

MySQL

A MySQL[13][14] egy többszálú, többfelhasználós, SQL-alapú relációs adatbázis-kezelő szerver. A szoftver eredeti fejlesztője a MySQL AB cég, amit a Sun felvásárolt 2008 januárjában. A dolog érdekessége, hogy utóbbit felvásárolta az Oracle Corporation két évvel később, így a MySQL is Oracle tulajdonba került.

Az MySQL az egyik legelterjedtebb adatbázis-kezelő, aminek egyik oka lehet, hogy a teljesen nyílt forráskódú LAMP (Linux–Apache–MySQL–PHP) összeállítás részeként költséghatékony és egyszerűen beállítható megoldást ad dinamikus webhelyek szolgáltatására.

Elérhetősége programnyelvekből

Egyedi illesztőfelületekkel az adatbázis-kezelő elérhető C, C++, C#, Delphi, Eiffel, Smalltalk, Java, Lisp, Perl, PHP, Python, Ruby és Tcl programozási nyelvvel. Egy MyODBC nevű ODBC interfész további, ODBC-t kezelő nyelvek számára is hozzáférhetővé teszi az adatbázis-kezelőt. A MySQL számára az ANSI C a natív nyelv.

Adminisztrációja

A MySQL adatbázisok adminisztrációjára a mellékelt parancssori eszközöket (mysql és mysqladmin) használhatjuk. A MySQL honlapjáról grafikus felületű adminisztráló eszközök is letölthetők: MySQL Administrator és MySQL Query Browser. Széles körben elterjedt és népszerű adminisztrációs eszköz a PHP nyelven írt, nyílt forráskódú phpMyAdmin.

Platformok

A MySQL az alábbi platformokon futtatható: AIX, BSDi, FreeBSD, HP-UX, Linux, Mac OS X, NetBSD, 0Netware, OpenBSD, OS/2 Warp, QNX, SGI IRIX, Solaris, SunOS, SCO OpenServer, SCO UnixWare, Tru64, MacOSX 10.4,Windows 95, Windows 98, Windows NT, Windows 2000, Windows XP és a Windows frissebb verziói.

CSS

A CSS[15][16] (***C****ascading* ***S****tyle* ***S****heets*) egy stílus leíró nyelv. Leggyakrabban HTML, xHTML és XML formátumú struktúrált dokumentumok megjelenítésére van hatással.

Ezekkel a stíluslapokkal egyszerűen alakíthatjuk az oldal megjelenítését, meghatározhatjuk, hogy bizonyos HTML tagek (címsor, táblázat, paragrafus... stb) hogyan nézzenek ki. Több stíluslapot, meghatározást is megadhatunk egyszerre, és egy stílus több elemre is érvényes lehet. Az oldal hierarchiájának megfelelően öröklődnek a stílus definíciók. Nagy segítséget nyújt a CSS fájlok elemzéséhez, valós időben való szerkesztéséhez a Firebug[17] program, ami Mozilla Firefox böngésző egyik kiegészítője.

Történelem

A HTML nyelvet fejlesztők elsődleges célja a tartalomközlés volt. Egy idő után megnőtt az igény arra, hogy a nyomdai megjelenítéshez hasonlóan lehessen befolyásolni a weboldalak kinézetét. A HTML szabványt kiegészítették ezekkel a lehetőségekkel.

A nyelv egyre bonyolultabb lett, a weboldalak forrásának nagy része már nem a tartalomról szólt, hanem a megjelenítésről. Megnőtt a méretük, így lassabban lehetett őket letölteni, ráadásul kompatibilitási problémák léptek fel, így a különböző böngészőkben nem volt egységes a kinézetük.

Ezeknek a problémáknak a kiküszöbölésére jött létre a CSS szabvány. Rugalmas, és egyre inkább kiforrottabb megoldás. Sokkal rövidebbé és szabadon kezelhetővé teszi a HTML dokumentumokat.

A CSS1 specifikációja elég régóta létezik, hiszen 1996. decemberében látott napvilágot, de a teljes megvalósításra körülbelül 3 évet kellett várni. A Microsoft Internet Explorer 5 volt az első böngésző, ami majdnem teljesen támogatta a szabványt. Az 1998 májusában megjelent CSS2 verzió ajánlásait még 2005-ben egyetlen böngésző sem tudta teljesíteni. Így a fejlesztők kénytelenek voltak az összes jelentősebb platformra optimalizálni a kódjukat. A hibákból tanulva elkészítették a CSS2.1 javaslatot, ami az előző verzióból átveszi a működő részeket, és azokat, amiket egyetlen böngésző se valósított meg átdolgozták, vagy törölték.

Nem minden böngésző támogatja még a CSS3-at, és amelyek igen, azok sem teljes körűen. Annál is inkább, mert – akárcsak a HTML 5 esetében – nem kész szabványról, hanem egyelőre csak vázlatról van szó, amelyet jelenleg is formálnak, finomítanak.

Ennek fényében megpróbáltam olyan stíluslapot létrehozni, ami minden platformon ugyanolyan formában jelenik meg. Csak olyan elemnek adtam szűkebb körben támogatott attribútumot, amiről tudtam, hogy a megjelenítés minőségén nem változtat. Ilyen például a szöveges tartalom mögött megjelenő halvány árnyék.

Stílusok rangsorolása

Több stíluskészlet elhelyezhető az xHTML lapon, így van néhány alapelv arra vonatkozóan, hogy a rangsorolt stíluslapok hogyan működnek. Erre azért van szükség, hogy a stílusok közötti esetleges ütközéseket egyértelműen feloldja.

Ennek vizsgálata során fontos alapelv a *szűkítési*, és *öröklési* tényező.

Az *öröklés* azon a tényen alapszik, hogy a dokumentumban az elemek egymásba ágyazhatók, és a köztük lévő viszony hasonló ahhoz, mint amilyen az ős és leszármazott között van.

A *szűkítés* azt teszi lehetővé, hogy a stílusoknak súlya (más néven fontossága vagy rangja) legyen, mégpedig egy adott stílus pontossága alapján. A szűkebb (pontosabb) stílus súlya nagyobb, mint a tágabbé, ezért a böngésző azokat fogja használni a formázáshoz.

Ha elegendő időt szánunk a rangsor, valamint az öröklés és szűkítés alapelvének megismerésére, könnyebben megértjük, hogyan, hol és mikor kell stílusokat felvennünk annak érdekében, hogy az xHTML lapok úgy jelenjenek meg, ahogy mi szeretnénk.

Böngésző stílus

Mindegyik böngésző saját stíluskészlettel rendelkezik. Ezek a stílusok adják az alapértelmezett megjelenítési tulajdonságokat. A böngészőstílus, a rangsor elején helyezkedik el.

Felhasználó stílus

A böngészők lehetővé teszik, hogy a felhasználó saját stílusbeállítást adjon meg.

Külső stílus

A böngésző stílusokra, és a felhasználói stílusokra a weblap készítőjének nincs ráhatása. Az első olyan lehetőséget, amelyeket mi, mint tervezők vihetünk be a rangsorba, a külső stíluslapok használata jelenti.

Beágyazott stílus

A stílusok elhelyezhetők magának az xHTML dokumentumnak a fejlécében is. Az ilyen stílusokat belső stílusoknak, vagy beágyazott stílusoknak nevezik. A beágyazott stílusok csak arra a dokumentumra érvényesek, amelyben elhelyezték őket. A beágyazott stílusok használatának, csak akkor van értelme, ha különböző weblapot, vagy egylapos dokumentumot készítünk., valamint akkor, ha külső stíluslap kapcsolódik ugyan a weblaphoz, de kismértékben szeretnénk valamit változtatni, de csak a stíluslaphoz kapcsolódó weblapok egyikén.

Apache

Az Apache HTTP Server[18][19] (továbbiakban Apache) egy robosztus, nyílt forráskódú web kiszolgáló alkalmazás. Nagy szerepet játszott a World Wide Web elterjedésében. A fejlesztők célja egy olyan webszerver program létrehozása, és fejlesztése, ami biztonságos, és megállja a helyét az üzleti környezetben.

Statikus, és dinamikus oldalak kezelésére egyaránt alkalmas, de nem csak weboldalak, hanem tetszőleges tartalom megosztására is használják. A központi maghoz kapcsolódó kiegészítők segítségével valósítható meg például a Perl, Python, és PHP nyelvek támogatása. További kiegészítőkkel irányíthatjuk a hitelesítést, a logolást, és a mod_rewrite segítségével az URL átírást.

HTML-ről röviden

A HTML (*HyperText Markup Language*) leíró nyelvet a weboldalak készítéséhez fejlesztettek ki, és mára már internetes szabvánnyá vált a W3C[16] támogatásával. Az xHTML[20] (*Extensible HyperText Markup Language, azaz kiterjesztett HTML*) a HTML-től annyiban különbözik, hogy XML alapokra épül, emiatt jóval szigorúbb szabályoknak kell megfelelnie.

Ennek köszönhetően jól formázott, valamint egyszerűen feldolgozható. Lényeges, hogy elkülönítsük az adatot a megjelenítésért felelős stílusoktól.

Egy (x)HTML állomány három fő részre bontható:

- A dokumentum típus definíció (DTD) az állomány legelején:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

- a HTML fejléc <head>, ami technikai és dokumentációs adatokat tartalmaz, melyeket az internet böngésző nem jelenít meg, tehát átlag felhasználó ezeket nem látja.

- a HTML törzs <body>, amely a megjelenítendő információkat tartalmazza.

Egy xHTML oldal felépítése így néz ki:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
    <head>
      <meta http-equiv="content-Type" content="text/html; charset=utf-8" />
      <title>Oldal címe</title>
      <!-- További fejléc információk -->
    </head>
    <body>
    Az oldal tartalma
    </body>
</html>
```

Subversion

A Subversion (SVN) egy verziókezelő rendszer. A CollabNet Inc. indította 2000-ben azzal a céllal, hogy a legkompatibilisebb utódja legyen a népszerű Concurrent Versions System (CVS)-nek.

A verziókezelő rendszerek lényegét sok oldalról meg lehet közelíteni. Legegyszerűbb egy olyan háttértárként gondolni rá, amelynek tartalmához megfelelő jogosultság birtokában hozzáférhet a felhasználó. Ezt az SVN tárhelyet *repository* nevezik, ami „emlékszik" a tárolóba másolt fájlok korábbi verzióira.

A rendszer egy pozitív egész számot (*revision*) rendel a fájlokhoz, ami minden egyes módosítás után eggyel nő. A fájlok tárolóba való töltését *commit*-nak, és azok elérését *checkout*-nak hívják. A *commit*-okhoz egy rövid szöveges leírást lehet és érdemes rendelni, így később megtalálhatóak az egyes változtatások (pl. „hozzá adtam az xy dokumentumot", „töröltem xzy-t mert elavult", „elgépelés javítása", stb.). Az egyes állapotokra ezután ezekkel a számokkal lehet hivatkozni.

A verziókezelő rendszereket főleg csoportmunkára találták ki, de sok mindenre használhatóak. Hasznos lehet akkor is, ha egyedül dolgozunk. Egyszerűen készíthetünk fontosabb dokumentumainkról *backup*-ot, így ha elvesztjük őket, akkor is könnyedén visszaállítható az utolsó mentett állapot.

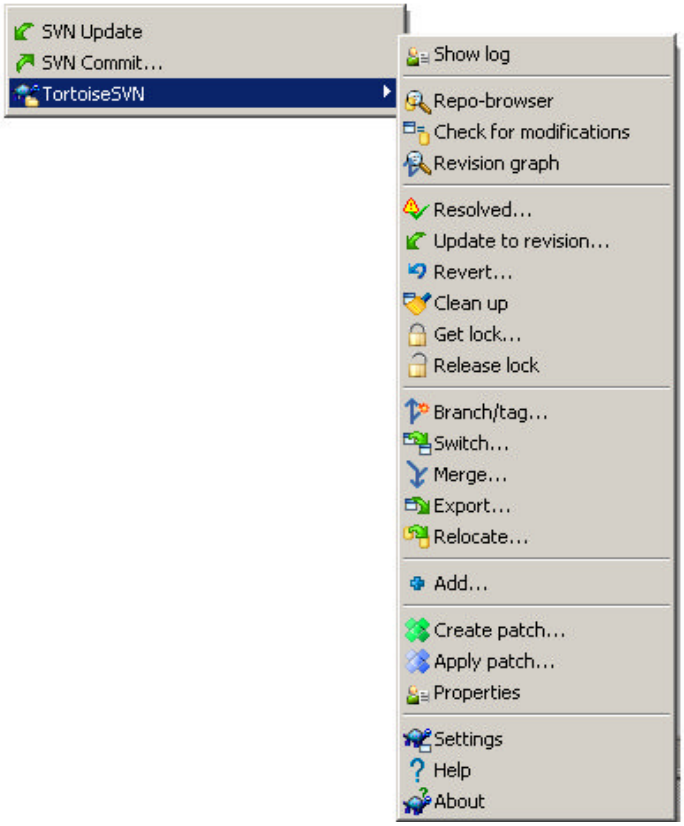
Általában forráskódok tárolására használják. A fejlesztés során rengeteg segítséget nyújt a verziókezelő rendszer. Ha többen ugyanazon a fájlon dolgoznak, akkor lehetőséget ad arra, hogy összefésüljék (*merge*) a sorokat.

A *commit* parancsot atominak tekinthetjük, így nem fordulhat elő, hogy két fejlesztő egyszerre töltse fel, és írja felül a másik változtatásait. Ha mégis egyszerre sikerül a *commit* parancsot kiadni, akkor a korábban beérkező utasítást hajtja végre a rendszer, és konfliktus lép fel. Aki később akarta a módosítását véglegesíteni, annak azt fogja jelezni a rendszer, hogy nem volt friss helyi változata a fájlnak.

A verziókezelők rengeteg információt tárolnak, és a korábbi változatokhoz is hozzáférhetünk, ha valamit elrontottunk, vagy meg akarunk nézni egy régebbi verziót. Le tudjuk kérdezni, hogy egy adott fájlhoz ki nyúlt utoljára, és milyen változtatásokat hajtott végre.

Az alkalmazás fejlesztése során egy ingyenes svn tárolót [21] vettünk igénybe. A *repository*-ba való sikeres feltöltés esetén egy e-mailt küld a rendszer a projektben dolgozó fejlesztőknek. A levél tartalmazza a frissített fájlok listáját, így könnyen elkerülhető a konfliktus kialakulása.

A tortoise svn, programot [22] használtam a verziókezeléshez. Ez a Windows kliensalkalmazás egyszerűbbé teszi a verziókezelést. Feltelepítése után beépül az Explorerbe. *(6. ábra)*



6. ábra

IV. Az alkalmazás bemutatása

Tervezés

A weblap készítése során érdemes előre gondolkodni. Nagy hibát követhetünk el, ha nem vagyunk eléggé előrelátóak. Érdemes mindent időben megtervezni, így megspórolva rengeteg felesleges munkát. Mérlegelnünk kell, hogy milyen weboldalt szeretnénk készíteni. El kell döntenünk, hogy van -e szükségünk adatbázisra, és ha igen, akkor ennek a felépítését jól meg kell tervezni. A funkciókat alaposan meg kell fogalmazni, és ki kell találni, hogy ezeket milyen eszközökkel szeretnénk megvalósítani.

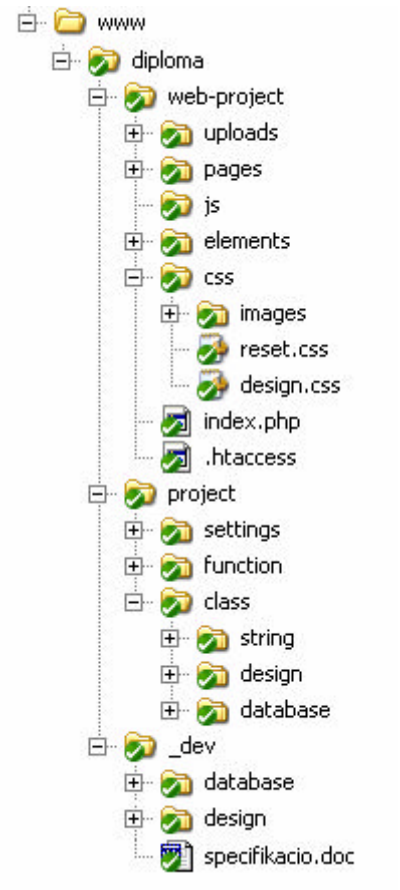
Jelen esetben egy olyan keretrendszer készítése a cél, amely tartalmaz egy japán szótárat, és tananyagok tárolását teszi lehetővé. Mindemellett lehetőséget ad szavak keresésére, és a dokumentumok közötti böngészésre.

Ennek megoldása során át kell gondolnunk, hogy kik fogják használni, és milyen funkciókra lesz szükség. Az oldal megjelenésével kapcsolatosan lényeges, hogy jól áttekinthető legyen. Továbbá fontos az is, hogy minden oldalon a navigációs megoldások, és a grafikák megegyezzenek, hogy a felhasználót ne zavarjuk össze, és egy logikus rendszerben, egyszerűen ki tudja használni a weboldalban rejlő lehetőségeket.

Programozás szempontjából úgy kell megírni az oldalt, hogy különböző böngészőkkel való megtekintés esetén is egyformán jelenjen meg. Ennek érdekében az alábbi programok legújabb verziójával teszteltem:

Mozilla Firefox, Microsoft Internet Explorer, Google Chrome, Opera.

Ha ennél is alaposabbak szeretnénk lenni, de nincs arra kapacitásunk, hogy az összes böngészőn, és operációs rendszeren megtekintsük az alkalmazást, akkor érdemes olyan szolgáltatást igénybe venni, ami leveszi a vállunkról ezt a feladatot [23].



Az érdemi munka megkezdése előtt átgondoltuk, hogy milyen mappaszerkezetbe lenne érdemes elhelyezni a dokumentumainkat. A jobb áttekinthetőség, és a közös fejlesztés megkönnyítése érdekében a bal oldalon látható elrendezést alakítottuk ki.

Három fő mappa található a gyökér könyvtárba. A *web-project* tartalmazza a weboldalt, és az *index.php*-t. Ezen belül az *uploads*-ba kerülnek a tananyagokhoz tartozó állományok. A *pages*-be az aloldalak (például a szótár, vagy az admin oldal), az *elements*-be pedig az ezeket felépítő kisebb alkotó elemek (például a menü, vagy a lábléc).

A *js* a javascript-eknek lett fenntartva, a *css*-be pedig a stílusleírások vannak, jól elkülönítve a grafikai elemektől, amik az *images* almappáva kerültek.

A *project* mappában helyeztük el a kód egy részét. A *settings*-ben tároljuk az alapvető beállításokhoz használt *properties.php*-t. A *function* és *class* alkönyvtárak értelemszerűen a saját függvényeinknek, és osztályainknak ad helyet.

A *_dev* alatt találatóak a tervezés során felhasznált, fájlok, amik nem kerültek publikálásra.

_dev/database: Az adatbázishoz kötődő fájlok (adatbázis terv, sql utasítások) találhatóak benne.

_dev/design: A felhasználói felület kialakítása során készült képernyőtervek, és vázlatok gyűjtőhelye.

Felhasználói felület

A felhasználói felület megtervezésekor figyelembe kell venni, hogy nem azonos méretű képernyőn fog megjelenni. Az oldalt legalább 1024*768 pixeles megjelenítésre terveztük, mert elhanyagolható azoknak a látogatóknak a száma, akik ennél kisebb felbontású monitorral böngésznek PC-ről.

Lényeges szempont volt, hogy a rendszer legyen egyszerű, és könnyen átlátható. Egyszerűen lehessen kezelni, hiszen nem csak informatikusok fogják használni, hanem kezdő felhasználók is. Nem szabad túlbonyolítani a kezelő felületet, nehogy elrémítsünk bárkit is az alkalmazás használatától.

Az oldalakat úgy alakítottuk ki, hogy a funkciók mindenhol azonos megoldással legyenek elérhetők. Nem szabad, hogy a kinézet a felhasználhatóság rovására menjen. A szöveges részeknél fontos a jó olvashatóság. Az oldalra belépő látogatónak egy pillanat alatt fel kell ismernie az oldal célját. Egy jó design-nak köszönhetően a tartalom, világosan, meghatározott hierarchiában jelenik meg, mely segíti az olvasót, hogy könnyedén kiválogathassa a számára fontos információkat.

Minden lap három részből áll. A fejléc tartalmazza a menü elemeit, amelyek egyik felét balra zárva, másik felét, pedig jobbra zárva jelenítek meg *(7. ábra)*. A menüpontok között nagyobb távolság segíti, hogy azok ne folyjanak össze. A menüsor jól látható helyen, az oldal tetején található. Ennek segítségével navigálhatunk.

7. ábra

Minden oldalon változatlan formában jelenik meg, ahogy a lábléc is. A kettő között található tartalom dinamikusan változik az url-től függően. Erről bővebben szó lesz később.

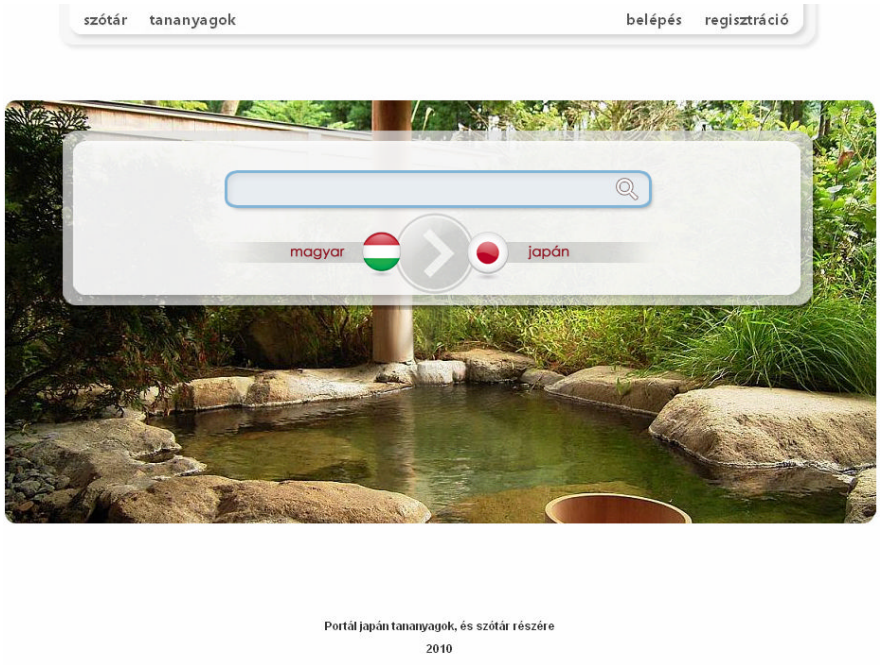
./web-project/elements/header.php

```
<!-- Fejléc -->

<div class="menuContainer">
    <div class="menuLeft">
        <a title="szótár" href="szotar">szótár</a>
        <a title="tananyagok" href="tananyag">tananyagok</a>
    </div>

    <div class="menuRight">
        <a title="belépés" href="belepes">belépés</a>
        <a title="regisztráció" href="regisztracio">regisztráció</a>
    </div>
</div>
```

./web-project/elements/footer.php

```
<!-- Lábléc -->

<div class="footer">
    Portál japán tananyagok, és szótár részére <br /><br /> 2010
</div>
```

A kezdő képernyő *(8. ábra),* ami fogadja a látogatókat, elég egyértelműen sugallja az oldal funkcióját. A lap közepén található keresőmező segítségével azonnal használhatjuk a szótárat.



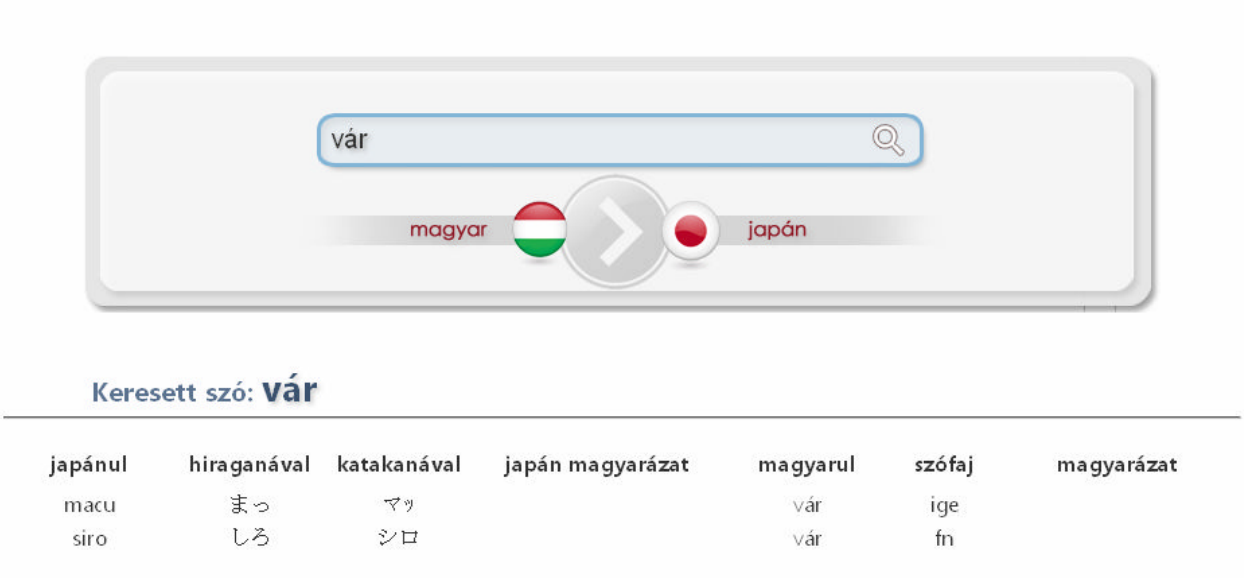
8. ábra

Szótár

A szótár menüben lehetőségünk van szavakat keresni. Nem hiszem, hogy ez a funkció hosszabb magyarázatot igényelne. Természetesen magyar-japán *(10. ábra)*, és japán-magyar *(11. ábra)* irányban is tudunk keresni, amit a beviteli mező alatti gombbal *(9. ábra)* változtathatunk meg.

9. ábra

Miután beírtuk a kifejezést, amire kíváncsiak vagyunk, feldolgozza az alkalmazás a kérésünket. Ennek eredményét táblázatba rendezve kapjuk. Ha nincs eredménye, akkor a „*A keresett szó nem szerepel az adatbázisban.*” üzenet olvasható.



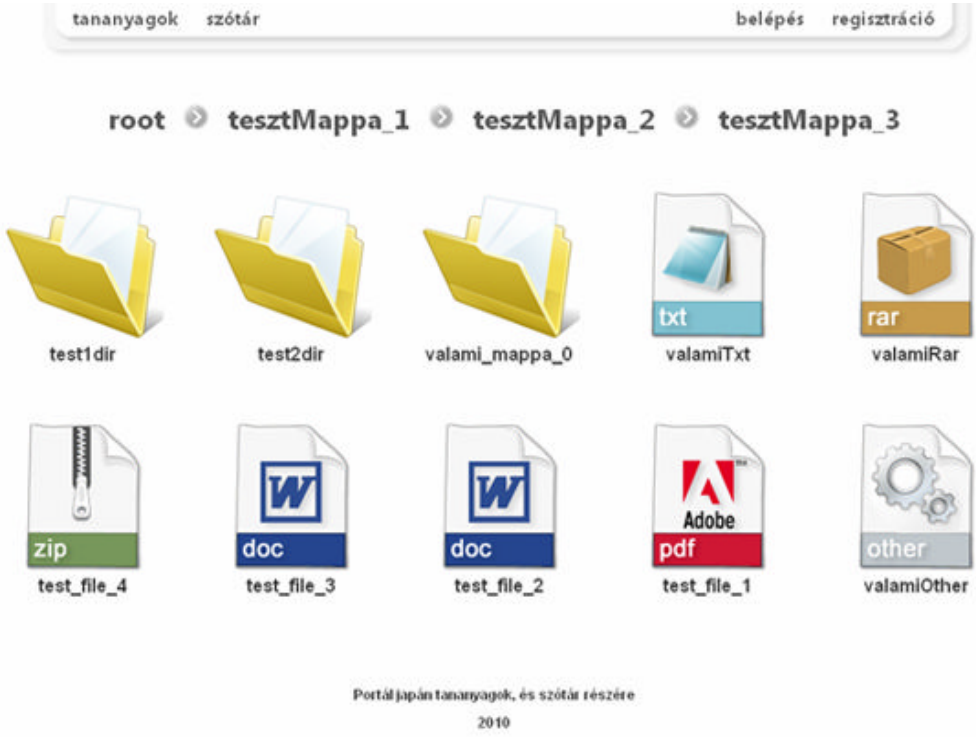
10. ábra


siro
magyar
japán
Keresett szó: siro
japánul hiraganával katakanával japán magyarázat magyarul szófaj magyarázat
siro しろ シロ vár fn
siro しろ シロ kastély fn
siro しろ シロ fehér fn szín

11. ábra

Tananyagok

A weboldalon japán tananyagok böngészhetők *(12. ábra)*, és letölthetők *(13. ábra)*. Ennek az oldalnak a megjelenítését úgy próbáltuk megtervezni, hogy könnyen áttekinthető, és logikus felépítésű legyen.



12. ábra

13. ábra

Az állományok fizikailag egy mappában találhatók a merevlemezen, a közöttük lévő hierarchiát a hozzájuk tartozó adatbázis rekordok írják le. Minden fájlnak van egyedi

azonosítója, és szülő eleme, így felépíthető egy fa. Ennek a fának egy adott csúcsához tartozó leszármazott elemeit listázzuk ki. A böngészés megkönnyítésére egy navigációs menü *(14. ábra)* található a fájlok fölött, amire tekinthetünk úgy is, mint a gyökértől az aktuális elemig tartó útra.

14. ábra

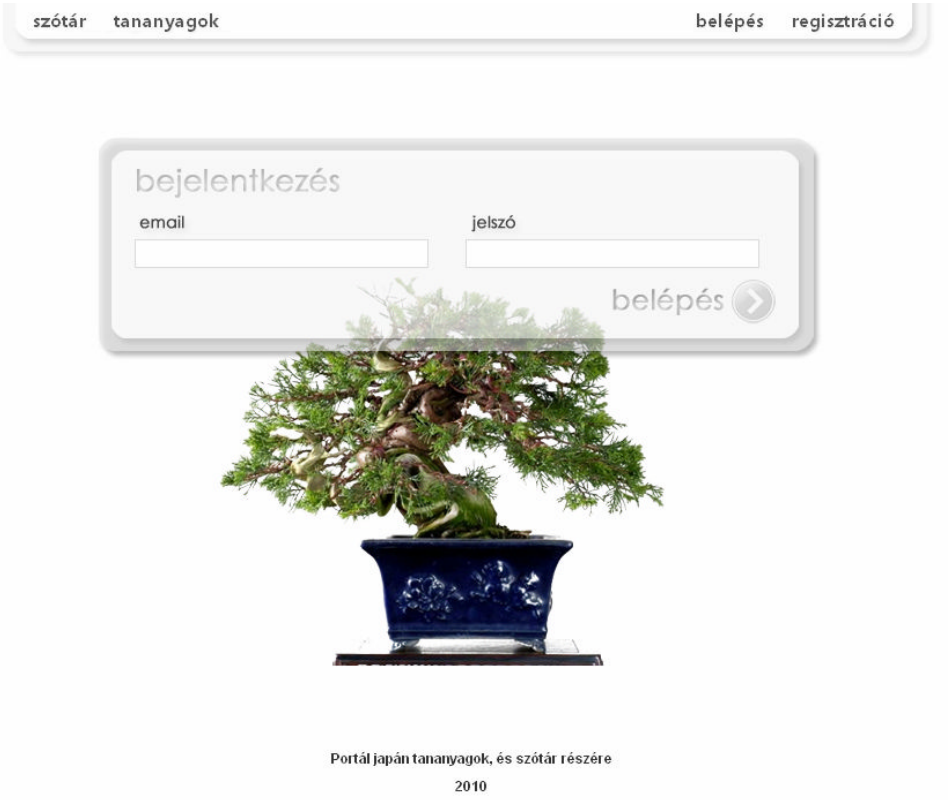
A fájlok felsorolásánál fontos szempont volt az egységes, de mégis jól elkülöníthető megjelenés. Úgy próbáltam meg létrehozni ezeket az ikonokat *(15. ábra)*, hogy első pillantásra egyértelmű legyen, hogy az adott ábra, milyen fájltípust szimbolizál. Természetesen ez a lista újabb képpel bővíthető.

15. ábra

Adminisztráció

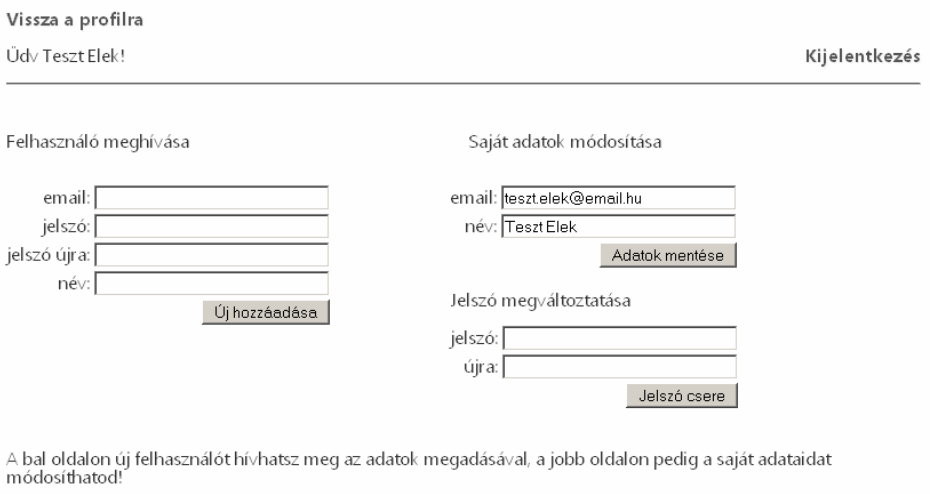
Az adminisztrátori jogkörbe eső feladatok az alábbiak: *bejelentkezés, saját adatok módosítása, felhasználó meghívása, szótár feltöltése, tananyagok kezelése.*

Bejelentkezés: A belépési oldalon *(16. ábra)*, az e-mail cím és jelszó helyes kombinációjának megadásával az adminisztrációs oldalra jutunk.



16. ábra

Saját adatok módosítása: A adminisztrátoroknak lehetőségük van a saját adataik megváltoztatására (e-mail cím, név), és jelszavuk cseréjére. Utóbbinál kétszer kell megadni az új jelszót, minimálisra csökkentve ezzel az elgépelésből fakadó kellemetlenségeket. *(17. ábra)*



17. ábra

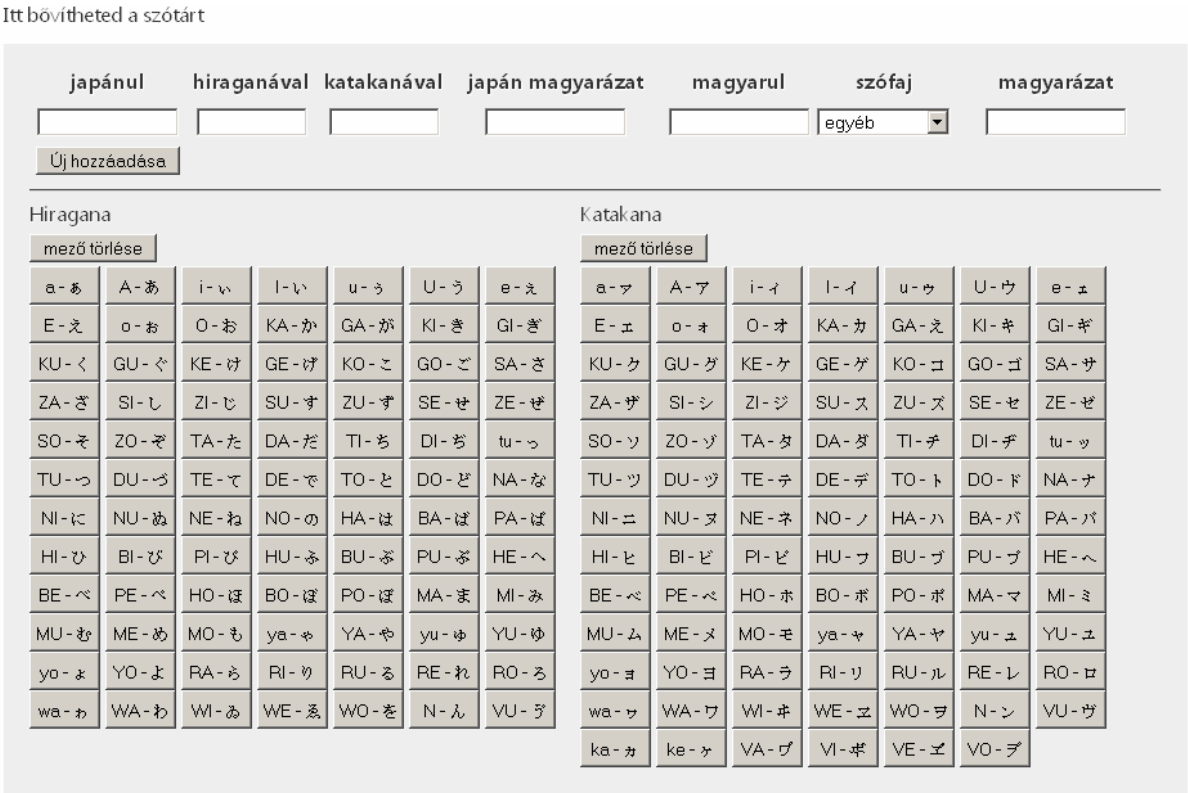
Felhasználó meghívása: A weboldalra nem lehet közvetlenül regisztrálni. Új tagot csak az aktív felhasználók hívhatnak meg, a leendő tag adatainak (név, jelszó, e-mail cím) megadásával.

Tananyagok kezelése: A szótár feltöltésével hasonló módon lehetőség van a tananyagok módosítására *(18. ábra)*. A könyvtárszerkezetben lépkedve módunkban áll fájlt feltölteni, újabb könyvtárat létrehozni, vagy az aktuálisat törölni. Ha hiba történik, akkor az alkalmazás egy rövid szöveges üzenet formájában tájékoztatat bennünket.

18. ábra

Szótár feltöltése: A szótár adatbázisának bővítését végezhetjük el, ha bejelentkezett állapotban meglátogatjuk szótár menüpontot. Ezt a műveletet megkönnyíti a beviteli mezők alatt

megjelenő hiragana, és katakana táblázat *(19. ábra)*, aminek segítségével egyszerűen beilleszthetőek a speciális japán karakterek. A szófaj kiválasztását egy legördülő menü segíti.



19. ábra

Rövid webcímek kezelése

Elég sok érv áll amellett, hogy miért érdemes kereső barát url-eket használni. Talán a legkézenfekvőbb az, hogy a rövid címek könnyen megjegyezhetőek, és sokkal több helyre kihelyezhetőek. Ha e-mailben tesszük közzé a hivatkozást, akkor nem kell tartani attól, hogy a webcím kettétörik a sortörésben. Egy ilyen rövid webcím könnyen elfér névjegykártyán, hirdetésekben, akár rádióban bemondva is.

A tartalommal zsúfolt oldalakon, amelyeken sok linket tesznek közzé, a hosszú webcímek kiírása jelentősen megnöveli az oldal méretét, mely a kiszolgálására és megjelenítésére fordított időt is növeli. Ha rövidebb webcímeket használunk, oldalunkon belüli kapcsolataink is kisebb HTML lapot eredményeznek. A méretcsökkenés különben a kereső optimalizálásban is segít, hiszen az oldal betöltési sebességének csökkenése is jótékony hatással van annak rangsorolására.

A programozó munkáját is elősegíti, hiszen ha rövid webcímeket szeretnénk használni, akkor jól át kell gondolnunk, hogy mit hova teszünk, annak érdekében, hogy a címek sokáig használhatóak legyenek. Ez ösztönöz minket arra, hogy átgondoljuk tartalmunkat, és egységes kialakítást hozzunk létre. A rövid címekkel könnyebb lesz a dolgunk, hiszen kezelésük programunkban is egyszerű.

A Google webmestereknek szóló dokumentációja tartalmazza, hogy a dinamikus oldalak indexelését visszafogottan végzik, nehogy összeomlasszák a mögöttük futó programokat a kérésáradattal. Nem tudni biztosan, de a közhiedelem szerint a webcímekben megjelenő minden újabb dinamikusságra utaló karakter (?, &) az indexelés sebességének csökkenését jelenti.

Fontos megemlíteni a keresőoptimalizálás szempontjából, hogy például a Google a webcímekben lévő kulcsszavakat is indexeli. Tehát ha olyan címet alakítunk ki, mint *http://pelda_url.hu/cirmos/cica.html* akkor ez az oldal már a címe miatt is jó eséllyel bukkan fel a „*cirmos cica*” keresésre. A tapasztalatok szerint a kötőjellel elválasztott szavak külön szónak számítanak az indexelésben.

Működés közben

Ha betöltünk egy url-t, akkor 404-es hibát ad a kiszolgáló, mert nem találja a megadott oldalt. A. htaccess fájlban megadhatjuk, hogy ilyen esetben melyik oldal jelenjen meg. Ellenkező esetben a jól ismert hibaüzenet fogadna minket *(20. ábra)*.

20. ábra

./web-project/.htaccess

```
ErrorDocument 404 /diploma/web-project/index.php
```

Ezzel azt érjük el, hogy hiba esetén átadjuk a feldolgozást az *index.php*-nak. Emellett érdemes beállítani a fejlécet, és tudatni a böngészővel, hogy az oldal mégis „létezik”.

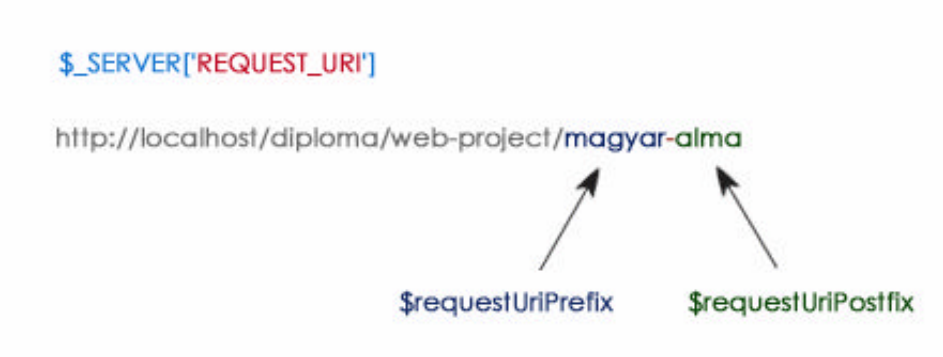
./project/settings/propeties.php

```
header("HTTP/1.1 200 OK");
header("Status: 200 OK", true, 200);
```

A folyamat ott tart, hogy meghívtuk az *index.php*-t. Azonnal feldolgozza az url-t, amit a `$_SERVER[]` globális tömbből olvas ki. Minket a hivatkozás legvége érdekel, amit két részre bontunk *(21. ábra)*.

./project/settings/propeties.php

```php
$requestUri = end(explode('/', substr($_SERVER['REQUEST_URI'], 1)));

$requestUriPrefix = current(explode('-', $requestUri));

$requestUriPostfix = substr($requestUri, strlen($requestUriPrefix) + 1);
```



21.ábra

Az alábbi linkekkel szeretném szemléltetni, hogy mennyivel esztétikusabbak a rövidebb hivatkozások:

http://localhost/diploma/web-project/**japan-harinezumi**
http://localhost/diploma/web-project/**index.php?page=japan&word=harinezumi**

http://localhost/diploma/web-project/**tananyag-24-tesztFile**
http://localhost/diploma/web-project/**index.php?page=tananyag&file=tesztFile&file_id=24**

Legvégül $requestUriPrefix alapján dönti el a rendszer, hogy melyik oldalt szeretnénk megtekinteni.

```
switch ($requestUriPrefix) {

    case 'szotar':
        $currentPage = './pages/search_main.php';
        break;

    case 'tananyag':
        $currentPage = './pages/edu.php';
        break;

    ...

    default:
        $currentPage = './pages/search_main.php';
        break;
}

if (file_exists($currentPage))
    include_once $currentPage;
else
    echo 'Hoppá. Valamit benéztek az oldal készítői.';
```

SEO

Egy ideje komolyan foglalkozni kezdtem a keresőoptimalizálással (*SEO – search engine optimalization*), így az oldal tervezése közben szem előtt tartottam a legfontosabb alapelveket. Erről szeretnék megosztani pár gondolatot.

Rengeteg weboldal van a világhálón. Ha megnézzük a Magyarországi Internet Szolgáltatók Tanácsának oldalát [24], akkor láthatjuk, hogy havonta nagyságrendileg 5000 domaint regisztrálnak (és ugyannyi kerül parkoló listára, vagyis valószínűleg törlik őket). Rengeteg információ kerül fel ezekre a site-okra, amit nehéz rendszerezni, és megtalálni őket.

Egy oldal lehet nagyszerű, de ha nem jut el a célközönséghez, akkor nem sokat ért a ráfordított energia. Több lehetőség van a honlapunk népszerűsítésére, ezáltal a látogatók számának növelésére. Az egyik legfontosabb eszköz talán a keresőkben való előkelő helyezés. Mivel a Google-nek a legnagyobb a részesedése Magyarországon, így főleg a hozzá kapcsolható rangsorolási tényezőt veszem figyelembe.

A *title*, a weboldal címe. Az egyik legfontosabb tényező a keresőoptimalizálás során Megjelenik a böngészők felső sávjában, illetve a keresők találati listáiban. A Google szemében ez a legfontosabb oldalon belüli tényező. Egy releváns, figyelmet felkeltő oldalcím jelentősen megnövelheti az oldal látogatóinak számát, főleg, ha az nem a találati lista élén jelenik meg. Minden egyes oldalnak saját, egyedi címet kell adni (ez elősegíti a jó helyezést a keresőkben és növeli a kattintás valószínűségét), és lehetőleg a fontos szavakat kell a cím elejére tenni. A címsor maximális hosszáról pontos információk nincsenek. Különböző értékekkel lehet találkozni az interneten. Általában 50 és 200 közötti számokat mondanak a források, ha azonban a Google-re koncentrálunk, akkor ez az érték 65 karakter.

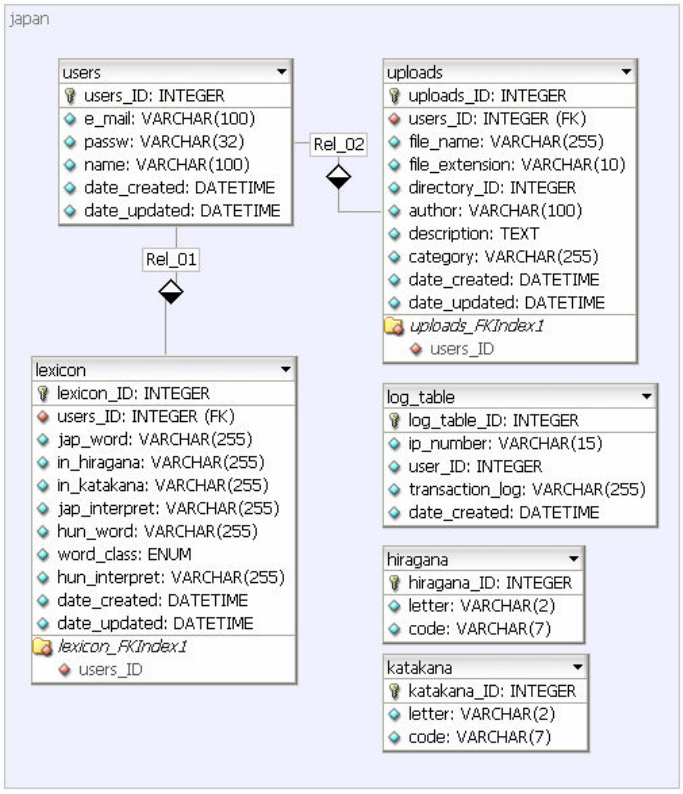
Eredetileg a weblapok leírására és besorolásuk megkönnyítésére találták ki a *description* és *keywords* meta elemeket. A *description*-ben elég egy-két mondatban tömören összefoglalni az oldal tartalmát, a *keywords*-ben pedig maximum 10-12 kulcsszót érdemes szerepeltetni. A látogatók előtt rejtve maradnak, ezért a weblapok létrehozói gyakran esnek túlzásba, teletömve mindenféle oda nem illő dolgokkal ezeket az elemeket. A kereső programok ugyan beindexelik, de valószínűleg túl nagy jelentőséget nem tulajdonítanak a meta elemeknek.

A keresőoptimalizálással kapcsolatos állításokat elég nehéz bizonyítani. A keresők fejlődnek, egyre kifinomultabban szűrik a tartalmat. A találati listán (*SERP - search engine results page*) szereplő sorrend, több tucat tényező együttes hatására alakul ki[25]. Ezeknek a változóknak a súlya napról napra módosulhat, így örökérvényű megállapításokat nehéz lenne kimondani. Egy dolgot mégis érdemes megfogadni:

Ne a robotoknak, hanem a felhasználónak készítsünk weboldalt.

Az adatbázis

Ahogy a bevezetőben is említettem az adatbázis megtervezése, nem az én feladatom volt, így csak pár szóban említeném meg annak felépítését. A lenti képen *(22. ábra)* jól látható a táblák közötti kapcsolat.



22. ábra

users: Felhasználók adatait tárolja.

lexicon: A szótárban található szavakat, és a hozzájuk kapcsolódó információkat tartalmazza.

uploads: Feltöltött fájlok (tananyag) tulajdonságait, és hierarchiáját írja le.

log_table: Adminisztrálja a felhasználók tevékenységét.

hiragana, katakana: Japán írásjelek kódját tárolja.

Adatbázis kapcsolatot létrehozó objektum példányosítása:

```
$database = Database::getInstance();
```

Adatbázis kapcsolat lezárása:

```
if (isset($database))
    $database->close();
```

Sikertelen kapcsolódás esetén hibaüzenetet kapunk *(23. ábra)*

23. ábra

Előfordul, hogy van egy objektumunk, amit nem szeretnénk, ha több példányban fordulna elő. Erre a problémára szolgáltat receptet az egyke (*singleton*) tervezési minta, ami azt írja elő, hogy egy adott osztályból csak egyetlen példány létezhessen. Mivel adatbázis kapcsolatból is elegendő egy darab, így az alábbi módon implementálható a Database osztály:

./project/class/database/class.database.php

```
final class Database
{
    private $resource;
    protected static $instance;

    protected function __construct() {

        $config = new Config();

         $this->resource = @mysql_connect($config->dbUrl, $config->dbUser, $config-
         >dbPassword);

        if (!$this->resource) {

             $errorMessage = new Error('Az adatbázis kapcsolódás sikertelen! <br \>
             mySQL: ' . mysql_error() . '<br \><br \> Kérem látogasson vissza
             később!');

            $errorMessage->showError();

            exit;

        } else
            mysql_select_db($config->dbName);
    }

    public function close() {
        if ($this->resource)
            mysql_close($this->resource);
    }

    public function getResource() {
        return $this->resource;
    }

    protected function __clone() {
    }

    public static function getInstance()  {
        if (self::$instance === null) {
            self::$instance = new self();
        }
        return self::$instance;
    }
}
```

V. Összefoglalás

A szakdolgozatom végén illene összefoglalnom az alkalmazásfejlesztés közben szerzett tapasztalataimat. Célom, egy működő weboldal, és felhasználói felület készítése során alkalmazott eszközök bemutatása volt.

A csapatmunkában való részvétel előnyeit, és hátrányait is sikerült megtapasztalnom. Feladatom elvégzése ugyan nagyobb felelősséggel járt, de cserébe számíthattam mások segítségére. Mivel céges környezetben bevált szokás a kisebb csapatban való fejlesztés, ezért mindenképpen hasznunkra válnak az elmúlt időszakban szerzett ismeretek.

Véleményem szerint, sikeres volt az együttműködés, hiszen megszületett a portál. Nem állítom, hogy tökéleteset alkottunk, de megtettünk minden tőlünk telhetőt. Célkitűzésünk teljesítettük, egy működő alkalmazást csatolhattunk dolgozataink mellé.

Egy olyan keretrendszert hoztunk létre, amelyben egyértelműen elkülöníthetők a különböző feladatokat megvalósító forráskódok. Így könnyebben áttekinthető a fájlok struktúrája, és egyszerűbbé válik a további funkciók implementálása.

Néhány dolgot felsorolnék, amivel bővíthető lehetne az alkalmazás:

- Több jogosultsági szerepkör bevezetése, és az ehhez kapcsolódó szolgáltatások bővítése (például hozzászólások, értékelések, stb. engedélyezése felhasználói oldalról)
- Adminisztrációs felület bővítése
- Statisztikai oldal készítése a látogatottsági adatokból
- Online kitölthető tesztek integrálása
- RSS csatorna
- Hírlevél
- Közösségi oldalakkal való kapcsolat kiépítése (*social bookmarking*)

Közel egy éve kezdtem érdeklődni a keresőoptimalizálás iránt. Arra gondoltam, hogy hasznos lenne ebből a szempontból is támogatni az oldalt. Erről a témakörről szerintem egy külön dolgozat születhetett volna, így csak felületesen tudtam érinteni a dolgot. Sajnos ennek a munkának az eredménye nem látható, hiszen a weboldal nem került a nyilvánosság elé. Ha mégis sikerült volna publikálni, akkor is hosszú hetekbe telt volna, hogy alaposabb következtetéseket vonhassunk le az eredményességéről.

Úgy érzem, hogy sokat tanultam az elmúlt pár hónapban, és ezt a tudást fel tudom használni az elkövetkező feladataim megoldása során.

„A jól elvégzett munka egyetlen jutalma, hogy elvégezhettük.”

Mahatma Gandhi

VI. Irodalomjegyzék

I. Bevezetés

[1] Web 1.0 http://**en.wikipedia.org**/wiki/Web_1.0

[2] Web 2.0 http://**hu.wikipedia.org**/wiki/Web_2.0

[3] What Is Web 2.0 http://**oreilly.com/**web2/archive/what-is-web-20.html

[4] Többrétegű alkalmazás http://en.wikipedia.org/wiki/Multitier_architecture

[5] Orvos Gergő Attila: **Portálkészítés japán szótár és tananyagok számára PHP nyelven**

II. A Japán nyelv

[6] Japán nyelv http://**hu.wikipedia.org**/wiki/Japán_nyelv

[7] Kandzsi http://**hu.wikipedia.org**/wiki/Kandzsi

[8] Kanjik http://**cosjpop.animehq.hu**/nihongo/kanji.htm

III. Fejlesztői eszközrendszer

PHP

[9] PHP dokumentáció http://**www.php.net/**

[10] Dave W. Mercer, Allan Kent, Dan Squier, David Mercer, Wankyu Choi, Steven D. Nowicki: **Bevezetés a PHP5 programozásába** Panem Könyvkiadó (2006)

JavaScript

[11] JavaScript http://**en.wikipedia.org**/wiki/JavaScript

Uniform Server

[12] Uniform Server http://**www.uniformserver.com**/

MySQL

[13] Mysql hivatalos oldala http://**www.mysql.com/**

[14] Mysql http://**en.wikipedia.org**/wiki/MySQL

CSS

[15] Virginia DeBolt: **HTML és CSS Webszerkesztés stílusosan** Kiskapu Kft. (2005)

[16] W3 Consortium http://**www.w3.org/**

http://**www.w3.hu/**

[17] Firebug http://**getfirebug.com**/

Apache

[18] Apache hivatalos oldala http://**www.apache.org**/

[19] Apache http server http://**hu.wikipedia.org**/wiki/Apache_HTTP_Server

xHTML

[20] xHTML http://**www.w3.org**/TR/xhtml1/

Subversion

[21] Freesubversion repository http://**freesubversion.com**/

[22] Tortoise svn http://**tortoisesvn.tigris.org**/

IV. Alkalmazás bemutatása

Tervezés

[23] Browsershots http://**browsershots.org**/

SEO

[24] Magyar ISZT http://**www.domain.hu**/domain/

[25] SeoMoz http://**www.seomoz.org**/article/search-ranking-factors

VII. Függelék

reset.css

```
html, body, div, span, applet, object, iframe,
h1, h2, h3, h4, h5, h6, p, blockquote, pre,
a, abbr, acronym, address, big, cite, code,
del, dfn, em, font, img, ins, kbd, q, s, samp,
small, strike, strong, sub, sup, tt, var,
b, u, i, center,
dl, dt, dd, ol, ul, li,
fieldset, form, label, legend,
table, caption, tbody, tfoot, thead, tr, th, td {
	margin: 0;
	padding: 0;
	border: 0;
	outline: 0;
	font-size: 100%;
	vertical-align: baseline;
	background: transparent;
}
body {
	line-height: 1;
}
ol, ul {
	list-style: none;
}
blockquote, q {
	quotes: none;
}
blockquote:before, blockquote:after,
q:before, q:after {
	content: '';
	content: none;
}

:focus {
	outline: 0;
}

ins {
	text-decoration: none;
}
del {
	text-decoration: line-through;
}


table {
	border-collapse: collapse;
	border-spacing: 0;
}
```

design.css

```
@charset "utf-8";

body{
    font-family:"Segoe UI",Candara,
    "Bitstream Vera Sans","DejaVu Sans",
    "Trebuchet MS",Verdana,sans-serif;
    color:#333;
    text-shadow:1px 1px 1px #fff;
}

a{
    font-weight:bold;
    font-family:"Segoe UI",Tahoma,Geneva,
    sans-serif;
    color:#555;
    text-decoration: none;
    text-shadow:1px 1px 1px #eee;
}

h1{
    color:#5B728F;
    font-size:20px;
    font-weight:bold;
    line-height:120%;
    margin-left: 65px;
}

span.strongShadow{
    font-size:28px;
    font-weight: bold;
    text-shadow:2px 2px 4px #ccc;
    color:#3B526F;
}

div.container {
    width: 920px;
    min-height: 550px;
    margin: auto;
    margin-bottom: 30px;
}

/*  Hibajelzés. */

div.errorBox {
    background:#FFF4A8
    url(./images/error_yellow_background.jpg)
    repeat-x scroll center top;
    border:thin solid #F3D763;
    min-height:20px;
    overflow:hidden;
    padding:14px 10px 10px;
    color:#5B728F;
    font-family:"Segoe UI",Candara,"Bitstream Vera Sans",
    "DejaVu Sans","Trebuchet MS",Verdana,sans-serif;
    font-size:20px;
    font-weight:bold;
    line-height:120%;
    text-align:center;
    text-shadow:1px 1px 1px #fff;
```

```
}


/*  Kereső  */

div.searchInput {
    background:transparent
    url(./images/search_input.png)
    no-repeat scroll 0 0;
    height:45px;
    width:455px;
    margin:auto;
}

input.searchField{
    background:transparent none repeat scroll 0 0;
    border:medium none;
    font-family:Arial,Helvetica,sans-serif;
    font-size:20px;
    height:34px;
    outline-style:none;
    outline-width:medium;
    padding-left:10px;
    padding-right:5px;
    padding-top:6px;
    width:424px;
    color:#333;
    text-shadow:2px 2px 4px #aaa;
}

input.searchButton {
    background:transparent
    url(./images/search_button.png)
    no-repeat scroll 0 0;
    border:0 solid #FFFFFF;
    color:#666666;
    cursor:pointer;
    height:24px;
    margin-left:-30px;
    width:24px;
}


/* Főoldali kereső */

div.mainImage {
    background:transparent
    url(./images/search_image.jpg)
    no-repeat scroll 0 0;
    height:448px;
    width:920px;
    margin:auto;
    padding-top:32px;
}

div.mainBackground {
    background:transparent
    url(./images/search_background.png)
    no-repeat scroll 0 0;
```

```
    height:190px;
    width:800px;
    margin:auto;
    padding-top:42px;
}

/* Menü */

div.menuContainer {
    background:transparent
    url(./images/menu_background.jpg)
    no-repeat scroll 0 0;
    height:52px;
    width:813px;
    margin:auto;
    margin-bottom:50px;
    word-spacing:20px;
}

div.menuContainer a {
    font-weight:bold;
    font-family:"Segoe UI",Tahoma,Geneva,
    sans-serif;
    color:#555;
    text-decoration:none;
    text-shadow:1px 1px 1px #eee;
}

div.menuLeft {
    background:transparent ;
    height:52px;
    width:375px;
    float:left;
    padding-top:7px;
    padding-left:30px;
    text-align:left;
}

div.menuRight {
    background:transparent ;
    height:52px;
    width:370px;
    float:left;
    padding-top:7px;
    padding-right:35px;
    text-align:right;
}


/* Lábléc */

div.footer {
    font-family:arial;
    background:transparent;
    height:52px;
    width:813px;
    margin:auto;
    text-align:center;
    font-size:12px;
```

```
    font-weight:bold;

}

/* Fordítás irányát választó gomb */

div.buttonJpnHun {
    background:transparent
    url(./images/jpn_hun.png) no-repeat scroll 0 0;
    height:86px;
    width:479px;
    margin:auto;
    cursor:pointer;
}

div.buttonHunJpn {
    background:transparent
    url(./images/hun_jpn.png) no-repeat scroll 0 0;
    height:86px;
    width:479px;
    margin:auto;
    cursor:pointer;
}

/* Time */

span.time {
    color:#000000;
    font-family:Verdana,Geneva,Arial,
    Helvetica,sans-serif;
    font-size:11px;
    font-weight:bold;
    padding-left:19px;
    padding-right:5px;
}

div.time {
    background-image:url(./images/time.gif);
    background-repeat:no-repeat;
    display:block;
    font-weight:bold;
    height:24px;
    padding-top:6px;
    float:left;
}

/* Tananyagok */


div.fileContainer{
    margin:  0px 20px 40px 20px;
    float: left;
    width: 140px;
    height: 160px;
    text-align: center;
    font-family: arial;
    font-weight: bold;
    text-shadow:1px 1px 2px #ddd;
    cursor: pointer;
```

```
}

div.filePdf{
    width: 140px;
    height: 140px;
    background-image:url(./images/file_pdf.png);
}

div.fileDoc{
    width: 140px;
    height: 140px;
    background-image:url(./images/file_doc.png);
}

div.fileZip{
    width: 140px;
    height: 140px;
    background-image:url(./images/file_zip.png);
}

div.fileFolder{
    width: 140px;
    height: 140px;
    background-image:url(./images/file_folder.png);
}

div.fileTxt{
    width: 140px;
    height: 140px;
    background-image:url(./images/file_txt.png);
}

div.fileRar{
    width: 140px;
    height: 140px;
    background-image:url(./images/file_rar.png);
}

div.fileOther{
    width: 140px;
    height: 140px;
    background-image:url(./images/file_other.png);
}

div.filePath {
    width: 920px;
    min-height: 40px;
    margin-top: auto;
    text-align: center;
    font-size: 24px;
    line-height: 2em;
}

/* Belépés */

div.login{
    width: 920px;
    height: 540px;
    background-image:url(./images/_test_enter.jpg);
```

```
}

/* arrow */

img.arrow{
    margin: 0px 10px 0px 10px;
}

/* Táblázatok */

table.searchResult{
    width: 100%;
    border:none;
    text-align: center;
}

tr.searchResultHead{
    font-weight: bold;
    height: 30px;
    border-top: 10px;
}

tr{
    height: 25px;
}

td.w8{
    width: 8%;
}

td.w12{
    width: 12%;
}

td.w14{
    width: 14%;
}

td.w20{
    width: 20%;
}

div.infoPage {
    background:transparent
    url(./images/flower.jpg) no-repeat scroll 0 0;
    height:600px;
    width:800px;
    margin:auto;
}

div.adminLoginPage {
    height:600px;
    width:800px;
    margin:auto;
}

/* Form */

form.myForm{
```

```
    width: 880px;
    padding:20px;
    background-color: #eee;
    margin-bottom: 10px;
    overflow: hidden;
}

span.error{
    font-size: 150%;
    color: red;
    text-shadow:2px 2px 2px #ddd;
}

span.accept{
    font-size: 150%;
    color: blue;
    text-shadow:2px 2px 2px #ddd;
}
```

VIII. Köszönetnyilvánítás

Ezúton szeretnék köszönetet mondani témavezetőmnek, Dr. Horváth Gézának a dolgozatom megírásához nyújtott útmutató tanácsaiért.

Hálás vagyok mindazoknak, akik önzetlen segítségükkel hozzájárultak munkám sikeres elvégzéséhez.

Köszönöm Orvos Gergő Attilának, a közös célok megvalósítása érdekében tanúsított kitartó munkáját.