# Adatbázis alapú webes alkalmazás mindössze 18 sornyi kóddal

Webes, adatbázis alapú alkalmazás készítése a nulláról indulva, nyolc könnyed lépéssel.



A *LAMP* összeállítás – *Linux*, *Apache*, *MySQL* és valamilyen programozási nyelv, általában *Perl*, *Python* vagy *PHP* – rendkívül sokoldalú megoldás. Aki készített már webes, adatbázis alapú alkalmazást, az, hogy az ismert fordulattal éljünk, mindet elkészítette. Programozói szemszögből a dolgok hamar ismétlődővé, unalmassá válnak.

Első webes alkalmazásomat a közelmúltban készítettem el. Természetesen linuxos alapokra építettem, *Apache* alatt futott, és *MySQL*-lel folytatott párbeszédet. Ragasztónyelvként, a többi nyelven készült összetevő összefogására szolgáló nyelvként *Perlt* használtam, és az egészet gazdagon megszórtam *CGI*-díszítéssel. Megírtam minden a *MySQL* egyik táblájának bővítéséhez, módosításához és frissítéséhez szükséges kódot, ahogy azt illik. Működött, szép és jó volt. A bajok akkor kezdődtek, amikor ugyanezeket a feladatokat az adatbázis többi táblájához is el kellett végezni. A *Perl* programozók körében régóta bevált szemléletet követve elkezdtem azon gondolkodni, hogyan lehetnék egyszerre lusta és termelékeny. Kellett lennie valami jobb megoldásnak.

Néhány hamvába holt kísérlet és némi keresgélés után rátaláltam a Maypole-ra.

A *Maypole* egy gyors alkalmazásfejlesztési keretrendszer webes alkalmazásokhoz. Eredetileg *Simon Cozens* hozta létre, karbantartását *Sebastian Riedel* vállalta fel. A *Maypole* honlapján azt az ígéretet olvastam, hogy akár 20 sornyi *Perl*-kóddal is teljes értékűen működő alkalmazásokat hozhatok létre. Túl jól hangzott ahhoz, hogy ne próbáljam ki.

Miután megismerkedtem a *Maypole*-lal, állíthatom, *Simon* és *Sebastian* nem hazudtak. Elég néhány sornyi kódot begépelnünk, máris sokoldalú alkalmazáshoz juthatunk. Némi telepítgetésre természetesen szükség van, de hadd védekezzek azzal, hogy az nem programozói munka. Ha a telepítéssel végeztünk, szinte onthatjuk az alkalmazásokat, mindegyik elkészítéséhez csupán egy pár sornyi kódot kell összeállítanunk. A továbbiakban végigvezetem a kedves olvasókat azon a folyamaton, amelynek végén – a *Maypole* közreműködésével – egy kész alkalmazást kapnak a kezükbe.

## 1. lépés: A Linux telepítése – ha szükséges

Az első lépésről külön cikket lehetne írni. Huszáros vágással intézzük el ennyivel: válasszuk ki a nekünk tetsző terjesztést, majd telepítsük.

Mivel nemrég új gépet vettem, letöltöttem a *Fedora Core 3*-at, majd egyedi telepítést választva mindent feltettem a gépre. Aki ezt a luxust nem engedheti meg magának, az az alábbi csomagokat mindenképpen válassza ki: *httpd*, *httpd-devel*, *mod_perl*, *mod_perl-devel*, *mysql* (ügyfél és kiszolgáló) és *Perl*.

## 2. lépés: Az Apache/mod_perl környezet előkészítése

A korszerűbb terjesztésekben már az *Apache 2*-es, valamint a *mod_perl 1.99*-es változata található meg, bár néhányan leragadtak az *Apache 1.3.x* kiadásainál. Szerencsére a *Maypole* az *Apache* bármely változatával képes együttműködni, és – ha a mod_perl nem áll rendelkezésre – *CGI* alapú üzemre is átállítható. Saját *Fedora* telepítésemben az *Apache 2.0.52*-es és a *mod_perl 1.99_16-3*-as változata szerepelt, így a továbbiakban ezek használatát feltételezem.

A *Maypole* levelezési lista tagjai számos különböző *Linux* változatra, köztük *SuSE*, *Debian* és *Red Hat* terjesztésre telepítették már sikeresen a keretrendszert. A *Maypole* az *Apple Mac OS X* operációs rendszere alatt is futtatható, sőt, némi erőfeszítéssel *Microsoft Windows* alatt is életet lehelhetünk belé.

Rootként módosítottam a *Fedora Apache* beállító fájlját (*/etc/httpd/conf/httpd.conf*), és megjegyzésbe tettem a `ServerTokens` utasítást. Gondoskodtam arról, hogy rendszerindításkor az Apache is elinduljon, majd az alábbi parancsokkal üzembe helyeztem a webkiszolgálót:

```
chkconfig httpd on
service httpd start
```

A kiszolgáló állapotát legkönnyebben a szöveges *lynx* böngészővel tudjuk ellenőrizni:

```
lynx -head -dump http://localhost/
```

Az eredmények – lásd a kimenet harmadik sorát – igazolják, hogy az *Apache* és a *mod_perl* üzemkész:

```
HTTP/1.1 403 Forbidden
Date: Wed, 17 Nov 2004 23:30:01 GMT
```

```
Server: Apache/2.0.52 (Fedora)
            mod_perl/1.99_16 Perl/v5.8.5 DAV/2
Accept-Ranges: bytes
Content-Length: 3931
Connection: close
Content-Type: text/html; charset=UTF-8
```

Minden rendben volt, ezért újra megnyitottam a *httpd.conf* fájlt, és kivettem megjegyzésből a `ServerTokens` utasítást, ugyanis nem túl szerencsés, ha webkiszolgálónk lapjaiba be engedjük tekinteni az esetleges támadókat. Ha már úgyis meg volt nyitva a fájl, a `ServerAdmin` utasításban megadtam a saját e-mail címemet, majd a `ServerName` utasításban a kiszolgálóm *DNS*-nevét. Egyúttal feljegyeztem a `DocumentRoot` utasításban megadott könyvtár elérési útját; esetemben ez a */var/www/html* volt.

## 3. lépés: A MySQL előkészítése

A kiválasztott terjesztéstől függően lehetséges, hogy a *MySQL* már telepítve van a gépünkre. Ha nem ez a helyzet, akkor a *MySQL*-t a terjesztésünkhöz tartozó letöltések közül vagy a *MySQL* weboldaláról szerezhetjük be. Saját, *Fedora* alapú gépemen – rootként bejelentkezve – a szokásos parancsokkal készítettem fel használatra a *MySQL*-t:

```
chkconfig mysqld on
service mysqld start
```

Miután elindult a *MySQL*, megadtam a hozzá tartozó felügyeleti jelszót:

```
mysqladmin -u root password 'jelszo'
```

## 4. lépés: A Maypole telepítése

A *Maypole* és az *Apache* között a *mod_perl* közreműködésével közvetlen párbeszéd folyik. Ha az *Apache 2*-es változatával szeretnénk dolgozni, akkor a *libapreq2* nevű fejlesztői könyvtárat is le kell töltenünk a *CPAN* gyűjteményből, majd telepítenünk kell a *Perl* csomagok közé. Jómagam a *libapreq2-2.04_03-dev.tar.gz* fájlt töltöttem le a *CPAN*-ról. A könyvtár telepítése előtt frissítettem a *Perlhez* alapesetben is hozzá tartozó `ExtUtils::XSBuilder` modult. Rootként elegendő az alábbi parancsot kiadni:

```
perl -MCPAN -e "install ExtUtils::XSBuilder"
```

Ha ez a *CPAN* héj első futtatása, akkor lehetőséget kapunk a helyi *CPAN* modul beállításainak megadására. Amikor rákérdez azoknak a moduloknak a letöltésére, melyek megléte a működés előfeltétele, akkor igyekezzünk odafigyelni. Miután frissítettem a modult, telepítettem a *libapreq2* könyvtárat; ennek során a szokásos, a Perl modulok telepítésére használt parancsokat adtam ki:

```
tar zxvf libapreq2-2.04_03-dev.tar.gz
cd libapreq2-2.04-dev/
perl Makefile.PL
make
make test
su
make install
<Ctrl-D>
```

A *Maypole* tényleges telepítése a *CPAN* héj rootként való meghívásával kezdődik:

```
perl -MCPAN -e "shell"
```

Mivel a *Maypole* működéséhez számos *CPAN* modul jelenlétére van szükség, a telepítés eltart egy ideig. Mielőtt a *CPAN* héjat a *Maypole* tényleges telepítésére utasítanánk, az alábbi parancsok kiadásával ellenőrizhetjük, hogy az időnként gondot okozó modulokkal is minden rendben van-e:

```
cpan> install CGI::Untaint::date
cpan> force install Class::DBI::mysql
```

Nálam a `Class::DBI::mysql` telepítését úgy kellett ráerőltetni a gépre, miután több ellenőrzés is sikertelen volt, illetve az önműködő telepítés is megszakadt. A telepítés erőltetésekor a sikertelen ellenőrzéseket a program figyelmen kívül hagyja, így végig tud futni a folyamat. Ha az előzetes feltételek teljesülnek, a *Maypole* telepítését a következő *CPAN* paranccsal kezdhetjük meg:

```
cpan> install Maypole
```

A parancs kiadása után a program különféle önműködő párbeszédeket folytat le a *CPAN* gyűjteménnyel. Figyeljünk, hogy mi történik, ugyanis bizonyos pontokon egyszerű kérdésekre kell válaszolnunk. Amikor végeztünk, pihegjük ki magunkat – szép munkát végeztünk, a *Maypole* legújabb (írásom születésekor 2.04-es) változata ott vár bennünket a gépünkön.

## 5. lépés: Az adatbázis és néhány tábla létrehozása

Visszatérve a *MySQL*-hez, bejelentkeztem mint rendszergazda, és az alábbi parancsok kiadásával az összes alapértelmezett fiókot töröltem:

```
mysql -u root -p
mysql> use mysql;
mysql> delete from user where User = '';
mysql> flush privileges;
```

Ezután létrehoztam egy új adatbázist, valamint egy felhasználót is, akit az adatok tulajdonosának szántam:

```
mysql> create database CLUB;
mysql> grant all on CLUB.* to manager identified
          by 'jelszo';
mysql> quit
```

A fenti parancsok hatására létrejön egy *CLUB* nevű adatbázis, valamint bekerül a rendszerbe egy *manager* nevű felhasználó. A példa kedvéért tegyük fel, hogy alkalmazásunk célja egy gyermek futballklub tagjainak nyilvántartása. A rendszer tárolja a játékosok személyes adatait, azt, hogy

melyik csapatba tartoznak, valamint egészségi állapotukat. Nézzünk néhány *SQL*-fájlt, ezek segítségével adtam meg a *CLUB* adatbázison belüli táblákat. Az első fájl (*create_player.sql*, játékos létrehozása) a *player (játékos)* táblát hozza létre:

```
create table player
(
    id                  int not null
                        ➥auto_increment
                            primary key,
    name                varchar (64) not null,
    date_of_birth       date,
    address             varchar (255),
    contact_tel_no      varchar (64),
    squad               int,
    medical_condition   int
);
```

A második fájl (*create_squad.sql*, csapat létrehozás) a *csapatok (squad)* kezdő listáját hozza létre:

```
create table squad
(
    id    int not null auto_increment
          ➥primary key,
    name  varchar (32) not null
);
insert into squad (name) values ('--');
insert into squad (name) values ('Under 8');
insert into squad (name) values ('Under 9');
insert into squad (name) values ('Under 10');
insert into squad (name) values ('Under 11');
insert into squad (name) values ('Under 12');
```

A csapat táblát megpróbáltam értelmes kezdő adatokkal feltölteni (például *„under 8” - „8 év alattiak”* stb). A harmadik, s egyben utolsó fájl, a *create_condition.sql* a lehetséges egészségi állapotok listáját hozza létre:

```
create table condition
(
    id    int not null auto_increment
          ➥primary key,
    name  varchar (64) not null
);
insert into condition (name) values ('--');
insert into condition (name) values ('Asthma');
insert into condition (name) values ('Epilepsy');
```

A `squad` táblához hasonlóan a `condition` táblába is beilleszt-tettem néhány alapadatot. A `squad` és a `condition` tábla adattételének neve `name` *(név)*. Ennek jelentőségét később még látni fogjuk.
Az *SQL* fájlokkal tudjuk létrehozni az adatbázison belül a táblákat:

```
mysql -u manager -p CLUB < create_player.sql
mysql -u manager -p CLUB < create_squad.sql
mysql -u manager -p CLUB < create_condition.sql
```

Bizonyára mindenki kitalálta már, hogy a *CLUB* adatbázis a játékosok adatait fogja tárolni. A játékosok egy csapathoz tartoznak, és van valamilyen egészségi állapotuk.

## 6. lépés: Az alkalmazás beállítása

Készen áll az adatbázis, a *Maypole* is telepítve, eljött tehát az alkalmazás beállításának ideje. Az *Apache httpd.conf* beállító fájljához egy új utasítást kell hozzáadni, amelynek alapján a *mod_perlre* bízzuk a *Maypole* alkalmazás kezelését. Én a következőket írtam a fájl végére:

```
<Location /Club>
    SetHandler perl-script
    PerlHandler ClubDB
</Location>
```

A fenti sorokból tudja meg az *Apache*, hogy amikor a */Club URL* iránti kérést kap, akkor azt a *ClubDB Perl* parancsfájlnak – ezt a következő lépésben írjuk meg – kell átadnia feldolgozásra. A következő parancsokat rootként kiadva adtam meg az *URL* helyét:

```
mkdir /var/www/html/Club
cd /var/www/html/Club
cp -r ~/.cpan/build/Maypole-2.04/templates/* .
cp maypole.css ../club.css
```

Először létrehoztam egy könyvtárat, az *Apache* gyökérkönyvtárán belül ez tartalmazza az alkalmazásom *URL*-jét, majd belemásoltam a *Maypole*-hoz mellékelt alapértelmezett sablonokat. A *Maypole CSS* fájlját is bemásoltam a webkiszolgáló *DocumentRoot* könyvtárába, majd az alkalmazásomnak megfelelően átneveztem. Az utolsó telepítési jellegű lépés egy beállító fájl létrehozása az *Apache /etc/httpd/conf* könyvtárán belül, melynek feladata az alkalmazás *MySQL* felhasználónevének és jelszavának tárolása. Ebbe a *ClubDB.conf* nevű fájlba a következő sorok kerültek:

```
[client]
user=manager
password=jelszo
```

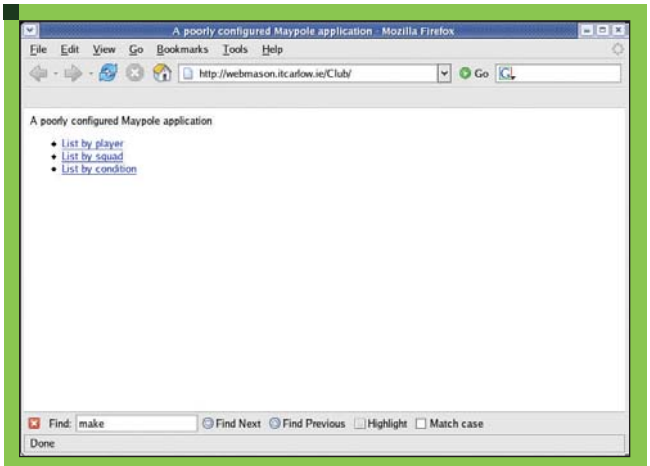
## 7. lépés: A 18 sornyi kód megírása

A labdarúgó klub adatbázis-kezelő kódja a *ClubDB.pm* fájlban található. Minden *Maypole* alkalmazás egy *Perl* névteret megadó `package` utasítással kezdődik. Bekapcsoljuk a *„szigorúságot” (strictness)*, majd használatba vesszük az `Apache::MVC` nevű *Maypole* alapmodult:

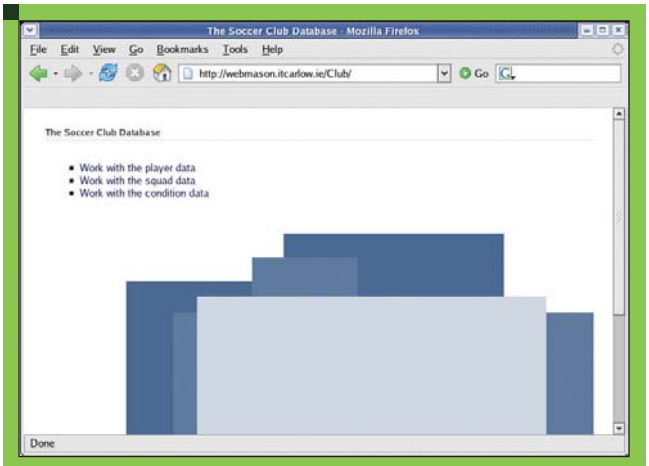
```
package ClubDB;
use strict;
use base 'Apache::MVC';
```

A kód ezután a megadott beállító fájlban szereplő felhasználónév és jelszó használatával kapcsolatot létesít az adatbázissal:

```
ClubDB->setup( "dbi:mysql:CLUB;
    mysql_read_default_file=
        /etc/httpd/conf/ClubDB.conf" );
```




*1. ábra* A Maypole alapértelmezett nyitóképernyője




*2. ábra* A labdarúgó klub testreszabott nyitóképernyője



A következő sorok tájékoztatják a ***Maypole***-t az alkalmazás kezdő webcíméről, továbbá megadják, hogy az adatbázis mely tábláinak elérésére van szükség. Mivel egyszerű programról van szó, a legjobb, ha minden tábla elérhetőségét biztosítjuk:

```
ClubDB->config->{uri_base} =
    "http://webmason.itcarlow.ie/Club/";
ClubDB->config->{display_tables} =
    [ qw[ player squad condition ] ];
```

A csapatokra térve: az alkalmazás lehetővé teszi a csapatnevek megtekintését, módosítását és törlését. Ennek megadása további pár sort tesz ki, ezek egyike újabb névteret ad meg:

```
package ClubDB::Squad;
sub display_columns{ "name" };
ClubDB::Player->untaint_columns(
    printable => [ "name" ] );
```

Az `untaint_columns` eljárás az oszlopban várt adat típusát adja meg, és itt jelezzük azt is a ***Maypole***-nak, hogy az oszlop a webes felületen keresztül módosítható. Az egészségi adatok kezelése hasonló módon történik:

```
package ClubDB::Condition;
sub display_columns{ "name" };
ClubDB::Condition->untaint_columns(
    printable => [ "name" ] );
```

A `player` táblához tartozó kód egy picit bonyolultabb, de csak picit. Újabb névteret adunk meg, majd a has_a eljárást kétszer meghívva létrehozzuk a `player` és a többi tábla közötti kapcsolatokat. A kapcsolatok kifejezése a korábban megadott névterekkel történik:

```
package ClubDB::Player;
ClubDB::Player->has_a(
    squad => "ClubDB::Squad" );
ClubDB::Player->has_a(
    medical_condition => "ClubDB::Condition" );
```

A `players` esetében a megjelenítendő oszlopokat a `display_columns` eljárás segítségével soroljuk fel. Ezzel a módszerrel a programozó szabályozhatja az oszlopok webes felületen való megjelenésének sorrendjét. Ha a `display_columnst` elhagyjuk, a ***Maypole*** ABC sorrendben jeleníti meg az oszlopokat, ami nem feltétlenül felel meg az igényeinknek. Az `untaint_columns` meghívásával határozzuk meg, hogy az oszlopokon belül milyen típusú adatokat szabad módosítani. A kód a ***Perlben*** jól ismert „`1;`” lezárással végződik, mely minden Perl modul elmaradhatatlan kelléke:

```
sub display_columns{ qw( name address
    date_of_birth contact_tel_no
        squad medical_condition ) };
ClubDB::Player->untaint_columns(
    integer    =>
        [ "squad", "medical_condition" ],
    printable =>
        [ "name", "address", "contact_tel_no" ],
    date       =>
        [ "date_of_birth" ] );
1;
```

Számoljuk meg a pontosvesszőket! Ha figyelembe vesszük, hogy a fenti kódrészleteket a nyomdai kötöttségeknek megfelelően kellett tördelni, akkor láthatjuk, hogy bizony összesen 18 sornyi kódot írtunk. Végső lépésünk a ***Perl*** modul olyan helyre másolása, ahol az ***Apache*** és a ***mod_perl*** is meg tudja találni:

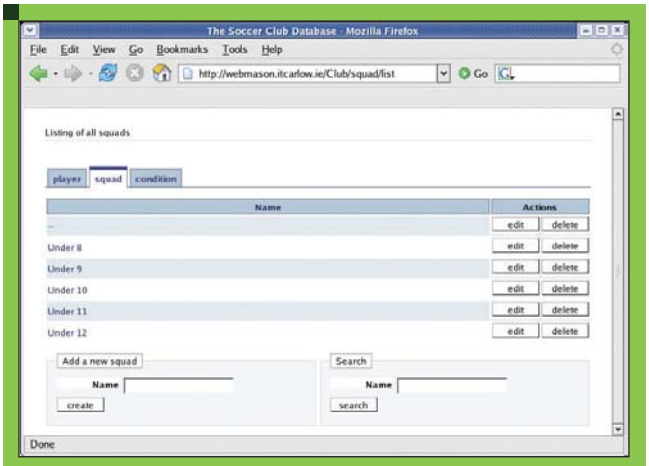
```
mkdir -p /etc/httpd/lib/perl/
cp ClubDB.pm /etc/httpd/lib/perl/
```

## 8. lépés: Rajta!

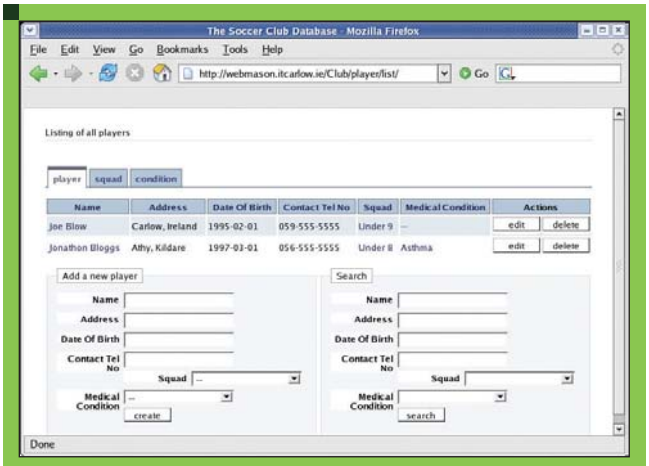
A ***Maypole*** alkalmazás használatba vétele előtt indítsuk újra az ***Apache***-ot:

```
service httpd restart
```

Beírtam a ***http://webmason.itcarlow.ie/Club/*** címet a ***Firefox*** címsorába, aminek hatására az *1. ábrán* láthatók jelentek meg. Jobb, mint a semmi, de nem tökéletesen az, amire számítottam.




*3. ábra* A Maypole felülete a squad táblához




*4. ábra* A Maypole felülete a player táblához

Ha valakinek nem volna nyilvánvaló: csinos *CSS* kimenetet vártam, nem nyers *HTML*-t. A hiba javításához megkerestem a 6. lépésben a webkiszolgáló alá másolt alapértelmezett sablonfájlokat. Ezek módosításával úgy tudjuk megváltoztatni alkalmazásunk megjelenését, hogy annak forráskódjába nem kell belenyúlnunk. Remélem, mindenki felfigyelt az utolsó mondatomra! A lényeg, hogy az alkalmazás megjelenését a *CSS* sablonok határozzák meg, működésének módját a kód adja, míg az adatokat a *MySQL* szolgáltatja. A feladatok elkülönítésével rendkívül termelékeny környezetet lehet teremteni, ugyanis az alkalmazás adott részének megváltoztatása semmilyen hatással nem lehet a többire.

A sablonok a *factory* nevű alkönyvtárban találhatók, ezt az alkalmazás *URL*-je – ez esetben *Club*/ – alatt találjuk. A *factory (gyári)* sablonok a *Maypole* alapértelmezett sablonjai, és egészen addig használatban maradnak, míg egy másik, *custom (egyedi)* nevű könyvtárba helyezett sablonokkal felül nem bíráljuk őket.

Miután létrehoztam a *custom* könyvtárat a *Club*/ *URL* alatt, a *factory* könyvtárból átmásoltam a *customba* a *fejrész (header)* fájlt, majd a vi segítségével átírtam. A */maypole.css* nevét */club.css*-re változtattam, valamint az A *„poorly configured" (rosszul beállított)* üzenet helyett is valami méltóbbat adtam meg. A *címlap (frontpage)* fájlt szintén átmásoltam a *factoryból* a *customba*, és megfelelőbb alkalmazásleírást adtam meg benne. Ezután módosítottam a *custom/frontpage* fájl hivatkozás címkéjét, az alapértelmezett *„List by player" (Listázás játékos szerint)* szöveget *„Work with the player data" (A játékosok adatainak kezelése)* szövegre változtattam. Miután végeztem a módosításokkal, rákattintottam a *Firefox* frissítés gombjára. A *2. ábrán* látható eredmény – azt hiszem, egyetértünk – jóval szebb volt.

A menüpontok bármelyikére kattintva csinos, a *3.* és a *4. ábrán* láthatóhoz hasonló beviteli képernyőt kapunk.

A 4. ábrán a két kitalált játékos adatainak bevitele utáni tartalom látható. Vegyük észre az „ingyen" kapott lehetőségeket. A táblákhoz tartozó fülek a képernyő tetején láthatók. A táblák adatait egyszerűen a megfelelő fülekre kattintva érhetjük el. Minden sorhoz tartozik egy szerkesztést és egy törlést lehetővé tévő gomb. Bármelyik oszlop fejlécére kattintva az adott oszlop adatai szerint rendezhetjük a listát, keresni vagy játékosokat hozzáadni pedig az e célra szolgáló űrlapokkal tudunk. A játékosok csapatát vagy egészségi állapotát a megfelelő mezőre kattintva, legördülő listából választhatjuk ki, ez az apró varázslat annak köszönhető, hogy korábban a has_a utasítással előírtuk, hogy minden játékosnak van csapata és egészségi állapota. Alapesetben a *Maypole* a hivatkozott tábla *name (név)* adatoszlopában szereplő adatok alapján állítja elő a legördülő listák tartalmát.

Mit is kaptunk? Teljes értékű webes felületet a háttérben futó adatbázishoz – mindössze nyolc egyszerű lépéssel. A *Maypole* még újdonságszámba megy, ám máris aktív közösség épült köré. A levelezési lista nemrég oszlott kétfelé, az egyik a fejlesztők, a másik a felhasználók eszmecseréjét szolgálja; weblapja pedig a *perl.org*-ra került.

Remélem, sikeresen szemléltettem: a *Maypole* használata – ha túlestünk a telepítésen – gyerekjáték. A webes alkalmazásokhoz szükséges elemek túlnyomó részét készen kapjuk kézhez, ezeket szükség szerint tudjuk bővíteni. A *Maypole* nemcsak *MySQL*-lel, hanem bármely más *SQL* alapú adatbázis-kezelő rendszerrel is használható. További tájékoztatást a *Maypole* honlapján szereplő leírásokban lehet találni.

*Linux Journal 2005. március, 131. szám*

***Paul Barry*** (paul.barry@itcarlow.ie)
Az írországi Carlow Műszaki Intézetében oktat. Az általa tartott tárgyakról, könyveiről és egyéb írásairól a glasnost.itcarlow.ie/~barryp weboldalon lehet bővebb információkat találni.

## KAPCSOLÓDÓ CÍMEK

- maypole.perl.org
- search.cpan.org
- www.mysql.com