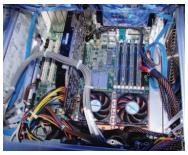
Hírek

8 dudás egy csárdában



A *las vegas-i CES 2007 (Consumer Electronics Show)* kiállításon az *Intel* bemutatta a nyolc magot számláló munkaállomását. A nyolc mag két darab négymagos *Xeon* processzor alkalmazásával állt elő. A *GPW-771E* jelű munkaállomás a bejelentés időpontjában 4083 és 9493 dollár között áron érhető el. Az *Intel* jelenleg még nem lát túlzott keresletet a bemutatott gépre. (Kétprocesszoros megoldást először az *AMD* demonstrálta 2 darab kétmagos processzorral.)

➲ http://www.pcmag.com/article2/0,1895,2080511,00.asp

Microsoft-Novell és a Wal-Mart

A *Wal-Mart Suse Linux* előfizetést vásárol a *Microsoft*tól informatikai infrastruktúrájához. A Microsoft ugyanis az öt évre szóló *Microsoft-Novell* egyezmény keretében olyan utalványokat értékesít, illetve szállít, amelyekkel az ügyfelek frissítési előfizetést és műszaki támogatást vehetnek igénybe a *Novell*től a *SUSE Linux Enterprise Server* termékhez.

➲ http://linux.slashdot.org/article.pl?sid=07/01/23/224254

Nokia N800

Már kapható 400 dolláros áron a *Nokia 770* fejlesztett változata az *N800*. A *770*-hez képest az alábbi változások történtek: *VGA* felbontású webkamera, két normál méretű *SD kártyahely* – üröm az örömben, hogy maximum 2 gigabájtos kártyák használhatóak –, sztereó hangszoró.

A *RAM* méret 64 megabájtról 128-ra emelkedett, míg a flash 64-ről 256-ra. (Ebből 80 megabájtot foglal a *Linux*.) A processzor 220 MHz helyett 320 MHz-en ketyeg. A kijelző változatlan maradt (800x480 képpont).

➲ http://www.linuxdevices.com/news/NS9981902594.html

Linux kernel dióhéjban

Greg Kroah-Hartman ingyenesen elérhetővé tette angol nyelven a korábban *O'Reilly* kiadásban megjelent *Linux kernel dióhéjban* című könyvét. A könyv *PDF* és *DocBook* formátumban is letölthető fejezetenként, de akár egy tar fájlként is.

➲ http://www.kroah.com/lkn/

Linuxos médialejátszó

Az *AOL* bemutatta *Linux*os médialejátszóját, mely a kínai *Haier* cég gyártósorán készült. A *Smartscreen* 30 gigabájtos merevlemezzel érkezik, lejátssza az *MPEG4* és *WMV* formátumú videókkal. Vezetékes fejhallgató mellett használhatunk *Bluetooth*osat is. Az már csak hab a tortán, hogy a beépített *wifi* segítségével *internet*es rádiókat is hallgathatunk. A boltok polcaira előreláthatóan az év közepén kerül.

➲ http://www.linuxdevices.com/news/NS4183422998.html

Linux + Windows = MotoRizr Z6

A legújabb *Motorola* telefon elég érdekes kombinációként mutatkozott be. Noha *Linuxot* futtat, mégis teljes *Windows Media DRM (Digital Rights Management)* lett implementálva benne. A híradás csupán a *Windows*os együttműködésről ír. Hogy linuxos létére képes-e asztali *Linuxszal* kommunikálni, arról egyelőre nincs információnk.

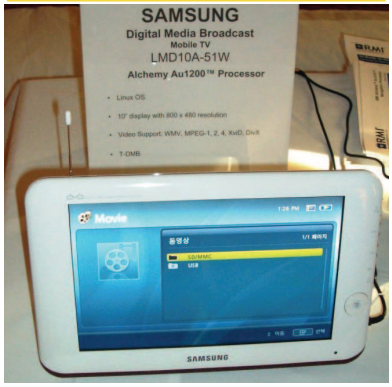
➲ http://www.linuxdevices.com/news/NS8033931935.html

Navigare necesse est

A víz alatt sem árt a pontosság. Ennek a jegyében készül a linuxos ***NodeExplorer v3***. A ***GPS***-es ***PDA*** búvárkodás közben segítheti a navigációt, hiszen 35 méteres mélységig használható. A ***v3***-as modell 400 MHz-es processzora és a 320x240 felbontású kijelzővel akár kedvenc filmjeinket is megnézhetjük a víz alatt – ha bírjuk levegővel – hiszen a készülék 8 órás folyamatos videólejátszást ígér mindössze háromórás töltés után. Az eszköz maximum 4 gigabájtos kártyával bővíthető, a külvilággal pedig ***wifi***n vagy ***USB***-n tud kommunikálni.

➲ http://www.linuxdevices.com/news/NS7352337851.html

Samsung hordozható digitális TV-je



A ***Samsung*** piacra dobta ***MIPS*** alapú ***Linux***os ***TV***-jét (***LMD10A-51W***), mely 10 hüvelykes, 800x480 képpont felbontású kijelzővel kapható. Az eszköz a ***DMB-TV*** szabvány mellett támogatja az 1.2 megabites sávszélességet adó ***T-DMB*** – 352x288 képpont felbontású - szabványt is. Emellett memóriakártyáról ***DivX***, ***XviD***, ***WMV*** és ***MPEG*** (1,2 vagy 4) tömörítésű videókat játszik le, de nem okoznak gondot az ismertebb audió formátumok sem. Az készülék lelke a ***Raze Microelectronics*** gyártósoráról kikerülő ***Alchemy Au1200***-as egychipes processzor.

➲ http://www.linuxdevices.com/articles/AT4612631999.html

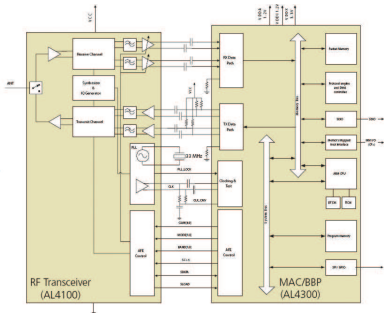
Arc, ujjlenyomat? Rendben...

A litván ***Neurotechnologija*** bemutatta ***FaceCell*** és ***FingerCell*** fejlesztői könyvtárának 2.0-ás verzióját, amely ***Windows CE***, ***Windows Mobile*** mellett ***Linux*** alatt is megy. Az arcfelismerő modul akár 3 ezer arcot is képes felismerni másodpercenként, míg az ujjlenyomat-felismerő másodpercenként 700 ujjlenyomatot azonosít. Nem jelent problémát, ha nem a megadott szögben helyezi rá a felhasználó az ujját az olvasóra vagy csak részleges ujjlenyomatot kap a rendszer.

A két rendszer külön-külön is megállja a helyét, de együtt alkalmazva segít a munkaképtelen vagy olvashatatlan ujjlenyomatú emberek azonosításában is. A fejlesztői könyvtár forrásként vagy binárisként is megvásárolható a cégtől vagy partnereitől.

➲ http://www.linuxdevices.com/news/NS7395311579.html

Wireless USB Linux alatt



Az ***Alereon*** már szállítja ***vezeték nélküli USB*** chipjét (***AL4300***), melyet mérete alapján mobileszközökbe és ***PC*** kiegészítőkbe is beépíthetnek a gyártók (10x10 milliméter) Az ***AL4300*** rendelkezik ***Linux***os meghajtóprogrammal és az ***AL4100***-al kiegészítve teljes értékű ***wireless USB*** környezetet ad, mely megfelel a ***wireless USB 1.0***-ás és a ***WiMedia 1.0***-ás szabványoknak. A chip darabonkénti ára tízezres rendelés esetén 13 dollár.

➲ http://www.linuxdevices.com/news/NS4745171942.html

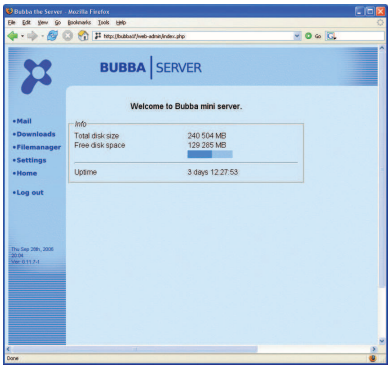
Fedora, de nem Core

A ***RedHat*** támogatással bíró ***Fedora*** nevet váltott. Nem lesz többé ***Core*** és ***Extras***. A 7-es kiadás már csak egyszerűen ***Fedora 7***-ként jelenik meg.

➲ http://www.linuxdevices.com/news/NS2138609241.html

Halk személyes szerver Debian alapokon

A svéd ***Excito Bubba*** szervere mindössze 10 wattos fogyasztásával és 27 dB-es zajszintjével nagyon valószínű, hogy kedvenc nonstop gép lesz.



A 200 MHz-es ***ARM*** processzor és a 80-500 gigabájtos merevlemez teljes értékűvé teszi az eszközt, hiszen gyárilag ***torrent***, ***web*** és ***ftp letöltési*** lehetőséggel bír, továbbá ***web***, ***fájl***, ***email*** és ***nyomtató szervert*** – ***USB-s nyomtatókat*** kezel – is kapunk vele. Az eszköz webes felületen keresztül konfigurálható. Két színben kapható (fehér és kék), az ára pedig a merevlemez méretétől függően 264 és 432 euró közötti.

➲ http://www.linuxdevices.com/news/NS4105652894.html

Merevlemezes rögzítő

Moxi néven tervezi piacra dobni ***Diego*** két legfrissebb merevlemezes felvevőjét, melyek közül legalább az egyik ***Linux***os lesz. A céget sokan onnan ismerhetik, hogy itt dolgozik ***Andrew Morton***, ***Linus Torvalds*** jobbkeze.

➲ http://www.linuxdevices.com/news/NS8029845408.html

Amerikai focit közvetít a Linux

Az ***FSMLabs Cablecam*** termékével lehetővé vált, hogy stadionok sportesemányeit olyan pozícióból is rögzítse kamera, ahonnan eddig nem volt rá mód. A stadion négy sarkából indulva (biztonsági okok miatt) két-két drótkötél van a kamerához rögzítve, amely így szinte a teljes játéktér fölött és állítható magasságban (4-15 méter) képes mozogni. A rendszer üzemeltetéséhez az ***RTLinux***-ot választotta a cég, hiszen a valósidejű üzem létfontosságú a 4 darab 20 lóerős motor precíz vezérléséhez. Az eszközt két személy kezeli. Az egyik a kamerát pozicionálja, míg a másik a kamerát kezeli (zoom, forgatás, stb.)

➲ http://www.linuxdevices.com/news/NS2805234558.html

FreeNAS

A ***FreeNAS*** egy teljes értékű ***FreeBSD***, mely lehetővé teszi, hogy tetszőleges gépből fájlszervert csináljunk. A letöltendő CD képfájl csupán 38 megabájt, ami szinte semmiség a mai ***Linux*** disztribúciókhoz képest. Telepítés után a rendszert webes felületen is konfigurálhatjuk. A projektet januárban a hónap projektjének választották a ***SourceForge.net***-en.

➲ http://www.freenas.org/

➲ http://www.linux.com/article.pl?sid=07/01/19/1514204

Linuxos miniPC

A ***BosaNova*** is bemutatta legújabb kis fogyasztású miniszámítógépét (***RBT-802***), melyet elsősorban hálózati munkaállomásnak ajánlanak. 500 MHz-es LX800-as processzor, 128 megabájt memória és 64-128 megabájt flash memória található benne. A klasszikus PS/2-es csatolókat hiába keressük rajta, azonban 4 ***USB 2.0***-ás port, hang, 10/100-as hálózati csatoló ***PXE*** támogatással és ***Wifi*** áll a rendelkezésünkre. A méretéhez és tudásához képest határozottan drága: 450 dollár az ajánlott ára.

➲ http://www.linuxdevices.com/news/NS7190647374.html

Dupla kijelzős vékony kliens Linuxszal

Már rendelhető a ***Wyse Technology***-tól ***Linux***os vékonykliensük, a ***V50***, mely az eddigi ilyen jellegű termékek közül azzal tűnik ki, hogy egy speciális elosztó kábellel akár két monitorral is használható, ekkor azonban meg kell elégednünk 1280x1024 képpontos felbontással 60 Hz mellett. Az eszköz magja egy ***Via C7***-es processzor, mely 256 megabájt memóriával gazdálkodhat. Az operációs rendszert 128 megabájtos flash memóriából tölti. (Egyéni igény szerint a memóriából maximum 1 gigabájtot, flashből pedig maximum 2 gigabájtot kérhetünk.) A párhuzamos és a soros port mellett 3 darab ***USB 2.0***-es aljzatot is találunk. ***Wireless*** csatoló csak kérhető opcióként van jelen. További érdekesség: a hangkimenet 16 bites, a bemenet viszont csak 8 bites. A ***V50***-ért 399 dollárt kérnek és ebben még nincs benne a speciális videójelosztó kábel.

➲ http://www.linuxdevices.com/news/NS3534571400.html

Linux Windows alatt

A *StarNet* terméke ott lehet érdekes, ahol *Windows* van a munkaállomásokon, de szükség lenne grafikus *X-es* alkalmazások futtatására is.
Az *X-Win32 Flash* bedugásakor egy *X Window* kliens indul el. A program nem telepít semmit, így azonnal használhatjuk. A cég szerint eddig nem volt ilyen, vagy hasonló megoldás. (Korábban a *Cygwin* hasonló, de elég sok helyet foglalt és előre kellett telepíteni.) A program oktatási (23 dollár) és kereskedelmi (225 dollár) licensszel jelenik meg. Természetesen kérhetünk egy hónapos, ingyenes kipróbálásra jogosító licenszt is. Ha még nincs saját pendrive-unk, akkor azt is kérhetünk 1-4 gigabájtos méretben. Ebben az esetben 300 dollárról indul az ára.

➲ http://www.linuxdevices.com/news/NS7845417909.html

Ablakkezelés másképp

A *Metisse* újraértelmezi az ablakkezelést. Számos olyan újítást behozott, amelyet eddig más ablakkezelőben nem tapasztalhatott a felhasználó. Ilyen például, ha egy háttérben lévő ablakban jelölünk ki. Ilyenkor a rajta elterülő ablakok diszkréten félre hajlanak. A *Mandriva*ba előreláthatóan tavasszal kerül be a *Metisse*.

➲ http://www.mandriva.com/projects/metisse

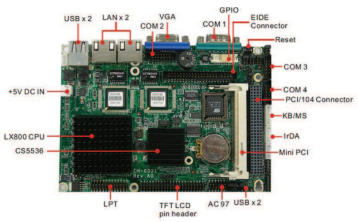
Qt/Qtopia programozói verseny

A *Qt* programozói versenyt hirdet, melynek célja: írjunk *Qt/Qtopia*-ra épülő programokat, függvénykönyvtárakat. (A *Qtopia Core* jelenleg az alábbi architektúrákra érhető el: *ARM*, *x86*, *MIPS*, *PowerPC*) Az alábbi kategóriákban indulhatnak a vállalkozó szellemű-ek: mobil alkalmazások, asztali alkalmazások, függvénykönyvtárak, *widgetek* és fejlesztői eszközök. A kategóriánkénti győzteseket az alábbi szempontok alapján választják: ötlet, mennyire befejezett, portolhatóság, kinézet, dokumentáció, a forráskód minősége, tesztek, függőségek, csapat mérete. A nyertesekre értékes díjak várnak. Pályamunkákat *2007 május 31-ig* lehet beadni a *QtCentre.org* oldalán.

➲ http://contest.qtcentre.org/

➲ http://www.linuxdevices.com/news/NS9820024664.html

Energiatakarékos apróság



Az *Aewin*től már rendelhető az *EM-6331*, mely egy 145x102 milliméteres miniszámítógép *PCI/104* és *mini PCI* csatlakozókkal. A rendszer lelke egy 500 MHz-es 0.9 watt fogyasztással bíró *Geode LX800*-as processzor. Az alaplap egy darab, maximum 1 gigabájtos *DDR* memóriát fogad. Háttértárnak *IDE* eszközt használhatunk vagy *CompactFlash* memóriakártyát. Az eszköz csatlakozását 2 darab 10/100-as hálózati, 4 darab *USB 2.0*-es, 4 darab soros – ebből egy lehet *RS422/485* is –, 1 darab *IrDA* és 1 darab párhuzamos port adja. Fogyasztása csupán 12 watt. Az ára egyelőre sajnos ismeretlen.

➲ http://www.linuxdevices.com/news/NS4556806972.html

Medve Zoltán
(e-medve@e-medve.hu)
2001-ben kezdett „Linuxolni”, de már korábban is ismerkedett a szabad szoftverek világával. Ha éppen nem a gép előtt ül, akkor fotózgat, olvasgat vagy bicajozik.

Mi újság a rendszermag fejlesztése körül?

„Nem állíthatom, hogy nagy híve volnék az LGPL-nek, különösen, ha figyelembe veszem a GPLv3-mal kapcsolatban újabban napvilágra került problémákat. Hogy mi ennek az oka? Az a bajom vele, hogy az LGPL-t kifejezetten úgy alkották meg, hogy az összeegyeztethető legyen a GPL-lel, de annak bármely változatával (ezt a tulajdonságát ráadásul még csak korlátozni lehet olyan módon, amelyre maga az eredeti GPL lehetőséget ad). Megtehetjük például, hogy fogunk egy LGPL 2.1-es kódot, majd átírjuk, és ezzel együtt kijelentjük, hogy az így létrejött műre már a GPLv3 vonatkozik. Ilyenkor a módosítások az eredeti, GPLv2 projekt számára már hozzáférhetetlenek lesznek." – Linus Torvalds

Valószínű, hogy a rendszermagból hamarosan el fog tűnni a ***sysctl*** hívás, amivel eddig a kernelparamétereket lehetett beállítani. Ez alapvetően ellentmond annak az elhatározásnak, hogy a felhasználói tér elemei sérthetetlenek, ebben az esetben viszont mégis valószínűbb, hogy a fejlesztők kivételt tesznek. Az ok egyszerű: nem tudnak olyan a felhasználói térben működő programról, ami ***sysctl*** hívásokat használnak. Úgy tűnik, mostanra mindenki átállt valamilyen más módszerre. Ha tehát a tisztelt olvasó most felsikoltott, vagy tud olyan emberről, aki ekként fog viselkedni a hír hallatán, akkor fontolja meg, hogy ír egy levelet a fejlesztői listára, ahol a dolog egyelőre csak a javaslat szintjén van, tehát maradt még idő a beavatkozásra. Amúgy ***Linus Torvalds*** és ***Andrew Morton*** is kifejezte azon véleményét, hogy valóban a ***sysctl*** rendszermagból való eltávolítása lenne a legjobb megoldás. ***Linus*** szerint a dolgot egyszerűen az indokolja, hogy már senki nem használja a kérdéses szolgáltatást, ***Andrew*** szerint pedig egyenesen szégyen volna, ha hagynák, hogy mindenféle használaton kívüli kódrészletek összerondítsák a forráskódot, miközben már régen létezik megfelelő alternatíva. Arra az esetre, ha mégis kiderülne, hogy valami nagyon nem fog működni a ***sysctl*** nélkül, ***Albert Cahalan*** elvállalta, hogy átveszi a ***sysctl*** hivatalos fenntartójának szerepét.

A ***Mutimedia Card*** alrendszert mostantól hivatalosan ***Multimedia Card and Secure Digital*** alrendszernek hívják, ***Pierre Ossman*** pedig benyújtott egy foltot, amely immár őt nevezi meg a kód hivatalos gazdájaként. ***Russel King***, az előző gazda ugyanis egy ideje felhagyott a fejlesztéssel, és „árvára" (***orphaned***) állította annak állapotjelzőjét. Mindeközben ***Jiri Slaby*** új fönntartói bejegyzésekkel bővítette a ***Moxa SmartIO/IndustIO Serial Card*** meghajtó, illetve a ***Mutitech Multiport Card*** kódját. Mindkét esetben önmagát jelölte meg hivatalos fenntartóként.

Egy magát megnevezni nem kívánó rendszermagtesztelő olyan méréseket tett közzé, melyek szerint az ***ext4*** fájlrendszer körülbelül 20 százalékkal gyorsabb, mint az ***ext3*** vagy akár a ***Reiser4***. Bár az eredmény hitelesnek tűnik (sőt kifejezetten örömre ad okot), ***Theodor Ts'o*** az eset kapcsán arra hívta fel a figyelmet, hogy pillanatnyilag a legnagyobb szükség egy olyan automatikus rendszerre lenne, amivel bármely fájlrendszer sebességét, illetve általános teljesítményét mérni lehet. Ez ugyanis lehetőséget teremtene arra, hogy egyrészt szabványos módon összehasonlítsák a különböző fájlrendszereket, másrészt hitelesen megvizsgálják, hogy egy-egy folt alkalmazása gyorsítja vagy lassítja a dolgokat. Ehhez csatlakozva többen máris javasolták hasonló eredmények közzétételét más fájlrendszerekkel kapcsolatban is. Ezzel amúgy az eredeti levél beküldője (illetve ***Ted***) is egyetértett, ugyanakkor hangsúlyozta, hogy egy ilyen automatizált infrastruktúra megalkotása kifejezetten nagy falat. Összefoglalva tehát a közeljövőben nem reménykedhetünk abban, hogy automatikus tesztrendszerünk lesz, bár az is igaz, hogy az ilyen kijelentésekkel nem árt óvatosan bánni...

Úgy tűnik wiki-szezon van kernelföldén. ***Valerie Henson*** rögtön két wikit is összehozott: egyet a fájlrendszerekről (➲ ***http://linuxfs.pbwiki.com***) egy másikat pedig a nagy méretű memórialapokról (➲ ***http://linux-mm.org/HugePages***). Amint az várható volt, a fájlrendszerekkel kapcsolatos wiki jóval aktívabb, mint a másik. A két közösségi projekthez kapcsolódva ***Valerie*** elindított két új ***IRC*** csatornát is az ***irc.oftc.net*** kiszolgálón ***#linxfs*** és ***#hugepages*** néven. Eközben ***Darren Hart*** és ***Theodore Ts'o*** is elindítottak egy wikit, amely a valós idejű alkalmazásokkal, illetve azok támogatásával foglalkozik. Ez a wiki a ➲ ***http://rt.wiki.kernel.org*** címen érhető el. Itt talán érdemes még megjegyezni, hogy a ➲ ***wiki.kernel.org*** webhely örömmel ad helyet minden olyan közösségi információs adatbázisnak, amely valamilyen hivatalos kernelfejlesztési projekthez kapcsolódik. Ha tehát valaki maga is indítani szeretne egy az előzőekhez hasonló wikit, nincs más dolga, mint megkérni a rendszergazdákat, hogy állítsák be neki a szükséges tárhelyet. Ugyanakkor ***Ted*** arra is fölhívta a figyelmet, hogy wikit indítani csak akkor van értelme, ha előre tudni lehet legalább két-három olyan emberről, aki szívesen elvállalja a szerkesztő és fenntartó szerepét. Ennek hiányában ugyanis a wikik általában elég elhagyatottak szoktak lenni. Márpedig egy elárvult wikinél lehangolóbb dolog kevés van ezen a világon.

Linux Journal 2007., 154. szám

Zack Brown

Tanuljunk JavaScriptet, óra indul...

Cím: Tanuljuk meg a JavaScript használatát 24 óra alatt
Szerző: Michael Moncur
Kiadó: Kiskapu (eredeti: SAMS)
Oldalszám: 472
Ár: 5980 Ft

Valószínűleg a kedves Olvasó már akarata ellenére is találkozott ***JavaScript***-tel az interneten böngészve. A legtöbb regisztrációs oldalnál az adatok elküldése előtt például ***JavaScript*** figyeli, hogy minden szükséges mezőt kitöltöttünk-e. Vagy például webes levelezőklienseknél is figyelmeztethet, ha nem adtunk meg címzettet.
A ***24 órás sorozat***ban 2006 őszén megjelent ***JavaScript***-es könyv nem kevesebbet céloz meg, minthogy az Olvasóból - ha rászánja magát a tanulásra – ***JavaScript*** gurut faragjon.
A címben szereplő 24 óra kicsit talán kevés a könyvben tárgyalt anyag elsajátítására és megfelelő mértékű gyakorlására, de mindenképp érdemes belefektetni időt.
Aki ismeri a sorozat többi programozási témájú tagját (***PHP4***, ***PHP5***, ***Java***, ***XML***, ***Mysql***, stb.), az tudja, hogy a szerzők minden esetben az alapoktól kezdenek, így nem jelent hátrányt, ha az Olvasó az adott témában esetleg teljesen járatlan.
Ennek a jó szokásnak megfelelően az első blokk (négy fejezet) bevezet a ***szkriptnyelvek*** és a ***DOM (dokumentumobjektum-modell)*** alapjaiba, míg a tényleges ***JavaScript*** bevezetés a második részben található. A ***JavaScript*** nagy előnye a többi programozási nyelvvel szemben, hogy szkript nyelv, így a könyvbeli példák kipróbálásához nem szükséges bonyolult fejlesztői környezet vagy szerver. Elég csupán egy szövegszerkesztő (legyen az ***vi***, ***scite*** vagy bármi más) és egy webböngésző bekapcsolt ***JavaScript*** támogatással. Szerver csupán az ***AJAX***-os fejezetnél létfontosságú.
A ***DOM*** igen fontos szerepet játszik a ***JavaScript*** használhatóságában és népszerűségében, így a szerző a harmadik – és egyben a leghosszabb – részben tárgyalja a témát.
A haladó lehetőségek (negyedik rész) felsorolása közül nem maradhat ki a manapság oly divatos ***AJAX (aszinkron JavaScript és XML)*** és a ***Firefox*** kiegészítése, a ***Greasemonkey*** sem. Ebben a részben tér ki a szerző a ***JavaScript*** kulturált használatára is, mint például hogyan írjunk a felhasználó szemszögéből nézve nem tolakodó ***JavaScript*** alkalmazást, illetve hogyan tehetjük böngészőfüggetlenné művünket.
Az ötödik részben a multimédia kapott helyet, kulcsszavakban: dinamikus képek, diabemutató, ***Flash*** animációk beillesztése, hangok.
A hatodik – és egyben utolsó – rész levezetésképp a lenyíló menük hogyanját és a ***Javascript*** jövőjét vázolja.
Munka után pedig jöhet a megérdemelt pihenés: a 22. fejezetben egy ***JavaScript***-es pókerpassziánszt mutat be a szerző.
Minden fejezetet (órát) a sorozatra jellemzően összefoglaló blokk és ismétlő kérdéssor zár.
A könyvet egyaránt tekintheti a kedves Olvasó tankönyvnek, hisz az alapoktól kezd. Tekintheti gyors referenciának is, hisz a tartalomjegyzékben tematikusan, míg a bőséges tárgymutatóban kulcsszavak alapján kereshetünk.
AJAX tanulás esetén a most bemutatott ***JavaScript***-es könyv mellé jó választás a ***24 órás sorozat XML***-ről, valamint a ***PHP5***-ről szóló része is.

Medve Zoltán
(e-medve@e-medve.hu)

2001-ben kezdett „Linuxolni”, de már korábban is ismerkedett a szabad szoftverek világával.
Ha éppen nem a gép előtt ül, akkor fotózgat, olvasgat vagy bicajozik.

Újabb három vállalat részesül a Microsoft és a Novell együttműködésének előnyeiből

A Deutsche Bank AG, a Credit Suisse és az AIG Technologies a nyílt forráskódú és a jogvédett szoftverek együttes alkalmazásával tér át vegyes forráskódú rendszerre

A *Microsoft* és a *Novell* 2006. november 2-án széles körű üzleti és műszaki megállapodást jelentett be, amelynek célja a két vállalat termékeinek jobb együttműködését biztosító megoldások fejlesztése, értékesítése és támogatása. *Stuart Cohen*, az *Open Source Development Labs.* vezérigazgatója szerint a *Microsoft* és a *Novell* bejelentése jelentős mérföldkő a *Linux* bevezetésének történetében. Ezek a megoldások hatékony virtualizációs lehetőségeket, valamint a *Microsoft* és a *Novell* termékeinek jobb együttműködését kínálják.

A megállapodás részeként a *Novell* és a *Microsoft* három fontos elkötelezettséget is bejelentett. Először, a *Microsoft* együttműködik a *Novell* vállalattal és aktívan hozzájárul számos nyílt forráskódú szoftverprojekthez, többek között *Office* fájlformátumokkal és webes szolgáltatások felügyeletével kapcsolatos projektekhez. Másodszor, a *Microsoft* nem érvényesíti szabadalmait a nem kereskedelmi nyílt forráskódú szoftverek fejlesztőivel szemben. Harmadszor, a *Microsoft* ígéretet tesz arra, hogy nem érvényesíti szabadalmi igényeit az *OpenSUSE.org* olyan egyéni fejlesztőivel szemben, akiknek kódját a *SUSE Linux Enterprise* platform tartalmazza, beleértve a *SUSE Linux Enterprise Server* és *SUSE Linux Enterprise Desktop* termékeket.

Stuart Cohen, az *Open Source Development Labs.* vezérigazgatójának véleménye szerint azzal, hogy a Microsoft az együttműködést választotta, elismeri azt, hogy a nyílt forráskód kulcsfontosságú szerepet játszik a vállalati informatikai infrastruktúrában. Az *Open Source Development Labs.* nagyra értékeli a *Novell* szerepét a *Microsoft* és a nyílt forráskód közösség közötti híd felépítésében és örömmel látja, hogy a két vállalat együttműködik a szabadalmak által okozott, a fejlesztőket és ügyfeleket érintő jogi fenyegetések megszüntetése érdekében is. Ez növeli az ügyfelek *Linux* iránti bizalmát, és jó hatással van a nyílt forráskód közösségre, valamint a szélesebb körben értendő informatikai-gazdasági rendszerre.

Az ügyfelek már most élvezhetik a megállapodás eredményeit

A *Deutsche Bank AG*, a *Credit Suisse* és az *AIG Technologies* az elsők között részesül a *Microsoft Corp.* és a *Novell Inc.* – *Windows* és *Linux* rendszerek együttműködését javító – széles körű üzleti és műszaki megállapodásának eredményeiből. A *Microsoft* három különálló ügyfél-megállapodás keretében mindegyik vállalat részére *SUSE Linux Enterprise* előfizetési tanúsítványokat biztosít, így lehetővé teszi számukra a *Microsoft* és a *Novell* között született megállapodás előnyeinek kihasználását. A *Credit Suisse*, a *Deutsche Bank* és az *American International Group Inc.* tagjaként működő *AIG Technologies* hangsúlyozza, hogy az együttműködés legfőbb előnyei, a szabadalmi együttműködési megállapodás és a kétirányú virtualizációs megoldásokra vonatkozó tervek döntő tényezők voltak választásuk során.

Deutsche Bank: kihasználhatják a platformválasztás előnyeit

Clemens Jochum, a *Deutsche Bank AG* műszaki igazgatója örömmel tekint a két vállalat megállapodása elé, melynek köszönhetően a jövőben javul a különböző rendszerek közötti együttműködés. A *Deutsche Bank* a *Novell* és a *Microsoft* termékeinek már most jelentős felhasználója, és régóta fontosnak tartja a *Windows* és *Linux* rendszerek együttes alkalmazását, azonban a két rendszer együttműködésének megteremtése jelentős időt és erőforrást igényelne. A *Microsoft* és a *Novell* megállapodásának eredményeképpen a vállalat kihasználhatja a platformválasztás előnyeit, és az ebből adódó rugalmasság révén tovább növelhetik vállalatuk versenyképességét és hatékonyságát.

Credit Suisse: lehetőség a nyílt forráskódú és a jogvédett szoftverek együttműködésére

Tom Sanzone, a *Credit Suisse* informatikai igazgatója szerint a *Credit Suisse* számára kulcsfontosságú ez az együttműködés, hiszen a *Windows* és a *SUSE Linux* rendszerek folyamatosan fejlődő, stratégiai fontosságú platformok. A vállalat örömmel üdvözli, hogy a *Microsoft* és a *Novell* mindkét platformot támogatva folytatja az együttműködés fejlesztését, és az új modell az ügyfelek számára eddig nem tapasztalt lehetőséget biztosít a nyílt forráskódú és a jogvédett szoftverek együttműködésére.

AIG Technologies: felfedezi, miként válhatna még rugalmasabb szervezetté

Mark Popolano, az *AIG* globális informatikai igazgatójának célja, hogy vállalata piaci áron magas értéket kínáljon – így kulcsfontosságú számára, hogy műszaki ajánlatai előretekintők,

integráltak és megfelelően pozicionáltak legyenek. *Popolano* véleménye szerint a *Microsoft* és a *Novell* elkötelezettsége az együttműködés és a szellemi tulajdonjogok biztosítása terén nagymértékben hozzájárul e cél eléréséhez. Az *AIG* a programba való belépéssel azt tervezi, hogy felfedezi, miként válhatna még rugalmasabb szervezetté.

A *Deutsche Bank*, a *Credit Suisse* és az *AIG Technologies SUSE Linux Enterprise* bevezetésére vonatkozó – közvetlenül a két vállalat együttműködésének bejelentése után született – döntése megerősíti a *Microsoft* és a *Novell* közös munkájának eredményét. Az ügyfelek igényeihez igazodva a *Novell* egy valós probléma megoldásával biztosít egyedülálló lehetőséget a *Linux* bevezetéséhez. Ez is azt bizonyítja, hogy a *SUSE Linux Enterprise* teljes mértékben készen áll az adatközpontokban való alkalmazásra.

Az ügyfelek 95 százaléka támogatja a Microsoft és a Novell közötti együttműködést

A *Microsoft* és a *Novell* közös, a *Penn, Schoen & Berland Associates Inc.* független piacelemző cég által végzett, 201 informatikai szakembert megkérdező felmérése szerint a válaszadók 95 százaléka egyetért a *Microsoft* és a *Novell* közötti megállapodással, és úgy véli, hogy az informatikai rendszerek együttműködésének javítása előnyöket biztosíthat ügyfelei számára. Eddig több mint 16 000 új *SUSE Linux Enterprise*-tanúsítványt aktiváltak a *Microsoft* és a *Novell* együttműködési megállapodásának keretében. A felmérés főbb eredményei:

- A megkérdezettek 95 százaléka egyetért a *Novell* és a *Microsoft* között létrejött megállapodással.
- A megkérdezettek 78 százaléka gondolta úgy, hogy az ügyfelek számára előnyös, ha a *Microsoft* és a vezető *Linux*-disztribútorok szorosabban együttműködnek.
- A megkérdezettek 67 százaléka nagyobb valószínűséggel választaná a *Novell SUSE Linux* termékét.
- A megkérdezettek 97 százalékának véleménye szerint szükség van arra, hogy a platformszolgáltatók javítsák termékeik együttműködését.
- A megkérdezettek 98 százaléka szeretné, ha a műszaki vállalatok felelősséget vállalnának a termékeikben szereplő szellemi tulajdonért.

A teljes felmérés a ➲ ***http://www.novell.com/news/press/item.jsp?id=1257*** címen érhető el.

KAPCSOLÓDÓ CÍMEK

A megállapodásról további információt a
➲ http://www.novell.com/linux/microsoft és a
➲ http://www.microsoft.com/interop/msnovellcollab weboldalakon talál.

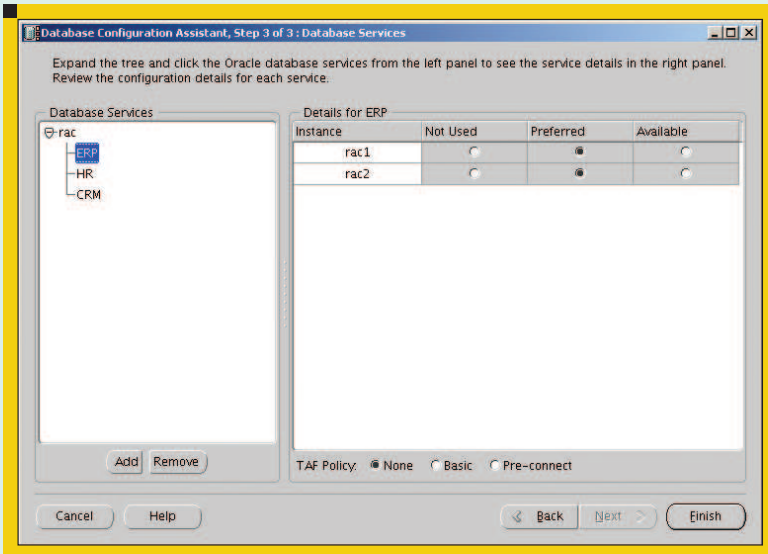
Oracle Real Application Clusters Linuxon

Hagyományos fürtözés

A fürtözés *Linux* platformon már régen nem számít kuriózumnak. A *Linux* a szerver piacon tapasztalható térnyerését nem csak az olcsó processzor architektúrák folyamatos teljesítmény növekedésének köszönheti, hanem mert képes megfelelni más vállalati igényeknek is, mint például a magas szintű rendelkezésre állás követelményének. A fürtözés manapság a legelterjedtebb megoldás szerver meghibásodások kezelésére.
Az operációs rendszerek által kínált fürtözés a szolgáltatások folyamatos elérhetőségét úgy biztosítják, hogy kijelölnek egy aktív-passzív szerver párost. Ezen túlmenően az adatbázis szolgáltatás magas szintű rendelkezésre állásának a biztosításához mindenképpen szükséges az adatbázis adatokat tartalmazó fizikailag közös, merevlemezes tároló, amelyet logikailag mindig az éppen aktív szerver tud csak elérni.

Mire szolgál a RAC?

Az *Oracle Real Application Clusters (RAC)* az operációs rendszer fürt által kínált funkcionalitásnál többet nyújt. A *RAC* estében ugyanis a szerverek egy időben képesek elérni a közös diszken lévő adatbázist. Lehetővé teszi egyetlen adatbázis transzparens telepítését egy több hardverből álló szerverfürtre, biztosítva a hibatűrő működést hardverhibák vagy tervezett leállások esetén is. Az *Oracle RAC* fürtözött működése maximális rendelkezésre állást és méretrugalmasságot biztosít az olcsó hardvereken futtatható *Oracle* rendszereknek. Ez a megoldás *Linux* platformon termékeny talajra talál, hiszen nagyvállalati igények kiszolgálását teszi lehetővé olcsó eszközökkel.



1. ábra DBCA

Az *Oracle RAC* együttműködik a legkülönbözőbb elterjedt üzleti alkalmazásokkal, köztük az online tranzakciós (*OLTP*) és a vezetői döntéstámogató (*DSS*) környezetekkel, de hatékony módon támogatja a vegyes *OLTP*/*DSS* környezeteket is. Így a fürtözött adatbázis a saját fejlesztésű alkalmazások mellett jól használható az olyan ismert kereskedelmi szoftverekkel is, mint az *SAP*, *PeopleSoft*, *Siebel* és *Oracle E-Business Suite*.

Méretrugalmasság

Az *Oracle Real Application Clusters* lehetővé teszi az alkalmazások rugalmas méretezését. Nagyban segíti a költségek csökkentését, hogy a fürtözött rendszereket szabványos, tömegcikk-árfekvésű szerverekből, tárolókból és hálózati eszközökből lehet összeállítani. Ha nagyobb feldolgozási kapacitásra van szükség, akkor egyszerűen egy újabb kiszolgálóval lehet bővíteni a rendszert anélkül, hogy a felhasználóknak félbe kellene szakítani munkájukat. Az *Oracle Clusterware* és az *Oracle RAC* egy fürtben akár száz csomópontot is képes kiszolgálni.
Így amikor a szervezet kinőtte hardverkapacitását, semmit nem kell lecserélni, hiszen az *Oracle RAC* „horizontális" méretezést tesz lehetővé a fürt újabb hasonló szerverekkel bővítése útján, ami egyúttal a folyamatos rendelkezésre állást is szolgálja. A *RAC* a teljesítménynövekedést, a fürtben szereplő node-ok számát figyelembe véve, közel lineárisan képes biztosítani.

Automatikus terheléselosztás

Az *Oracle Database 10g* esetében az alkalmazások feldolgozási terhelése szolgáltatásokra bontva definiálható, amelyek mindegyike külön-külön felügyelhető és szabályozható. Az adatbázis-adminisztrátor határozza meg, hogy az egyes szolgáltatások milyen

erőforrásokat (node-okat) vehetnek igénybe normál üzem esetén, illetve meghibásodás miatti átkapcsolásnál. A koncepció az, hogy minden alkalmazás adat egyetlen fürtözött adatbázisban található és minden node-on fut az adatbázisnak egy példánya (memória és processzek). A hozzáférés szabályozás érdekében, természetesen a különböző alkalmazások adatai más-más adatbázis sémában helyezkednek el. Fizikailag minden adat minden node-ról elérhető, de a szolgáltatásokkal a rendszergazda meghatározhatja, hogy egy adott szolgáltatás mely node-okról érhető el.
A szolgáltatások az ***Oracle Database Configuration Assistant (DBCA)*** eszközével egyszerűen létrehozhatók (szerverek: ***node1***, ***node2*** ; adatbázis neve: ***rac*** ; példányok: ***rac1***, ***rac2***) (*1. ábra*).

Meg kell adni a szolgáltatás nevét és azt, hogy az adatbázis példányok milyen feladatot látnak el a szolgáltatásban:

- Semmilyent: nem vesz részt a szolgáltatás nyújtásában.
- Elsődleges: normál esetben a szolgáltatás ezen a példányon keresztül elérhető.
- Elérhető: valamelyik Elsődleges példány kiesése esetén ezen keresztül is elérhető a szolgáltatás.

Egy szolgáltatás egy alkalmazáshoz, vagy akár annak egy részészéhez (pl.: lekérdező modulhoz) lehet hozzárendelve. Amikor az alkalmazás kapcsolódik az adatbázishoz, akkor csak azt kell megadnia, hogy milyen szolgáltatáshoz szeretne hozzáférni. Az adatbázis példányok bejelentkeznek a listener-hez (az ***Oracle*** szerver oldali hálózati komponenséhez) és közlik vele, hogy mely szolgáltatásokhoz nyújtanak hozzáférést, valamint rendszeres időközönként jelentik a példány terheltségét. A listener állapot képernyője így néz ki: *1. Lista*.
A listener az adatbázis példányok terheltségi adatai alapján az új adatbázis kapcsolódási kéréseket, ahhoz a példányhoz irányítja, amely a legjobb kiszolgálást tudja nyújtani.
Az ***ERP*** szolgáltatást elérni szándékozó kliens ***JDBC*** kapcsolat leírása így néz ki:

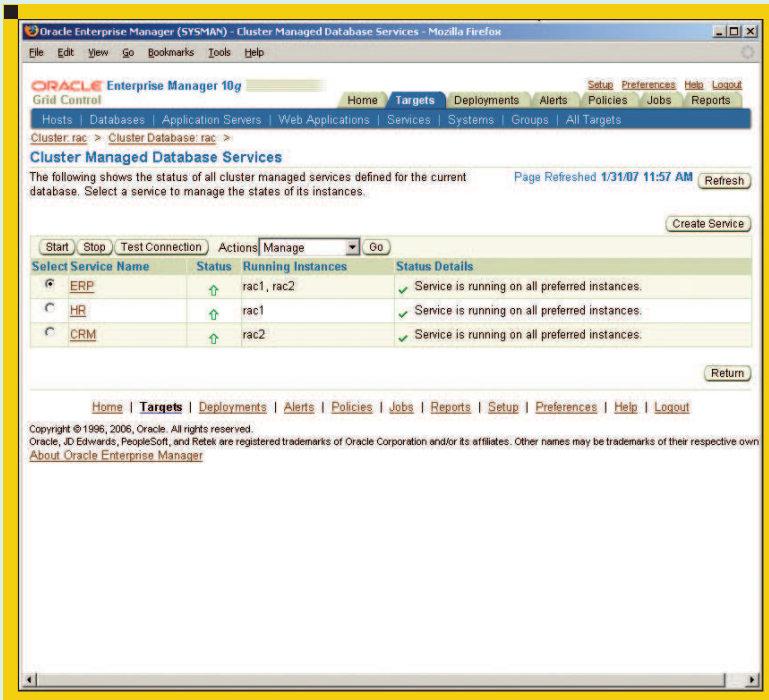
```
url="jdbc:oracle:thin:@
➥(DESCRIPTION=
➥(LOAD_BALANCE=on) (ADDRESS=
```

1. Lista

```
[oracle@node1 oracle]$ lsnrctl status

LSNRCTL for Linux: Version 10.2.0.2.0 - Production on 31-JAN-2007 11:50:55

Copyright (c) 1991, 2005, Oracle.  All rights reserved.

Connecting to (ADDRESS=(PROTOCOL=tcp)(HOST=)(PORT=1521))
STATUS of the LISTENER
------------
Alias                     LISTENER_NODE1
Version                   TNSLSNR for Linux: Version 10.2.0.2.0 - Production
Start Date                31-JAN-2007 11:30:57
Uptime                    0 days 0 hr. 19 min. 58 sec
Trace Level               off
Security                  ON: Local OS Authentication
SNMP                      OFF
Listener Parameter File   /home/oracle/db/network/admin/listener.ora
Listener Log File         /home/oracle/db/network/log/listener_node1.log
Listening Endpoints Summary...
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=10.172.212.116)(PORT=1521)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=10.172.212.118)(PORT=1521)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(KEY=EXTPROC)))
Services Summary...
Service "+ASM" has 1 instance(s).
  Instance "+ASM1", status BLOCKED, has 1 handler(s) for this service...
Service "+ASM_XPT" has 1 instance(s).
  Instance "+ASM1", status BLOCKED, has 1 handler(s) for this service...
Service "CRM" has 1 instance(s).
  Instance "rac2", status READY, has 1 handler(s) for this service...
Service "ERP" has 2 instance(s).
  Instance "rac1", status READY, has 2 handler(s) for this service...
```

1. Lista folytatás

```
  Instance "rac2", status READY, has 1 handler(s) for this service...
Service "HR" has 1 instance(s).
  Instance "rac1", status READY, has 2 handler(s) for this service...
Service "rac" has 2 instance(s).
  Instance "rac1", status READY, has 2 handler(s) for this service...
  Instance "rac2", status READY, has 1 handler(s) for this service...
Service "racXDB" has 2 instance(s).
  Instance "rac1", status READY, has 1 handler(s) for this service...
  Instance "rac2", status READY, has 1 handler(s) for this service...
Service "rac_XPT" has 2 instance(s).
  Instance "rac1", status READY, has 2 handler(s) for this service...
  Instance "rac2", status READY, has 1 handler(s) for this service...
The command completed successfully
```



2. ábra OEM ERP

```
➥(PROTOCOL=TCP)(HOST=node1)
➥(PORT=1521)) (ADDRESS=
➥(PROTOCOL=TCP)(HOST=node2)
➥(PORT=1521)) (CONNECT_DATA=
➥(SERVICE_NAME=ERP)))"
```

Az adatbázis kezelő szempontjából az alkalmazás szerver is egy adatbázis kliens. A kliens oldalon is lehetőség van terhelés megosztásra, de itt véletlenszerűen választ a felsorolt szerverek között. Ezt a választást a korábban említett módon a listener felülbírálhatja.

A szolgáltatások az adatbázis példányok újraindítása nélkül, menet közben is konfigurálhatók az ***Oracle Enterprise Manager*** nevű eszközével: *2. ábra*.

Vállalati számítóhálózatok

Az ***Oracle RAC*** lehetővé teszi a vállalati számítóhálózatok (gridek) kialakítását. A vállalati számítóhálózatokat szabványos, tömegcikk-árfekvésű szerverekből, tárolókból és hálózati eszközökből lehet összeállítani. A szerverek, tárolók, processzorok és operatív memória mind dinamikusan, igény szerint rendelhetők hozzá a feldolgozásokhoz, és a rendszer bármilyen átcsoportosítás vagy átkonfigurálás esetén elérhető marad.

Mosolygó Ferenc

az Oracle Hungary vezető adatbázis szakértője, 10 éve foglalkozik az Oracle adatbázis-kezelő magas szintű rendelkezésre állását biztosító megoldásokkal. Több országos adatbázis alapú rendszer tervezésében és telepítésében vett részt.

KAPCSOLÓDÓ CÍMEK

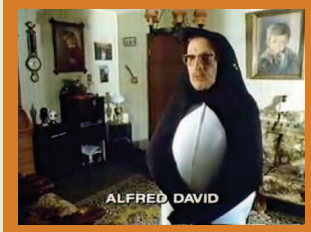
RAC telepítés VMware-re:
- http://www.oracle.com/technology/tech/linux/vmware/cookbook/index.html

- http://www.oracle.com/technology/pub/articles/chan-ubl-vmware.html

Oracle Database 10g Services:
- http://www.oracle.com/technology/products/database/clustering/pdf/twp_rac_services_10gr1_112503.pdf

Batman után szabadon – The Penguin Man

■ Azt mondják, az emberen már gyermekkorban eluralkodik a gyűjtőszenvedély; a lányok szalvétát gyűjtenek, a fiúk bélyeget. Bizonyos elméletek szerint e szenvedély csillapodik, sőt végképp alábbhagy, mikoris a felnőtté válás útján a két nem egymásra talál, majd ismét visszatér úgy 40 év fölött, a szexuális élet hanyatlásával.
Kissé korosabb műgyűjtők hevesen cáfolják az elméletet, mondván, hogyan is állná meg a helyét, hisz például ***Casanova*** is nagy gyűjtő volt – nőket gyűjtött. A gyűjtőszenvedély lélektanát azzal magyarázzák, hogy szükségük van valaminek az állandó jelenlétére, birtoklására, változatlanságára, illetve gyarapítására, ezáltal bebiztosítják az „örökkévalóság”, az el nem múlás illúzióját. Ezt én józan paraszti ésszel (és ötszöri átolvasással) meg is értem, csak valahogy nem tudom összekapcsolni a halhatatlanságot a söralátétekkel, porcelánkacsákkal és gombostűkre tűzött cserebogarakkal. Ám akármilyen cinikussá válhatnék, akkor sem tudnék lépést tartani a világ csodabogaraival.



Próbáltam magam ***Alfred David***, a belga öregúr, alias ***The Penguin Man*** helyébe képzelni: vajon mi vinne rá, hogy amolyan hobbiból a kertitörpémtől kezdve a sótartómig mindenem pingvineket formázzon, de még az alsómon és a toalettpapíromon is madarak díszelegjenek (➲ ***http://www.youtube.com/watch?v=AKZqbxiD1gI&eurl***)...
Azonban miután végignéztem, ahogy ***Alfréd*** bácsi egy szokványos délutáni napon pingvinjelmezben szaladgál a rekamié, a kredenc és a fikusz között, skandálva, hogy ő egy tipikus belga ember, rájöttem, mindenestől amputálni kellene engem, hogy párhuzamos gondolatokra lelhessünk mi kettecskén. És nem csak azért, mert nem vagyok belga... Minden elismerésem (és sajnálatom) ***Alfréd*** nénié, ki majdan urával közös, természetesen pingvin formájú koporsóban térhet meg örök nyugalomra. Azaz hogy mégsem, mert ***Alfréd*** bácsi feltett szándéka emberi létének elmúlását követően valamely állatkertben pingvinként reinkarnálódni – legalább rácsok mögötti biztonságban leszen. Nem valószínű, hogy a becses nej következő életében (is) ura állatkerti gondozójaként kívánna újjászületni, bár sosem lehet tudni... Könnyen megeshet, hogy ***Alfréd*** néni titokban képregényeket gyűjt a fészerben, s mikor a kamera nincs jelen, ***Macskanő*** jelmezben csatlakozik hites urához és a rekamiéhoz.

Halusz Léna

Ruby on Rails

Zope kontra Ruby on Rails

Néhány éve, a *.com láz* idején én is és az alkalmazottaim is tele voltunk tanácsadói munkával. Alig győztük elvégezni az összes feladatot. E virágzás közepén nyilvánvalóvá vált, hogy a projektek egészen hasonlók, így tulajdonképpen az időnk (és az ügyfelek pénzének) nagy részét minden projektnél újra és újra a spanyolviasz feltalálására fordítottuk. Ideje volt elgondolkodni azon, hogy a kódok, vagy legalább a technikák újbóli felhasználása miként valósítható meg az egyes projektek között. Feltételeztük, hogy ezáltal nemcsak versenyképesebbek leszünk, hanem napi feladataink is érdekesebbek lesznek. Hiszen sokkal érdekesebb dolog az egyes projektek új és különböző elemein dolgozni, mint állandóan szinte ugyanazt a felhasználói csoport jogosultsági rendszert létrehozni.

Hamar elvetettük a közös kódrendszer ötletét, részben azért, mert más fejlesztők nem csak, hogy megoldották már e problémák többségét, de közzé is tették megoldásaikat nyílt forráskódú licencekkel. Az évek során számtalan különböző projekthez alkalmaztunk webfejlesztő keretrendszereket – közülük be is mutattam néhányat.

Bárki, aki dolgozott már ilyen keretrendszerekkel, az tudja, hogy semmi sincs ingyen. Szinte kivétel nélkül mindegyik keretrendszer ránk erőltet egy bizonyos módszert, a kompromisszumos megoldások pedig vagy passzolnak fejlesztési elképzeléseinkhez, vagy nem. Jómagam az évek során többet is használtam e keretrendszerek közül, és bár akadt bennük kedvemre való tulajdonság, sosem éreztem, hogy szabadon engedhetem a fantáziámat. Ilyen előzmények után engem is roppantul felizgatott az új jövevény, azaz a ***Ruby on Rails***. Mint az kiderült, a ***Rails*** egy olyan keretrendszer, amelynek sok különböző funkciója van, ilyenek például az objektumrelációs leképező, az ***MVC (Model View Controller)*** tervezés, és a beépített teszteléstámogatás. A ***Rails*** nagyon népszerűvé vált első megjelenése óta, és bár itt-ott még akad rajta csiszolni való, azért kétségkívül kitűnő rendszer. Emellett népszerűsége abban is megnyilvánul, hogy az új keretrendszereket fejlesztők a ***Rails*** rendszert tekintik példaképüknek, ***Rails***-szerűt akarnak alkotni, vagy annál is jobbat.

Hogy a ***Rails*** miért érdekel ennyi embert? Vagy, fontosabb kérdés, hogy érdemes-e ezt használnunk következő web vagy adatbázis projektünkhöz? Végül, milyen kompromisszumokat erőltet ránk, és ezek hogyan befolyásolják döntéseinket?

A Rails előtt

Webes alkalmazásokat fejlesztek akkortól kezdve, amikor még a *„webes alkalmazás”* kifejezés e-mailküldő ***CGI*** programokat jelentett, nem egy milliárdos iparágat. Minden keretrendszer, amit használtam, letett valamit az asztalra, és így vagy úgy könnyebbé tette számomra az alkalmazásfejlesztést. Ugyanakkor viszont mindegyik valamilyen kompromisszumra kényszerített pusztán azért, hogy a rendszerrel dolgozni tudjak.

Az első általam használt keretrendszerek egyike a ***Mason*** volt, amely elkényeztetett rugalmasságával és könnyen használhatóságával. ***Perlben*** íródott, és a ***mod_perl***-hez és az ***Apache***-hoz tervezték. Telepítése és beállítása roppant egyszerűvé vált az évek során, persze ***mod_perl*** és ***Apache*** még szükséges hozzá a kiszolgálón. Továbbá a ***Mason*** tökéletesen együttműködik több, a ***CPAN***-on elérhető ***Perl*** modullal, azokkal a fejlett és hatékony fejlesztőeszközökkel, amelyeket a ***Perl*** közösség létrehozott az évek alatt. Amikor ***Perllel*** kell online rendszert létrehoznom, mindenképp a ***Masonhoz*** fordulok. Hátránya viszont, hogy kevés komponens érhető el hozzá. Persze megírhattam volna a felhasználói fiók-, csoport- és jogosultságkezelést. De írtam volna meg külön, minden egyes projekthez? És bár a ***Mason*** sablonjai igen hatékonyak, töménetlen ***Perl*** kódot és szokatlan szerkezetet tartalmaznak, ez pedig riasztóan hathat bizonyos fejlesztőkre.

Ezután megpróbálkoztam az *OpenACS*-sel. Ez egy nyílt forrású közösségi rendszer, és jóval kevesebb felhasználója van, mint a *Masonnak*. Az *OpenACS* sablonrendszere viszont minden megjelenített oldalt két részként kezel. Az egyik *Tcl*-ben íródott, a másik pedig módosított *HTML*, melyeket egy meghatározott „szerződés" kapcsol össze. Továbbá az *OpenACS* rendelkezik egy szabványos adatmodellel, amelyet a rendszer különböző alkalmazásai használnak. Nem kell olyasmivel bajlódnunk, mint egy ilyen regisztráció modul megszerkesztése, mert a rendszerben benne foglaltatik már egy. Továbbá fórumokat, blogokat vagy naptárakat sem kell magunknak létrehoznunk, mert ezek is adottak.
A központosított, szabványos adatmodell és az adminisztratív alkalmazások bizonyára igen vonzóak, de az *OpenACS*-nek is megvannak a maga problémái. A legnagyobb talán az adatmodelljének szokatlan megvalósítása: relációs adatbázissal követi a hierarchiákat és az objektumokat. A rendszernek van egyfajta intellektuális varázsa; a relációs adatbázisok gyorsak, szilárdak és olcsók, és az objektumorientált programozás segítségével bármilyen adattípus modellezhető. Viszont e kettő kapcsolata azzal jár, hogy a legegyszerűbb *OpenACS* alkalmazás is igen összetett lesz. Sőt, az *OpenACS* közösség növekedésével igen nehézzé vált az adatmodellek kis méretének megőrzése, hiszen a felhasználók igényei igencsak különbözőek.
Foglalkoztam *Zope*-pal is. Ez egy főleg *Pythonban* megírt webfejlesztői keretrendszer. Nagy és erős közössége van, fejlesztése folyamatos, és a *Zope Corporation* folyamatosan bővíti. Több ígéretes funkcióval rendelkezik, például tartalmaz egy nagy teljesítményű fejlesztői környezetet, alosztályokra tagolt „termékeket", amelyeket különkülön felhasználhatunk és bővíthetünk, valamint egy kifinomult felhasználó-, szerep- és jogosultságkezelő rendszert. A *Zope* elsőként állt elő az objektumok közvetlen megjelenítésének ötletével, amelyben egy *URL* adja meg az adott objektumon meghívandó tagfüggvényt. Tehát a */Foo/bar URL* azt jelenti, hogy meghívjuk a *Foo.bar*-t, bemenetet a *HTTP* kéréssel biztosítunk, a kimenet pedig *HTTP* válaszként jelenik meg.
A leggyakoribb kifogás a *Zope*-pal szemben az, hogy nehéz megtanulni. Nos, ez igaz is; beletelt egy időbe, mire megértettem magát a *„Zope zent"*. Szerintem viszont az egyszerűnek vélt dolgok is igényelnek a tökéletes működéshez némi kódakrobatikát – ami lehet a kódírási stílusom reflexiója, de lehet a *Zope*-tervezés remeke és a *Zope*-beli objektumhasználat átható módja is.
Régebben a *Zope* tervezők saját objektum-orientált adatbázis építésével próbálták elkerülni a relációs adatbázisokkal kapcsolatos problémákat. Ez egyrészt számos nagy előnnyel járt a *Zope* riválisaival szemben, mert a rendszer változtatásait vissza lehet vonni, beépített jogosultságkezelő és -tároló rendszere van, amelyek a *Zope* adattípusaihoz tökéletesen illeszkednek.
A relációs adatbázisok miatt viszont elkerülhetetlen az *SQL* használata.
A *Zope DTML* sablonnyelve lehetővé teszi az adatbázisokhoz való kapcsolódást és a velük való munkát.
Ez azzal jár, hogy a *Zope* termékek többségének – és az általam használt összes termékének – koordinálniuk kell a relációs és objektum adatbázisokat. Általában vége ez a megoldás nem olyan szörnyű, de a végén mindig azon töprengtem, hogy miért kell így túlbonyolítani a dolgokat. És minden kifinomultsága ellenére azt vettem észre, hogy folyton ugyanazokat a létrehozó-, frissítés- és törlés tagfüggvénytípusokat és sablonokat hoztam létre, újra és újra.

Hogy jön a képbe a Rails?

Nem meglepő tehát, hogy a *Ruby on Rails* részben a korábbi keretrendszerek által meghagyott űrt tölti be. Fenti leírásaim talán rávilágítanak arra is, hova kéne fejlődnie a *Railsnek* ahhoz, hogy igazán sikeres legyen.
A *Rails* legvonzóbb tulajdonságai a sebessége és az az egyszerűsége, ahogyan a fejlesztők a relációs adatbázisokkal kommunikáló kódot írhatnak. És, bár egyáltalán nem nyűgöznek le a negyed óra alatt létrehozható demók, tapasztalataim mégis azt igazolták, hogy ezek a demók igenis valószerűek. Ez azért van, mert a *Rails* feltételezi, hogy adatbázistábláinkat az ő szabályai alapján hozzuk létre. Ilyen szabályok például a többes számú táblanevek, az *id* nevű *ID* mezők és az *_at*-tel végződő időpont és dátum mezők.
Ha betartjuk ezeket a szabályokat, rácsodálkozunk majd, hogy milyen nevetségesen kevés kóddal milyen sok szokványos szituációt kezelhetünk. Csakugyan, azon kapjuk majd magunkat, hogy egy maroknyi kóddal elintézünk egy rakás modellobjektumot, mert a *Rails Active Record* leképezője elvégzi helyettünk a feladat nagy részét.
Ez azt jelenti, hogy a *Rails* alkalmazásokhoz szükséges munka jó része a vezérlőkre (azok az objektumok, amelyek tagfüggvényeit az *URL*-eken keresztül tettük hozzáférhetővé) és a nézetre (*Ruby-HTML* keverék sablonok) koncentrálódik. Minden vezérlőfüggvény előállíthatja saját kimenetét egyszerű szövegként vagy *HTML*-ként, vagy a nézet könyvtárban ugyanazon nevű sablonján keresztül.
A *Railsben* van egy beépített funkció is, amellyel fájlt küldhetünk a felhasználónak, így beállíthatunk bináris fájl letöltést anélkül, hogy a *MIME* típusok és fájlnevek meghatározásának szintaxisára figyelnünk kéne.
A *Zope* védelmezői kétségkívül azt fogják mondani, hogy ezek a funkciók a *Zope*-ban is megtalálhatók, ráadásul évek óta. Ez igaz is – de az újoncoknak őrjítően bonyolult lehet kiókumulálni, hogy hogyan kell használni őket és pontosan mit is csinálnak. Azzal, hogy néhány ésszerű alapbeállítást biztosít jó pár művelethez, és engedi ezeket az alapbeállításokat módosítani, a *Railsnek* sikerült az egyszerű eseteket egyszerűnek meghagyni, és a bonyolult esetek is csak mértékkel bonyolítani. Ezenkívül a *Rails scaffolding* generátora (állványzat generátora) bőségesen elég alapvető vezérlőt és sablont biztosít, amelyeket bárki felhasználhat anélkül, hogy órákat kéne eltöltenie különböző kódfájlok létrehozásával és módosításával.
Az ésszerű alapbeállítások miatt a kezdő *Rails* fejlesztőnek csak néhány objektummal és tagfüggvénnyel kell tisztában lennie egy alkalmazás létrehozásához. Ez teljesen az ellenkezője a fentebb bemutatott keretrendszereknek, melyek nagy objektum- és tagfüggvényismeretet igényelnek olyan részletekbe menően, hogy hogyan kell kapcsolódniuk az egyes elemeknek ahhoz, hogy működjenek is. Az igaz, hogy a *Rails* bővül és finomodik, s ez azzal a kockázattal jár, hogy ugyanúgy felduzzad, mint összetettebb társai.

Eddig még sikerült elkerülnie az ilyen problémák többségét, és úgy tűnik, a fejlesztők ragaszkodnak az egyszerűség megőrzéséhez.

Érvek és ellenérvek

Ahogy azt fentebb is írtam, mindegyik webfejlesztő keretrendszernek megvannak a maga tervezési kompromisszumai. Mi hiányzik még a ***Rails***-ből? Mit kell figyelembe vennünk, annak eldöntéséhez, hogy ***Railst*** vagy valami mást használjunk-e alkalmazásunkhoz?

Először is, a ***Railshez*** elengedhetetlen a ***Ruby*** nyelv ismerete. Mielőtt elkezdenék a ***Railsszel*** dolgozni, előtte mindig foglalkozom egy keveset a ***Ruby***-val, és jól szórakozom. Persze nyilván sok programozó kifogásolja a ***Ruby*** bizonyos elemeit, például a szintaxist és az objektummodellt. A ***Ruby*** a ***Perlhez*** és a ***PHP***-hoz képest éretlen, ha a külső gyártótól származó kiegészítő könyvtárakról van szó. Ez azt jelentheti, hogy néhány speciális eljárást magunknak kell megírnunk, és nem számíthatunk a közösség támogatására. Végül, a ***Ruby***-ból hiányzik a ***Unicode***-támogatás, így többnyelvű weboldalaknál egyelőre nem alkalmazható.

Ha a mérleg nyelve a ***Ruby*** felé billen, és ezt szeretnénk webfejlesztő nyelvként használni, miért ne használhatnánk a ***Railst*** is? Nem értek egyet azzal a kijelentéssel, hogy minél kisebb a fejlesztőcsapat és minél nagyra törőbb a projekt, annál inkább érdemes ***Railst*** használni. Egészen kicsi projekteknél a ***Ruby*** csak felesleges többletteher, és sokkal kézenfekvőbb megoldás a ***CGI*** és a ***PHP***. Onnantól kezdve viszont, hogy relációs adatbázist használunk egynél több táblával, a ***Rails*** hasznunkra lehet.

A ***Ruby*** és a ***Rails*** kis létszámú programozói csoportok, valamint egyedül dolgozók számára terveztetett. Ahhoz, hogy többen dolgozzanak ugyanazon a ***Rails*** projekten, nagyfokú összehangoltság kell, hogy nehogy módosítsák egymás fájljait. Az, hogy mindegyik ***Rails*** alkalmazás egyetlen könyvtárban kap helyet, növeli az ilyen zűrzavar valószínűségét.

A nagyobb projektek viszont inkább a nagyobb keretrendszerekkel járnak jól, például a ***Zope***-pal, vagy valamelyik, nemrégiben megjelent ***Java*** alapú rendszerrel. A ***Javával*** nekem a legnagyobb gondom az volt, hogy kicsit kényelmetlen és lassú, főleg ha a ***Perlhez***, a ***Pythonhoz*** vagy a ***Ruby***-hoz hasonlítjuk. Ha több programozó dolgozik egy nagy projekten, előnyt jelenthet a több fordítás közbeni ellenőrzés, a világos meghatározások és a védelmek – ezek a ***Ruby***-ból hiányoznak. Talán webtervezőink igényeit is figyelembe kell vennünk, mikor a ***Railsről*** mint környezetről gondolkodunk. Néhányuknak ínyére van, ha a kóddal a sablonokon belül dolgozhatnak, némelyikük viszont irtózhat tőle és képes törölni vagy megváltoztatni a kódot. Még mindig úgy vélem, hogy a ***ZTP*** és az ***OpenACS*** jobb sablonrendszereket kínálnak. Örömmel fogadtam viszont a ***Liquid*** megjelenését. Ez utóbbi egy sablonrendszer a ***Railshez*** a ***PHP Smarty***-jának stílusában. Régebb óta lenyűgöz a ***Smarty***, és úgy vélem, a ***Liquiddel*** közelebb került a ***Rails*** nagyobb rendszerekben való bemutatkozása.

Előnyei és hátrányai is vannak annak, hogy a ***Rails*** alkalmazások egyetlen könyvtárban, szöveges fájlokból állnak. Egyik nagy előnye, hogy minden könnyedén tárolható ***CVS***-ben vagy bármilyen verziókövető rendszerben; az alkalmazás telepítése egy új helyre így ugyanolyan egyszerű, mint a kód ellenőrzése. Viszont ez a megközelítés azzal jár, hogy valamivel nehezebb ugyanannak az alkalmazásnak több példányát létrehozni ugyanazon a kiszolgálón, mint ugyanezt az ***OpenACS*** csomagjaival vagy a ***Zope*** termékekkel megvalósítani. Mindig létrehozhatunk több másolatot a ***Rails*** alkalmazás könyvtárfájáról, de ugyanarról a csomagból több példány nem lehet. Korábban már elárultam, hogy eredetileg nagyon vonzott az ***OpenACS*** az egyszerű, szabványosított adatmodellje miatt. Most már tudom, hogy egy ilyen erősen központosított adatmodell majdnem mindig alkalmatlan, de még mindig azon gondolkodom, hogy a ***Railsnek*** miért nincsenek beépített jogosultságai vagy regisztrációs rendszere. Feltételezem, a válasz az, hogy a növekvő számú ***Rails*** bővítőmodulok miatt, amelyek között számos, a már meglevő ***Rails*** alkalmazásokba beépíthető regisztrációs rendszer akad.

Az is jó lenne, ha ezen a területen több szabványosítás történne – de valószínűleg ez meddő ötlet.

Az örökölt kódok minden környezetben, így a ***Railsben*** is problémát jelentenek. A ***Rails*** annyira új, annyira más, jelentheti ez azt, hogy az adoptálás miatt mindent el kell dobnunk? Elképzelhető, de nem szükségszerű. Ahelyett, hogy újraírtam volna ***Ruby***-ban egy érett ***Perl*** könyvtárt, csak egy burkolót írtam ***XML-RPC***-vel. A ***Ruby***-nak van egy ***XML-RPC*** ügyfele, ezt használtam ***Rails*** alkalmazásomban a ***Perl*** kóddal való kapcsolattartásra. Mindez simán és könnyen ment, így ez azt jelenti, hogy egyszerre élvezhetem a ***Rails*** és a ***CPAN*** előnyeit. Mivel a ***Railsben*** az adatbázis elnevezési konvenciók felülírhatók, ezért az már meglevő adatbázisokkal is használható, így a felhasználóknak nem kell új adatbázissémákat létrehozniuk a ***Rails*** szabályainak megfelelően.

Összefoglalás

Vannak, akik a ***Rails*** eljövetelét várják, mellyel új időszámítás kezdődik a webfejlesztésben. Jómagam úgy látom, hogy a ***Rails*** új szabványt teremtett, már ami a webfejlesztői keretrendszerek szolgáltatásait illeti. A fejlesztők nem élhetnek tovább abban a tévhitben, hogy terjedelmes kód kell mind a ***„Hello, World!"*** program megírásához, mind alapvető adatbázis-műveletek kezeléséhez.

A ***Rails*** arról is kezdi meggyőzni a fejlesztőket, hogy a megszokások útját állhatják a gyors, hibamentes fejlesztésnek. Évekbe telt, mire a programozók egyetértettek abban, hogy a szemétgyűjtéses nyelvek előrelépést jelentenek a `malloc()`-hoz képest, de szintén sokáig tartott, mire rájöttünk, a szabványok jobbak a beállításfájloknál.

De a ***Rails*** népszerűsége azt jelzi, hogy lassan megérünk egy ilyen változásra. Bár egyetlen webfejlesztői keretrendszer sem tökéletes, szerintem a ***Rails*** megüti a mércét a legtöbb, az évtizedben megírt alkalmazásomhoz. Mind a ***Ruby*** (a nyelv), mind a ***Rails*** (a keretrendszer) fejlődésben van, de ha ilyen, viszonylag éretlen állapotukban így működnek, milyenek lesznek majd készen? – Alig győzöm kivárni.

Linux Journal 2006., 142. szám

Reuven M. Lerner

Forrás:
➲ ***www.linuxjournal.com/article/8693***

Amazon webszolgáltatások

Egyetlen URL-lel webszolgáltatások széles tárházát vehetjük igénybe

Egyetemista koromban nem sok lehetőség volt szakkönyvek beszerzésére. Új példányokat a nagyon drága egyetemi könyvesboltban lehetett megvenni vagy a nagyon drága konkurenciánál, vagy használtakat más diákoktól, akik egy-egy félév végén mindig kiárusítást csináltak. Könyvvásárlási szokásaimat tehát a lakókörnyezetem határozta meg, ráadásul mindig azt tanultam, ami elérhető volt.
Ezek után talán nem meglepő, hogy amint lehetővé vált, beálltam az online könyvesboltok vásárlóinak sorába. Ilyen boltok a ***Bookpool*** és az ***Amazon***, őket legalább 1995 nyara óta „támogatom" vásárlásaimmal. Valóra vált álmok, mert a kitűnő árak mellett széles a választék, s a kényelem sem utolsó szempont. A gond csak az, hogy pénzt még mindig csak annyit költhettem így, mint a hagyományos könyvesboltokban. Szóval a könyvekre költhető anyagi keretem változatlan maradt, az ebből megvehető könyvek száma, valamint a választék az arányokat tekintve igen távol került egymástól.
A helyzet jobbra fordult, mikor az ***Amazon*** megnyitotta kapuit más könyvkereskedők előtt. Nem csak arra nyílt lehetőség, hogy összehasonlítsam az új könyvek árát saját asztalomnál ülve, de böngészhettem régi könyvek között és vásárolhattam is. Nem egészen egy dollárért (plusz postaköltségért) számtalan érdekes könyvből válogathattam, így lassan valamiféle könyvzabáló szörnyeteggé váltam. Könyvespolcomon gyűltek a könyvek, amelyek – reméltem – a kutatásaimhoz segítséget nyújtanak majd, de főleg azért vásároltam őket, mert volt rá lehetőségem. Most is, ha valami jót hallok egy könyvről, első reakcióm az, hogy utánanézek az Amazonon – vagy még jobb, az ***isbn.nu***-n, ahol össze lehet hasonlítani a különböző oldalakon található árakat.
Az ***Amazon*** az évek során hatalmas, temérdek könyv adatait tároló adatbázist épített fel. Biztos vagyok benne, hogy ez a könyv, vásárlói és eladói adatbázis továbbra is nagy segítségére van az ***Amazon*** döntéshozóinak.
Néhány éve azonban az ***Amazon*** egy meglepő dolgot tett: belső adatbázisuk egy részét megnyitották más fejlesztők számára az ***Amazon Web Services (AWS)*** program keretében. Így, az ***AWS***-szel a fejlesztők bármilyen feladatot végrehajthatnak, amit az Amazon oldalon is lehet, csak nem webböngészőt használnak a célra, hanem egy ügyféloldali programot. Az ***AWS*** a könyvkereskedőknek néhány más szolgáltatást is ajánl árkialakítással és beszerzés-ügyintézéssel kapcsolatban.
2005 második felében az Amazon számos olyan új webszolgáltatást vezetett be, amelyek már csak részben köthetők az adásvételhez. Ugyanebben az időben az ***eBay*** bejelentette, hogy nem kér ellenszolgáltatást webszolgáltatásainak használatáért, így két, a világ legnagyobbjai között levő kereskedelmi adatbázis vált elérhetővé. Természetesen a ***Google*** is régóta kínál saját webszolgáltatásokat; és annak ellenére, hogy az adatok jelenleg a főindexre vannak korlátozva, azért kijelenthetjük, hogy remek adatforrás.
Ebben a hónapban felfedező útra indulunk a kereskedelmi webszolgáltatások között, és olyan megoldásokat keresünk, amelyekkel a külső webszolgáltatások adatait saját alkalmazásaink részévé tehetjük. Az út során megismerjük a webszolgáltatások elérésének különböző módjait, a rendelkezésünkre álló lehetőségeket, és azt, hogyan hozhatunk létre meglevő webszolgáltatásokon alapuló új, érdekes alkalmazásokat.

Mit nevezünk webszolgáltatásnak?

A web első évtizedében annak célja főleg a felhasználókkal való interakció volt. Azaz a legtöbb ***HTTP*** ügyfél webböngésző volt, az általuk letöltött tartalom zöme pedig emberi fogyasztásra szánt ***HTML***-formátumú szöveg. Aztán egyszer csak a fejlesztők felismerték a ***HTTP***-ben rejlő lehetőségeket, nevezetesen, hogy az nem csak emberi olvasásra való dokumentumok továbbítására használható. Megkezdődött a ***HTTP*** programok közötti adatátvitelre való használata. A ***HTTP***, mint átviteli protokoll és az ***XML***, mint adatformátum kombinációja az ***XML-RPC***-hez vezetett. Mivel mind az ***XML***, mind a ***HTTP*** platformsemleges, nem volt szükség arra, hogy ugyanazon a nyelven írják meg az ügyfél- és a kiszolgálóoldali programokat, és arra sem, hogy ugyanazt az operációs rendszert használják. Ily módon az ***XML-RPC*** egy platformok közötti ***RPC***-lehetőséget (***RPC: remote procedure call, távoli eljáráshívás***) biztosít, és sokkal kevesebb többletterheléssel jár, mint ugyanezeknek a problémáknak más hasonló megközelítései (mint például a ***CORBA*** közvetítő szoftver).
Az ***XML-RPC*** egy jó, tiszta és könnyűsúlyú protokoll, de sok fejlesztő szerint nem elég kifinomult és vannak hiányosságai a hibakezelés

és az adattípusok terén is. A *SOAP* (a *Simple Object Access Protocol* rövidítése, magyarul egyszerű objektum-hozzáférési protokoll) számos újítást bevezetett, többek között azt, hogy elkülöníti egymástól az üzenet boríték és törzs részét.
Az *XML-RPC*-hez és a *SOAP*-hoz is szükséges egy kiszolgáló, amely megadott *URL*-eken figyel a függvényhívásokra. A kiszolgáló figyelhet mondjuk a */server*, */queries*, vagy valami hasonló *URL*-en. Az ügyfélnek jeleznie kell a kérésben, hogy milyen tagfüggvényt akar meghívni. *XML-RPC*-ben a *methodName* címkét használjuk és egy *HTTP POST* kéréssel elküldött *XML* borítékban adjuk át az összes paramétert és metaadatot.
Egy másik technika, a *REST*, a függvényhívásokat magában az *URL*-ben azonosítja. A paramétereket szabványos *GET* kérésként adja át.
A *REST* számos jó tulajdonsága közül a legfontosabb az egyszerűsége és könnyen használhatósága. Igen könnyű vele a hibakeresés is, mert az *URL*-eket akár egy webböngészőbe is beírhatjuk. Ennek ellenére még mindig sokan használják a *SOAP*-ot és az *XML-RPC*-t, főleg összetett adatszerkezetek kezelésére.
Az webszolgáltatások képezik az alapját azoknak a technológiáknak, melyeket az utóbbi időben *SOA*-ként, azaz szolgáltatás-orientált architektúraként (*Service-Oriented Architecture*) emlegetnek. A webszolgáltatások a web összes erősségét kihasználják, úgymint: platform- és programozási nyelv függetlenség, valamint az a képesség, hogy új változat terjesztése nélkül frissíthetők és változtathatók a szolgáltatások.
A *SOA* lehetővé teszi, hogy új szolgáltatásokat hozzunk létre, vagy akár a már meglévőknek vezessük be új változatait akár egy meglevő megvalósítás helyettesítésével, akár egy, a régivel párhuzamos új változat létrehozásával. A webszolgáltatások felhasználói hasznot húzhatnak a megnövekedett sebességből és hatékonyságból, vagy a teljesen új programozói felületekből anélkül, hogy az inkompatibilitás vagy a telepítés gondot okozna nekik. Ezenkívül amíg a fejlesztők a szolgáltatások közzétett specifikációit követik, addig bármilyen nyelvet és platformot használhatnak, és bármit létrehozhatnak a párbeszédes asztal alkalmazástól kezdve az automatizált kötegelt feladatokig, amelyek több gigabájt adaton rágják át magukat.

Az Amazon webszolgáltatásai

Az *Amazon* az elsők között kezdett webszolgáltatásokat nyújtani. Az *AWS* egy különböző programozói felületekből álló programcsomag, melynek bizonyos részei az Amazon adatbázisához kapcsolódnak, mások, mint a *Mechanical Turk*, vagy az *Amazon's Simple Queue Service* általánosabb célokra valók. A legnépszerűbb szolgáltatás az *E-Commerce Service* (*ECS*, elektronikus kereskedelmi szolgáltatás), amely lehetővé teszi, hogy a termékek adatait kikeressük az *Amazon* különböző depóiból, részletes információkhoz jussunk egy-egy adott termékről vagy terjesztőről, és alapvető e-kereskedelmi műveleteket hajthassunk végre, például, hogy létrehozhassunk kosarat és módosíthassuk annak tartalmát.
Az *ECS*-nek két alapvető működési módja van, a keresés (*search*) és a kikeresés (*lookup*). A keresés bizonyos kritériumoknak megfelelő termékek listáját adja vissza, például *Larry Wall* összes könyvét, vagy olyan könyveket, amelyek címében a *Python* szó szerepel, vagy a *Woody Allen* rendezte filmeket. A kikeresésnek akkor vesszük hasznát, ha ismerjük egy adott termék azonosítóját, az *ASIN*-t (*Amazon Standard ID Number*).
A könyvek *ASIN*-ja megegyezik az *ISBN* azonosítóval (*International Standard Book Number*), más terméktípusok azonosítóját az *Amazon* határozza meg.
Vegyünk egy példát: azt szeretném megtudni, hogy megtalálható-e az *Amazonnál* a *Pragmatic Programmers' book* című könyv a *Ruby on Rails*-hez, és jó lenne tudni az árát is. Mivel egy bizonyos dolgot keresek, érdemes az *ItemLookup* (elem kikeresése) műveletet igénybe venni. Ez azt jeleni, hogy tudnom kell a könyv *ISBN*-jét, ami pedig *097669400X*. (Az *ISBN*-t kötőjelek és más írásjelek nélkül kell megadni az *ECS*-nek.) Végül szükség van egy *AccessKeyId*-re (hozzáférésikulcs-azonosító) értékére, ami egy azonosítószám: azt jelzi az Amazonnak, melyik fejlesztő próbál hozzáférni a rendszerhez. (Az *AccessKeyId* könnyen beszerezhető és ingyenes, a részletekért lásd az online forrásokat.)
Az *ECS REST* kérések alap *URL*-je ➲ *http://webservices.amazon.com/onca/xml?Service=AWSECommerceService*.
A művelet, az *AccessKeyId* és az *ItemId* (elem azonosító) megadásához név-érték párokat adunk az URL-hez. A név=érték formátumot használjuk, a párokat & jellel választjuk el egymástól. Kombinált URL-ünk most így fest:

```
http://webservices.amazon.com/
➥onca/xml?Service=AWSECommerce
➥Service&Operation=ItemLookup&
➥AWSAccessKeyId=XXX&ItemId=
➥0735619530
```

Ezt egy webböngészőbe bemásolva (az XXX helyére egy aktuális *AccessKeyId* értéket kell beírni) egy, az Amazon kiszolgálójáról érkező *XML* dokumentumot kell kapnunk (*text/xml* tartalomtípussal). A dokumentum egy *ItemLookupResponse* címkével kezdődik, majd két részre válik, egy *OperationRequest*-re és egy *Items*-re. Az előbbi leírja az általunk küldött kérést, tartalmazza böngészőnk *UserAgent* fejlécét, valamint az összes, a szolgáltatásnak átadott argumentumot. Az *Items* az *Amazon* válaszait tartalmazza.
Példaképpen bemutatok egy választ, amit én kaptam egy kérésemre az Amazontól:

```
<ItemLookupResponse>
     <OperationRequest>
     <HTTPHeaders>
          <Header Name=
          ➥"UserAgent"
          ➥Value="Mozilla/5.0
          ➥(Macintosh; U; PPC
Mac OS X Mach-O; en-US; rv:1.8)
Gecko/20051111 Firefox/1.5"/>
     </HTTPHeaders>
     <RequestId>1NBTWT1FHDEH
     ➥JK2G16CT</RequestId>
     <Arguments>
          <Argument Name=
          ➥"Operation" Value=
          ➥"ItemLookup"/>
          <Argument Name=
          ➥"Service" Value="AWSE
          ➥CommerceService"/>
          <Argument Name="AWS
          ➥AccessKeyId" Value=
          ➥"XXX"/>
```

```
        <Argument Name="ItemId"
Value="097669400X"/>
    </Arguments>
    <RequestProcessingTime>
    ➥0.00745105743408203</
    ➥RequestProcessingTime>
    </OperationRequest>
    <Items>
    <Request>
    <IsValid>True</IsValid>
    <ItemLookupRequest>
    <ItemId>097669400X</ItemId>
    </ItemLookupRequest>
    </Request>
    <Item>
        <ASIN>097669400X</ASIN>
        <DetailPageURL>
http://www.amazon.com/exec/
➥obidos/redirect?tag=
ws%26link_code=xm2%26camp=2025%
➥26creative=
165953%26path=http://www.amazon
➥.com/gp/
redirect.html%253fASIN=09766940
➥0X%2526tag=
ws%2526lcode=xm2%2526cID=2025%2
➥526ccmID=
165953%2526location=/o/ASIN/
097669400X%25253FSubscriptionId
➥=XXX
        </DetailPageURL>
        <ItemAttributes>
        <Author>Dave Thomas</
        ➥Author>
        <Author>David Hansson</
        ➥Author>
        <Author>Leon Breedt</
        ➥Author>
        <Author>Mike Clark</
        ➥Author>
        <Author>Thomas Fuchs</
        ➥Author>
        <Author>Andrea Schwarz</
        ➥Author>
        <ProductGroup>Book</
        ➥ProductGroup>
        <Title>
        Agile Web Development
        ➥with Rails (The
        ➥Facets of Ruby
        ➥Series)
        </Title>
        </ItemAttributes>
    </Item>
    </Items>
</ItemLookupResponse>
```

Sok hasznos mező található ebben az *XML*-ben. Megtudhatjuk belőle, mennyi időt vett igénybe az Amazonnak, hogy feldolgozza a kérést (ez esetben 0,008 másodpercet) – ez az információ jól jön, ha alkalmazásunkban hibakeresést végzünk és/vagy mérjük a teljesítményét. A `DetailPageURL` tartalmazza azt az *URL*-t, ahova azokat a felhasználóinkat irányíthatjuk, akiknek több információra van szükségük. Olyan adatokat is megkapunk, mint a könyvcím és a szerző(k) neve, így a könyvinformációk remekül megjeleníthetők. Valóban nem okozhat nehézséget, hogy elemezzük ezt az *XML*-t, részleteit vagy az egészet megjelenítsük egy webes, grafikus, vagy parancssori alkalmazásban. Azt is megtehetjük, hogy az adatok egy részét hozzáadjuk egy nagyobb adatbázis-alkalmazásunkhoz, de győződjünk meg róla, hogy ezzel nem sértjük meg az *Amazon* adatvédelmi korlátozásait.

Válaszcsoportok

Legyenek bármilyen hasznosak ezek a fenti információk, eredeti kérdésünkre még mindig nem tudjuk a választ, miszerint kapható-e az *Amazonnál* a *Pragmatic Programmers'* könyv a *Ruby on Rails*-hez, és mennyibe kerül. Azt tudjuk, hogy kapható, de nem tudjuk, mennyiért. Ez azért van, mert az *ECS* alapértelmezetten csak kevés adatot ad vissza, mint az fent is látható. Válaszcsoportok meghatározásával azonban testre szabhatók az *Amazon* válaszai. Minden egyes válaszcsoport megfelel egy vagy több, az ECS válaszában visszaadott adattípusnak. Ahhoz, hogy alapvető árinformációkat szerezzünk egy könyvről, jelezzük, hogy az *OfferSummary* válaszcsoportot szeretnénk megkapni:

```
http://webservices.amazon.com/
➥onca/xml?Service=AWSECommerce
➥Service&Operation=ItemLookup&
➥AWSAccessKeyId=XXX&ItemId=073
➥5619530&ResponseGroup=Offer
➥Summary"
```

Az előző, magát a könyvet leíró lista helyett most egy olyat kapuk, amelyben egy adott könyv legalacsonyabb új és régi árai szerepelnek. A fenti lekérésre az *XML* válasz a következő:

```
<ItemLookupResponse>
<OperationRequest>
<HTTPHeaders>
<Header Name="UserAgent"
➥Value="Mozilla/5.0
➥(Macintosh; U; PPC Mac
OS X Mach-O; en-US; rv:1.8)
➥Gecko/20051111 Firefox/1.5"/>
</HTTPHeaders>
<RequestId>0SNXJ8T5V2JA18M8AJQC
</RequestId>
<Arguments>
<Argument Name="ResponseGroup"
➥Value="OfferSummary"/>
<Argument Name="Operation"
➥Value="ItemLookup"/>
<Argument Name="Service"
➥Value="AWSECommerceService"/>
<Argument Name="AWSAccessKeyId"
➥Value="XXX"/>
<Argument Name="ItemId"
➥Value="097669400X"/>
</Arguments>
<RequestProcessingTime>0.033176
➥8989562988</RequestProcessing
➥Time>
</OperationRequest>
<Items>
        <Request>
        <IsValid>True</IsValid>
        <ItemLookupRequest>
        <ItemId>097669400X</
        ➥ItemId>
        <ResponseGroup>Offer
        ➥Summary</Response
        ➥Group>
        </ItemLookupRequest>
        </Request>
        <Item>
        <ASIN>097669400X</ASIN>
        <OfferSummary>
        <LowestNewPrice>
        <Amount>2295</Amount>
        <CurrencyCode>USD</
        ➥CurrencyCode>
        <FormattedPrice>
        ➥$22.95</Formatted
        ➥Price>
        </LowestNewPrice>
        <LowestUsedPrice>
        <Amount>2341</Amount>
        <CurrencyCode>USD</
        ➥CurrencyCode>
        <FormattedPrice>
        ➥$23.41</Formatted
        ➥Price>
        </LowestUsedPrice>
        <LowestCollectible
        ➥Price>
        <Amount>3495</Amount>
        <CurrencyCode>USD</
        ➥CurrencyCode>
        <FormattedPrice>
```

```
            ➥$34.95</Formatted
            ➥Price>
            </LowestCollectible
            ➥Price>
            <TotalNew>41</TotalNew>
            <TotalUsed>12</
            ➥TotalUsed>
            <TotalCollectible>2</
            ➥TotalCollectible>
            <TotalRefurbished>0</
            ➥TotalRefurbished>
            </OfferSummary>
            </Item>
</Items>
</ItemLookupResponse>
```

Látható, a válasz eleje ugyanaz, mint az előbb volt. A második fele az `<Items>` címkék között azonban más, most `LowestNewPrice`, `LowestUsedPrice` és `LowestCollectiblePrice` címkék mutatják mennyiért vehetjük meg a könyvet.

Más válaszcsoportokat is választhatunk, szükség szerint kombinálhatjuk és összeilleszthetjük őket. Kérhetjük például a *Medium* válaszcsoportot, amely nem csak a kérésről és a könyvről ad információt, hanem a könyvvel kapcsolatos képeket is mutat (több méretben), megtudjuk a könyv méretét és súlyát, valamint elolvashatjuk a szerkesztői ismertetőt. Ha ennél is többet szeretnénk, például más Amazon-vásárlók által írt véleményt a könyvről, vagy a hasonló termékek listáját, kérjük a *Large* válaszcsoportot.

Összefoglalás

Az *Amazon* webszolgáltatások egy olyan eszközt bocsátanak rendelkezésünkre, amellyel termékinformációkat szerezhetünk egy hatalmas adatbázisban, akár személyes, akár kereskedelmi célra. Továbbá az *ECS* ízelítőt ad abból, hogyan hozhatók létre *REST*-szerű kérések, és az eredmények hogyan elemezhetők. Valamint csakúgy, ahogy a webfejlesztők is sokat tanulhatnak a *HTML*-ről és a *JavaScriptről* a meglevő oldalak alapján, mi is elleshetjük a webszolgáltatások létrehozásának titkait, csak tanulmányoznunk kell, az *Amazon* hogyan oldotta meg a kérdést. Én magam az *Amazon* válaszcsoport-megvalósítását kedvelem különösen, amely lehetővé teszi, hogy kombináljuk és összeillesszük a várt választípusokat, nos, ez olyan dolog, amit én is szeretnék a saját webszolgáltatásomba integrálni.

A következő hónapban meg is csináljuk azt, amiről most szó volt, azaz építünk egy saját webszolgáltatást, amely az Amazon, valamint egy helyi nyilvános könyvtár adatait használja fel egy személyre szabott könyvkereső rendszer létrehozására.

Linux Journal 2006., 143. szám

Reuven M. Lerner hosszú ideje web- és adatbázis-tanácsadóként dolgozik, emellett PhD-s oktatástudomány (Learning Sciences) hallgató a chicagói Northwestern Universityn. Chicago külvárosában él feleségével és három gyermekével, köztük Amotz David névre hallgató újszülött kisfiával.

A CIKK FORRÁSAI

➲ www.linuxjournal.com/article/8748

Planet Me blog aggregátor létrehozása

Bemutatjuk, hogyan hozhatunk létre saját blog aggregátort (hírolvasót) a „Planet” oldalakat (például a Planet Apache-ot) működtető kóddal.

A *Planet Projecttel* az online közösségek összegyűjthetik a közösség tagjainak blogjait egy központi weboldalon. A *Planet GNOME* és a *Planet Apache* közösségi blogok is a *Planet* kódon alapulnak például. A *Planet* kód ilyen online felhasználása nem kerül sokba, viszont az emberek szemmel tarthatják a közösséget. A cikkben bemutatjuk, hogyan hozható létre a *Planet* kóddal saját személyes blog hírolvasó gépünkön.

Telepítés

A *Planet* kódhoz a *Python 2.2*-re vagy későbbi változatára van szükség. A *Planet* telepítése úgy a legegyszerűbb, ha letöltünk a ➲ *planetplanet.org*-ról egy előző esti „pillanatfelvétel” (snapshot) tarballt és kicsomagoljuk saját könyvtárunkban. Én általában átnevezem a kitömörített könyvtárat, hogy szerepeljen a nevében a letöltés dátuma, és létrehozok rá egy hivatkozást, mint az aktuális *Planet Me* változatra.

Néhányszor saját könyvtáram elérési útvonalára hivatkozom a cikkben, ne felejtsük el azt saját adatainkkal helyettesíteni.

Az *1. Lista* két utolsó parancsa minta a hírcsatornák lehívására és kezdő *Planetünk* beállítására. Az utasítások változhatnak aszerint például, hogy kell-e proxy szervert használnunk az internet-hozzáféréshez. A parancsok futtatása után kell találnunk egy *Planet Me* nézetet böngészőnkben a *~/planet/me/index.html* címen. E lépések végrehajtása után *Planetünknek* az *1. ábrához* kell hasonlítania.

Nyilván be akarjuk állítani a megtekintendő hírcsatornákat. Ezt a *me-meta/config.ini* végén tehetjük meg. A beállításfájl szögletes zárójelek közé zárt szöveggel jelöli a szakaszokat. Egy-egy szakasz meghatározását *key=value (kulcs=érték)* párokként megadott opciók követik. Minden figyelendő bloghoz új szakaszt kell létrehoznunk, melynek neve az *RSS* csatorna *URL*-je. A *2. Listában* látható példa az alapértelmezett *config.ini* fájl egy részletét mutatja.

A fejlécben látszódik majd a blog neve, ahonnan a bejegyzések származnak, az arckép pedig a jobb oldalon jelenik meg, ha az alapértelmezett *HTML* sablonokat használjuk. Az arcszélesség és az arcmagasság alapértelmezés szerint szabadon beállítható. *Planet Me* oldalunk témaköreinek (*topic*) csinosítására felhasználható ikonokat számtalan oldalon találunk. A *3. Listában* például az egyik *Slashdot* szekció ikont használom (lásd az online forrásokat) a *Slashdot RSS* híreihez.

Ha a *Planet* beállításokat a cikk alapján csináljuk, a témaikonok helye a *~/planet/me/images* lesz. Saját *Slashdot* témaikon beállításom a *3. Listában* látható.

1. Lista A Planet telepítése

```
$ cd ~
$ tar xjvf planet-
➥nightly.tar.bz2
$ planetdated=planet-$(date
➥+'%d%b%y')
$ mv planet-nightly
➥$planetdated;
$ ln -s $planetdated planet
$ cd planet
$ cp -av fancy-examples
➥me-meta
$ cd me-meta
$ cp ../examples/*.xml* .
$ edit config.ini
name = Planet Me
link = file://home/ben/planet/
➥me/index.html
owner_name = John Doe
owner_email = root@localhost
# lejjebb a fájlban
# a sablonfájloknak egy sorban
# kell lenniük
template_files = me-meta/
➥index.html.tmpl
me-meta/rss20.xml.tmpl
➥me-meta/rss10.xml.tmpl
me-meta/opml.xml.tmpl
➥me-meta/foafroll.xml.tmpl
# lejjebb a fájlban módosítjuk
# fancy-examples/
# index.html.tmpl
[me-meta/index.html.tmpl]
items_per_page = 30
$ cd ..
$ mkdir cache
$ ln -s output me
# proxy nélkül
$ python planet.py
➥me-meta/config.ini
# szabványos squid proxyval
# a "dairiserver" gazdagépen
$ http_proxy=http://
➥dairiserver:3128/ \
  python planet.py
  ➥me-meta/config.ini
```

2. Lista Példa hírolvasó meghatározására

```
[http://www.gnome.org/~jdub/
➥blog/?flav=rss]
name = Jeff Waugh
face = jdub.png
facewidth = 70
faceheight = 74
```

3. Lista
Így szerzünk képeket a Slashdotról

```
$ cd ~/planet/me/images/
$ wget  \
http://images.slashdot.org/
➥topics/topicslashback.gif
# az ImageMagick convert
# parancsával alakítsuk át
$ convert topicslashback.gif
➥slashdot.png
```

1. ábra A Planet telepítése

4. Lista A Slashdot ikon haszálatának meghatározása

```
$ edit ~/planet/me-meta/
➥config.ini
[http://rss.slashdot.org/
➥Slashdot/slashdot]
name = Slashdot
face = slashdot.png
$ cd ~/planet
$ python planet.py
➥me-meta/config.ini
```

5. Lista Cron feladat a blogok összegyűjtéséhez

```
$ mkdir -p ~/mycron
$ cd ~/mycron
$ vi upd-planet.sh
#!/bin/sh
cd ~/planet;
http_proxy=http://dairiserver:
➥3128/ \
  python planet.py
  ➥me-meta/config.ini
$ chmod +x upd-planet.sh
$ echo \
'00 04  *  *  * /home/ben/
➥mycron/upd-planet.sh' \
 >|upd-planet.cron
# csak ha a cront a ~/mycron
# könyvtáron kívülről
# használjuk
$ crontab -l >|oldcrontab.cron
$ cat *.cron >|newtab
$ crontab newtab
$ rm -f oldcrontab.cron
```

A *4. Lista* a ***config.ini***-hez csatolandó új szakasz, amely beépíti a ***Slashdot*** ikont saját ***Planet Me***-nkbe.

Dinamikus tartalom

A blogok összegyűjtéséhez és a begyűjtendő blogok listájának egyszerű módosításához most futtatnunk kell a ***Planet*** kódot.
Az ütemezett begyűjtéshez a ***cron***-t használhatjuk. Az *5. Listából* megtudjuk, hogyan frissíthetjük a ***Planet Me***-t napi (éjjeli) rendszerességgel.

Könnyedén felvehetünk és eltávolíthatunk blogokat, ha egy blog-meghatározás fájl listát használunk magának a beállításfájlnak kézzel történő módosítása helyett. Erre a célra használható a *6. Listában* szereplő ***generate-config*** (beállítás-készítő) szkript, amely a blognevet és az ***URL***-eket egyszerű fájlokba helyezi a blog alkönyvtárban. Beállításfájlokat parancssorban vagy valamely fájlkezelőben is felvehetünk és eltávolíthatunk, de használhatunk akár egy ***Firefox*** kiterjesztést is, amellyel menüből adhatunk új RSS csatornákat a ***Planet Me***-hez. Az archívumok kezelése (ezt később bemutatjuk) is egyszerűsödik, ha a bloginformációkat ***config.ini***-n kívül tároljuk.

A kinézet módosítása

Két fájl szabályozza a ***Planet*** kinézetét: a ***me-meta/index.html.tmpl*** és a ***me/planet.css***. Az előbbi az oldaltartalom sablonja, utóbbi pedig a ***CSS*** stíluslap.

6. Lista Az összegyűjtendő blogokat meghatározó fájlok létrehozása

```
$ cd ~/planet/me-meta
$ mv config.ini
➥config.ini.template
$ edit config.ini.template
# töröljünk minden blog URL
# részt a fájl végéről
# keressünk a http: -ra
$ mkdir blogs
$ echo http://
➥rss.slashdot.org/Slashdot/
➥slashdot \
  >blogs/slashdot.blog
$ ./generate-config
```

7. Lista Fájlok használata beállítások létrehozására

```
#!/bin/sh
cp -av config.ini.template
➥config.ini
for if in blogs/*.blog
do
        base=$(basename
        ➥$if .blog);
        content=$(cat $if);
        echo "" >>
        ➥config.ini
        echo "[$content]"
        ➥>> config.ini
        echo "name = $base"
        ➥>> config.ini
        echo "face =
➥$base.png" >> config.ini
done
```

Alapértelemezésként az arc, a bejegyzés, a dátum és az oldalsáv meghatároz stílusokat, melyek a stíluslappal testre szabhatók. Saját betűtípusokat is használhatunk, mindössze a ***font-family CSS***-címkét kell módosítanunk. Az ***index.html.tmpl*** sablon további címkéket tartalmaz, melyeket a ***Planet*** kód a végleges ***index.html*** file létrehozásához használ. A legfontosabb címkék a `TMPL_LOOP`, a `TMPL_IF` és a `TMPL_VAR`. A hírcsatornákat a `<TMPL_LOOP Items>` ***HTML***-szerű címkével és a megfelelő zárócímkével

8. Lista Feltétel vizsgálata a TMPL_IF címkével a megjelenítéshez

```
<TMPL_IF title>
<a href="<TMPL_VAR link
➥ESCAPE="HTML">">
<TMPL_VAR title>
</a>
</TMPL_IF>
```

helyezzük a kimeneti oldalra. Ezen címkék közötti ***HTML*** elemek megjelennek minden megjelenítendő bejegyzéssel. Ezek határozzák meg, mi és hogyan jelenjen meg az egyes bejegyzésekkel.
Ezzel a változók jelölik a hírcsatornák tartalmának helyét. A ***Planet*** kód a `<TMPL_VAR title>` címkét például az aktuális hírelem címével helyettesíti. A `TMPL_VAR`-nak nincs zárócímkéje. A `TMPL_IF` címke bizonyos feltételek és adatok meglétét ellenőrzi. Például néha a híreknek nincs címük.
A *8. Listában* szereplő kód kiírja a címadatokat, ha vannak, és nem ír ki semmit, ha nincsenek. A `TMPL_VAR` címke `escape` attribútuma miatt a ***Planet*** ellenőrzi, hogy a hivatkozásváltozó értéke megengedett ***HTML*** attribútum formátumú-e.
A ***me-meta/index.html.tmpl***-t és a ***CSS*** fájlokat kell szerkesztenünk a csatornaikonnak a hírelemek bal szélére való mozgatáshoz. Alapértelmezésként az ***index.html.tmpl*** csak akkor jeleníti meg a csatornaikont, ha az aktuális hírelem más csatornáról származik, mint az azt megelőző.
Az ***index.html.tmpl*** arcképet megjelenítő része körül töröltem a `<TMPL_IF new_channel>` címkéket, ahogy azt a *9. Listában* látható részlet is mutatja. A csatorna kép ***CSS*** osztályát `news-item-icon`-ként, a hír fő részét `news-item`-ként, az aktuális csatornához tartozó képét pedig `embedded-face`-ként határoztam meg.
A *10. Listában* található új stíluslap kóddal beállítjuk a csatorna képet a hírelem bal oldalán. ***Planet Me***nknek most a *2. ábrához* kell hasonlítania. Ha a jobb oldalra szeretnénk tenni az ikonokat, változtassuk

9. Lista Új csatorna elemek szakasz a ~/planet/me-meta/index.html.tmpl-hez

```
<TMPL_LOOP Items>
<TMPL_IF new_date>
<h2><TMPL_VAR new_date>
➥</h2>
</TMPL_IF>
<div class="news-item-
➥icon">
 <a href="<TMPL_VAR
 ➥channel_link ESCAPE=
 ➥"HTML">"
    title="<TMPL_VAR
    ➥channel_title
ESCAPE="HTML">">
 <TMPL_VAR channel_name>
 <br/>
 <img class="embedded-face"
 ➥alt=""
src="images/<TMPL_VAR
➥channel_face ESCAPE=
➥"HTML">" >
 </a>
</div>
<div class="news-item">
 <TMPL_IF title>
 <h4><a href="<TMPL_VAR link
 ➥ESCAPE="HTML">">
        <TMPL_VAR title>
        ➥</a></h4>
 </TMPL_IF>
 <div class="entry">
 <p>
 <TMPL_VAR content>
 </p>
 <p class="date">
 <a href="<TMPL_VAR link
 ➥ESCAPE="HTML">">
<TMPL_IF creator>by <TMPL_VAR
➥creator> at </TMPL_IF>
<TMPL_VAR date></a>
 </p>
 </div>
</div>
</TMPL_LOOP>
```

meg úgy a stíluslapot, hogy a `news-item-icon float` címkéje `right` legyen, a hírelem `margin-left`-je pedig `0px`.
A csatorna meghatározás fájlban használt `face=akármi.png` sor nem feltétlenül szükséges. Bármilyen más változót megadhatunk az egyes

2. ábra Saját Planet Me oldalam testreszabott sablonnal és CSS fájllal

10. Lista A ~/planet/me/planet.css-hez adandó új stílusok

```
div.news-item-icon {
      float: left;
      position: relative;
      left: 4px;
      margin-top: 25px;
      padding: 0 20px 30px 0;
      width: 120px;
      text-align: center;
}
div.news-item-icon a {
      text-decoration: none;
}
div.news-item {
         margin-left: 140px;
}
```

csatornákhoz, s azok az *index.html.tmpl*-ben elérhetők lesznek. A *11. Listában* például a `foo` változó szerepel, melyet talán egy csatorna `foo=bar`-jaként adunk meg a csatorna leírása után a *config.ini* fájlban.

11. Lista A TMPL_IF-fet így is használhatjuk

```
<TMPL_IF channel_foo>
Have foo:<TMPL_VAR
➥channel_foo ESCAPE="HTML">
</TMPL_IF>
```

A *Planet Me* testreszabását más *Planet* weboldalak tanulmányozásával is elsajátíthatjuk: *HTML* és *CSS* fájljaikból megtudhatjuk, hogy ők hogyan módosították a kinézetet.

Archívumok tárolása és megtekintése

A *Planet* képes sok forrásból hírcsatornákat összegyűjteni és azokat visszamenőlegesen megjeleníteni egy oldalon. *Planet Me*-vel való helyi használat esetén egy hírcsatorna egy korábbi időintervallumban megjelent hírei is megtekinthetők.
A *Planet Me* létrehoz egy érvényes *RSS RDF* hírcsatornát, melyet felhasz-

12. Lista Hírcsatornák ismételt archiválása

```
$ cd ~
$ unzip Jena-2.3.zip
$ edit ~/.bashrc
# a classpath változó
# beállítása
JenaSetup() {
    for if in ~/Jena-2.3/
    ➥lib/*.jar; do
        export CLASSPATH=
        ➥$CLASSPATH:$if;
    done
}
$ . ~/.bashrc
$ JenaSetup
# hírcsatorna archiválása
# három ismételhető lépés
$ cd ~/planet/me
$ mv -f archive.xml
➥rss10-archive.xml
$ java jena.rdfcat rss10*xml
➥>archive.xml
```

13. Lista
Keresés Planet Me oldalunkon Jenával

```
$ cat rss-by-date.sparql
PREFIX dc:
➥<http://purl.org/dc/
➥elements/1.1/>
PREFIX xsd:
➥<http://www.w3.org/2001/
➥XMLSchema#>
DESCRIBE ?channel ?bnode
➥?a WHERE
{
 ?channel ?items ?bnode .
 ?bnode ?hasitem ?a .
 ?a dc:date ?date .
 FILTER ( xsd:dateTime(?date)
    >= xsd:dateTime
    ➥("2006-01-03T00:00:00")
 && xsd:dateTime(?date)
    <= xsd:dateTime
    ➥("2006-01-05T00:00:
    ➥00") )
}
$ cd ~/planet/me
$ java jena.sparql -data
➥archive.xml \
 -query rss-by-date.sparql
 ➥-results RDF/XML \
 >my-query-result.rss
```

nálhatunk ***Planetünk*** archiválására. Az ***RDF*** fájlok tartalma hármas csoportokat képez. A hármas csoportok tagjaira mint alanyra, állítmányra és tárgyra hivatkozunk. Egy hármas jelentheti azt, hogy a hírelemnek van egy megjelenési dátuma, például: `item57 has-date 3-Jan-2006`. Egy ***RSS*** hírcsatorna meghatároz egy hírcsatornát, hozzárendel híreket, minden hírhez pedig olyan tulajdonságokat, mint a címe, közreadásának időpontja és szöveges tartalma. A `has-date`-et és hasonlókat hosszú ***URI***-k írják le, hogy ne lehessen két hármasnak azonos karakterlánc értéke.
Könnyen, de egyszersmind hatékonyan archiválhatjuk ***Planetünk RSS***-ét a ***Jena Projecttel***. Ha van telepítve Java virtuális gépünk, akkor a ***Jena*** telepítéséhez csak egy tarballt kell letöltenünk, kitömörítenünk és a ***classpath***-hoz (osztályútvonal) adnunk.
A *12. Listában* a telepítés lépései és a hírcsatornák archiválásának

14. Lista
Jena lekérés eredményekkel módosíthatjuk Planet Me hírolvasónkat

```
$ cd ~/planet/me-meta
$ cp -av config.ini.template
➥config.ini
$ echo \
"[file:///home/ben/planet/
➥me/my-query-result.rss]"
➥ \
  >>config.ini
$ echo "name = archive"
➥>>config.ini
$ cd ~/planet
$ rm -f cache/file.home*
$ python planet.py
➥me-meta/config.ini
```

ismétlődő folyamata látható. Saját hírcsatornánk archívumát ***Jenával*** egy adatbázisba helyezhetjük, ha hosszú idő alatt tetemes mennyiségű csatornát halmoznánk fel.
Jenával igen hatékony lekérdezéseket hajthatunk végre az archívumunkban, és könnyen újraépíthetjük a ***Planetünket***.
A *13. Listában* egy egyszerű, időintervallum alapú hírcsatorna-lekérés látható. A lekérés ***SPARQL*** lekérdezőnyelven íródott, ez használatos ***RDF*** tárolókból való lekérésekhez.
A lekérés hivatkozik a csatornára, a hír- és a dátumelemre, mielőtt egy szűrőt alkalmazna arra, hogy a hírelem dátuma alapján melyik hírelemet kell visszaadnia.
Most már könnyedén megváltoztathatjuk ***Planet Me*** oldalunkat, hogy bemenetként csak a saját lekéréseinkből vett eredményeket használja.
Lásd a *14. Listát*: a blog ***URL***-jeit és metaadatait a fenti leírás alapján külön fájlokba helyeztük.
A fenti lekéréssel a csatornaikon ugyanaz marad, mert egyetlen hírcsatornát kérdezünk le: a sajátunkat. A `regex()` egy másik kifejezés, amelyet a `FILTER` szakaszban használhatunk. A *15. Lista* kódja az összes hírelemet szűri, és csak azokat mutatja meg, amelyek megfelelnek a kis- és nagybetűket meg nem különböztető szabályos kifejezésnek.

15. Lista
Szabályos kifejezéseket is használhatunk szűrőként

```
PREFIX dc:
➥<http://purl.org/dc/
➥elements/1.1/>
PREFIX xsd:
➥<http://www.w3.org/2001/
➥XMLSchema#>
PREFIX rss:
➥<http://purl.org/rss/1.0/>
PREFIX content:
➥<http://purl.org/rss/1.0/
➥modules/content/>
DESCRIBE ?channel ?bnode ?a
➥WHERE
{
 ?channel ?items ?bnode .
 ?bnode ?hasitem ?a .
 ?a content:encoded ?
 ➥content .
 FILTER ( regex(?content,
 ➥".*product.*", "i") )
}
```

Összegzés

A ***Planet Me*** online közösségi blog aggregátornak készült (tehát eredetileg nagy látogatottsághoz tervezték), de a ***Planet*** kód alapján igen hatékony blog hírolvasót hozhatunk létre személyes használatra is. Némi munkát igényel ugyan, de megéri, hiszen az eredmény: egy effektív, személyes blog hírolvasó, ahol a közösséget mi magunk alakíthatjuk, valamint archiválhatunk és egy hathatós lekérdezőnyelv segítségével kereshetünk is a régi hírek között.

Linux Journal 2006., 144. szám

Ben Martin

Virtuális fájlrendszerek (libferris) létrehozásával és a bennük való adatbányászattal tölti szinte minden idejét. Most e fájlrendszerek kiterjesztésén dolgozik, hogy az Emacs és Firefox fájlrendszerekként csatolhatók legyenek.

KAPCSOLÓDÓ CÍMEK

➲ www.linuxjournal.com/article/8830

Programozzunk Pythonban (3. rész)

Feltételes szerkezetek, listák, operátorok

Logika ismét, magasabb szinten

Programozással megoldható feladatokban gyakran „csak” egy kijelentés igazságtartalmát szeretnénk megtudni, s nem érezzük szükségét annak, vagy nem látjuk értelmét, hogy a biteket vizsgáljuk. Ilyenkor jól jönnek a magasabb szintű logikai műveletek. Sok más programozási nyelvhez hasonlóan a Python is azt az elvet követi, hogy a 0 értéket tekinti hamisnak, és minden mást igaznak vesz. Természetesen ehhez előbb saját magának elvonatkoztatva 0-ra és – általában az egyszerűség kedvéért – 1-re kell alakítania a kiértékelni kívánt kifejezést. Például nézzük meg újra a számok párosságát vizsgáló rövid kis programot átírva:

```
a=15
if a%2:    #mivel 15-öt 2-vel
#osztva a maradék 1,
#ez a kifejezés IGAZ
  print "Páratlan"
else :
  print "Páros"
```

Az eredeti példához képest kénytelenek voltunk megfordítani a vizsgálat eredményét kiíró részeket, így viszont megspórolhattuk az összehasonlítást (emlékezzünk rá, hogy eredetileg `if a%2 == 0:` szerepelt). A kiértékelés során az `if a%2:` sort az értelmező igaznak tekinti (mivel a maradék 1, azaz 0-tól különböző), ezért esetünkben a vezérlés az első szöveg kiírásával ér véget. Talán kevésbé nyilvánvaló a példa, ha nem számokat, hanem szöveges tartalmat kell elemezni:

```
elso=''  #üres sztring, két '
#jel között nincs semmi
masodik='Nem vagyok üres'
if elso:
  print "elso: IGAZ"
elif masodik:
  print "elso: HAMIS, masodik:
  ➥IGAZ"
else:
  print "elso: HAMIS, masodik:
  ➥IGAZ"
```

A feltételvizsgálat kissé bonyolultabbnak tűnhet, mint az eddigiek, valójában csak az összevetés további finomításáról van szó. Ha az első feltételnek (`if`) nem felel meg a kifejezés, akkor hajtódik végre az `elif` ág, s ha ebbe a kategóriába sem tartozik, akkor a végső `else` ág. Természetesen az `elif` ágakból lehetne több is, működésük hasonló más programozási nyelvek ***case – switch*** párosához. A példa értelemszerűen azt fogja kiírni, hogy az első vizsgálat hamis eredményt, míg a második igazat hozott, mivel egy üres karakterláncot nézve az értelmező automatikusan 0 logikai értékre alakította át saját maga számára. Lehetőségünk van a már megismert bináris logikai műveletek magasabb szintű (úgynevezett ***Boole algebra*** szerinti) használatára is. Ha két kifejezés közé egy ***and*** műveletet illesztünk, az előző példát egyszerűbben tudjuk leírni:

```
elso=''  #üres sztring, két '
#jel között nincs semmi
masodik='Nem vagyok üres'
if elso and masodik:
  print "IGAZ mindkettő."
else:
  print "HAMIS valamelyik
  ➥feltétel."
```

Egy nagy különbség mindjárt szembeötlik: utóbbi módszerrel csak azt tudtuk megmondani, hogy valamely feltétel hamis volt, azt nem, hogy melyik. Az ***és*** logikai művelet végrehajtása során az értelmező előbb megvizsgálja az első feltételt, s ha azt hamisnak találja, már meg sem nézi a többit, automatikusan hamis értékűnek tekinti az egész kifejezést. Ha az első igaz, a másiktól függ a kimenet, annak igaz volta az egész kifejezést igazzá teszi, hamis érték esetén pedig az egész hamis lesz.
Az `or` nevezetű ***vagy*** művelet működése megegyezik a bináris ***megengedő vagy*** műveletével, azaz csak akkor ad hamis értéket, ha mindkét feltétele hamis, minden egyéb esetben igaz lesz a végeredmény:

```
elso=''  #üres sztring, két '
#jel között nincs semmi
masodik='Nem vagyok üres'
harmadik='' #üres sztring, két '
#jel között nincs semmi
if (elso or masodik) and
➥(elso or harmadik):
  print "Az elso VAGY masodik,
  ➥valamint az elso VAGY
  ➥harmadik művelet
  ➥végeredménye IGAZ"
elif elso or harmadik:
  print "Az elso VAGY harmadik
  ➥művelet végeredménye IGAZ"
else:
  print "Az elso VAGY harmadik
  ➥művelet végeredménye
  ➥biztosan HAMIS, a többit
  ➥nem tudom"
```

A kiértékelés során az első feltétel igaz ugyan (`elso or masodik`), de a feltételvizsgálatba ezúttal két logikai kifejezés együttes vizsgálatát tettük, s a második feltétel (`elso or harmadik`) már hamis értéket eredményez. Ezért a vezérlés a következő ágon

folytatódik,s mivel ott már konkrétan csak az `elso` és a `harmadik` nevezetű változókat kapcsoltuk össze vagy művelettel, ennek eredménye egyértelművé teszi, hogy mindkettő hamis-e (igen, tehát az utolsó sort írja ki a program). Ettől még az első feltételvizsgálat eredményéről nem tudtunk meg többet, azaz az `elso` és `masodik` nevű változó viszonya ebből a programból nem derül ki. Szándékosan hoztam ilyen látszólag esetlen példát, talán ebből jól látható, hogy az egyszerű logikai műveletek során is nagyon figyelnünk kell, mi kerül egyáltalán a vizsgálat látóterébe, különben az információk hiányában rosszul következtethetünk.
A logikai tagadást a `not` szócska jelenti, az igazból hamis, a hamisból igaz értéket képez:

```
a = 0
if not a:
  print "Így már nem nulla
  ➥logikailag, azaz IGAZ
  ➥értéke viszont maradt:" , a
else:
  print "Logikailag is nulla,
  ➥azaz HAMIS értéke pedig:"
  ➥, a
```

Szót kell ejtenünk még a ***nem egyenlő*** jelentéssel bíró `!=` operátorról is, mely a már többször használt == egyenlőségvizsgálat ellentettje.

```
a = 0
b = '0'
if a != b:
  print "Nem egyenlő a(" , a ,
  ➥") és b(" , b , ") értéke"
else:
  print "A két érték egyenlő"
```

Túlterhelt operátorok

A korábbi példákból kitűnik, hogy bár a ***Python*** a szkriptnyelvek többségéhez hasonlóan nem tartozik a szigorúan típusos nyelvek közé, néhány alapvető adattípust meg kell különböztetnünk a hatékony használat érdekében. Van ugyan átjárhatóság a számok és a karakteres adatok között (erre volt jó példa a számnak látszó karaktersorozat felhasználása összeadásban), de ügyelni kell a megfeleltetésekre.
Említettük az általános jellemzésben, hogy a ***Python*** támogatja az objektum-orientált programozást és tervezést, sőt, e szemlélet egyik legfontosabb tulajdonsága e nyelvnek. Lehetséges például az operátor túlterhelés néven ismert jelenség felhasználása programjainkban. Ez alatt azt értjük, hogy egy eredetileg valamilyen célt szolgáló operátort (példánkban a számok összeadását jelképező + jel) teljesen más tulajdonsággal ruházunk fel, jelen esetben karakterláncokat kapcsolunk össze segítségével.
Íme egy példa:

```
a=14
b=27
c='császár'
d='pingvin'
print "Az a változó értéke: ",
➥a, " típusa: " , type(a)
print " A b változó értéke: ",
➥b, "típusa: ", type(b)
print "Az a + b művelet
➥értéke:", a+b, " tehát a +
➥operátor itt összeadást
➥jelentett."
print "A c változó értéke:", c,
➥" típusa: ", type(c)
print "A d változó értéke:", d,
➥" típusa:", type(d)
print "A c + d művelet
➥értéke:", c+d, " tehát a +
➥operátor itt összefűzést
➥jelentett."
```

Talán kevésbé meglepő ezek után a következő példaprogram kimenete:

```
a='császár'
b='pingvin'
c='ek'
if a+b+c != (a+b)+c:
  print "E két művelet
  ➥eredménye nem egyezik"
else:
  print "A " , a+b+c , " a " ,
  ➥a+b ,  " többesszáma, \n\
    csakúgy, mint a " , (a+b)+c
print "Három ilyen állat: " ,
➥3*(a+b) , "\n\
  nem ugyanaz, mint  három
  ➥ilyen állat: " , 3*a+3*b
```

Mit is láthattunk itt? Az operátortúlterhelés eredményeként összefűztünk három sztringváltozót, és az összeadás eredeti, asszocivitás néven ismert csoportosíthatósági tulajdonságával találkoztunk. Ráadásul a számok esetén szorzást jelentő operátort is túlterheltük, és segítségével a karaktersorozatokat sokszoroztuk meg. A szorzásjellel jelölt művelet végrehajtása ugyanúgy korábban történik, mint az összeadásjellel jelölteké (erre a precedenciasorrendre példa az utolsó sor). A karaktersorozatok jellegéből fakadóan azonban nem kommutatív a sokszorozás az összefűzésre nézve, azaz az utolsó két sor eredménye eltérő, noha az eredeti, számokkal végzett művelet esetén nem lenne különbség. (Az összefűzés művelete már önmagában sem felcserélhető, hiszen a végeredményben található karakterek megegyeznének ugyan, de értelmük teljesen más lesz.)
Az operátortúlterheléstől függetlenül újdonságot jelent a több sorba tördelt összetartozó szöveg is, erre lehetőséget a példaprogramban használt backslash biztosít, míg az előtte álló `\n` karakterek azt jelzik, hogy sortörést is kérünk. Ügyeljünk rá, hogy a lezáró ” jelet a következő sorban kell az adott szövegegység végén elhelyezni, különben az értelmező hibát jelez.

Listák, tuple

Az eddig részletezett egyszerű adattípusokon túl lehetőségünk nyílik összetett adattípusokat is igénybe venni. Ezek közül a listával már találkoztunk egy példa erejéig, új alkotása egyszerű:

```
lista=['13' , 43.7 , 1988 ,
➥'kutya' , 'macska']
```

A lista úgynevezett szekvenciális adattípus, sorban haladva érhetők el elemei. Hivatkozni rájuk sorszámuk alapján lehet, 0-tól kezdve a számozást, a lista neve után egy szögletes zárójelbe foglalva Jellemzője továbbá, hogy bővíthető illetve szűkíthető tetszőlegesen, és elemeinek nem kell egyforma típusúaknak lenniük. Például:

```
honap='január'
hideg_honapok=['november'  ,
➥'január' , 'február' ]
if honap in hideg_honapok:
  print honap , "is a hideg
  ➥hónapok között szerepel, \n\
     csakúgy, mint például " ,
     ➥hideg_honapok[0]
```

Ciklusszervezést már láthattunk, az itt használt hasonlít a `for i in range()` típusúakra, viszont itt a feltételt nem a hagyományos értelemben vett ciklusváltozóval állítjuk be, hanem egy tartalmazásra kérdezünk rá. Ha csak végig akarunk menni a lista elemein, a következő ciklust érdemes használnunk:

```
hideg_honapok=['november'  ,
➥'január' , 'február' ]
print "A hideg_honapok lista" ,
➥len(hideg_honapok) , " elemet
➥tartalmaz:"
for elemek in hideg_honapok:
  print elemek
```

Könnyű dolgunk volt, nem kellett foglalkozni a lista kezdő- és végpontjával, a ciklus minden lényeges teendőt ellátott helyettünk. Érdekességképp kiírattuk a lista hosszát a `len()` függvény segítségével, mely nem csak a listák, de egyéb adatszerkezetek esetén is hasznos. Ha saját magunknak kellene algoritmust gyártani egy lista bejárására, körülbelül így nézne ki:

```
lista = ['elem1' , elem2,
➥'elem3', … elemn]
cilusváltozó = 0
listavége = elemszám(lista)
ciklus ciklusváltozó
➥kezdőpontjától amíg
➥ciklusváltozó <  listavége:
  tároló változó = lista
  ➥[változó. eleme]
  kiíratjuk tároló változót
  cilusváltozó =
  ➥ciklusváltozó+1
ciklus vége
```

Próbálkozzunk meg néhány egyszerűbb listamanipulációval:

```
honapok=['november' , 'január' ,
➥'február' ]
honapok[1] = 'december'  #a
#második elem helyére tesszük
```

Konkrét helyet jelöltünk ki az elem számára, és mivel ott már szerepelt egy érték, ezt felülírtuk.

```
utolso_elem=honapok.pop()     #a
#lista legnagyobb indexű eleme
print utolso_elem
```

A lista utolsó elemét a `pop()` függvény segítségével olvastuk ki. Az objektum-orientált szemléletnek köszönhetően a ***Pythonban*** szinte minden – így a lista is – objektum. A használt függvény ennek egy metódusa, azaz hozzá tartozó függvénye, – ezt jelzi a lista neve után tett pont – melynek segítségével a külvilággal kommunikál. A memóriaszervezésben, tárfoglalásban van elsődlegesen szerepe az utoljára elhelyezett adatoknak, ezért is hoztak létre külön függvényt számára. Több olyan metódust is találhatunk, melyek a listákkal való hatékony munkavégzést segítik, ezek felkutatásában segíthet például a parancssorból kiadott `pydoc list` parancs.

```
del honapok[0]    #töröljük
#novembert
mas_honapok = ['március' ,
➥'április' , 'május' ,
➥'június']
honapok.extend(mas_honapok)
   #hozzáadunk egy teljes
#listát
honapok.append('12')   #egy
#elemet adunk hozzá
```

Jól látható a különbség egy elem hozzáadása és egy teljes lista (vagy egyéb szekvenciális adat) hozzáadása esetén: más függvényt használtunk. Törölni már az előzőektől eltérően egy külön utasítással tudtunk, azonban lehetőség van a listához tartozó `remove()` függvény használatára is, de figyeljünk a különböző lehetőségek különböző zárójeleire (`[]` között az elem sorszáma, `()` között a neve):

```
honapok.remove('december')
```

Arra is lehetőségünk van, hogy egyszerre hozzáadjunk valamit az összes elemhez, például írjuk mögéjük a „hónap” szót:

```
honapok = [elemek+' hónap' for
➥elemek in honapok] #bővítés
#minden elemnél
for kiir in honapok:
  print kiir
```

Az eddig használt megoldásoknál életszerűbb a tartományra hivatkozás, melyet a tartományokra vágás műveletével tudunk elérni:

```
honapok[2:]=['új elem']  #a 3.
#elemtől töröl, és ezt írja
#bele egyszer
del honapok[1:3]   #az 1. és a
#2. elemet törli
del honapok[:3]    #a 3. elemig
#töröl, csak a 4.-től maradnak
#meg
```

Komoly odafigyelést igényel eleinte a szeleteléses technika elsajátítása, hiszen figyelni kell a 0-val kezdődő indexszámokra, valamint arra is, hogy a szeletelés által megadott számok nem az elemeket vagy az indexet, hanem az indexszám előtti pozíciót jelölik.

Például:

```
lista = [1, 2, 3, 4, 5, 6, 7, 8, 9]
del lista[3:6]'
print lista
```

A végeredmény [1, 2, 3, 7, 8, 9] lesz, azaz a harmadik indexszámú (4) elem bal oldalától a hatodik indexszámú elem (7) bal oldaláig vágtunk ki a listából.
Ha csak annyit tudunk, hogy egy adott elem szerepel a listában, és például vele együtt a harmadik elemig bezárólag szeretnénk kivágni egy darabot, akkor segíthet az `index()` függvény:

```
del honapok[(honapok.index
➥('március hónap')):3]
```

A listákhoz nagyon hasonló adattípus az úgynevezett ***tuple***, két jelentős különbséget érdemes azonban megemlítenünk: nem változtathatóak az elemei, és nem szögletes zárójelbe kell őket tenni:

```
honapok_tuple = ('január',
➥'február', 'március')
```

Szekvenciális és egyéb adatszerkezetek segítségével már kisebb programokat érdemes írni, melyek nagymértékben megkönnyíthetik az adatfeldolgozáshoz, szövegmanipulációhoz kapcsolódó feladatokat. S mindeközben észrevétlenül ismét közelebb jutottunk az objektum-orientált szemlélethez is.

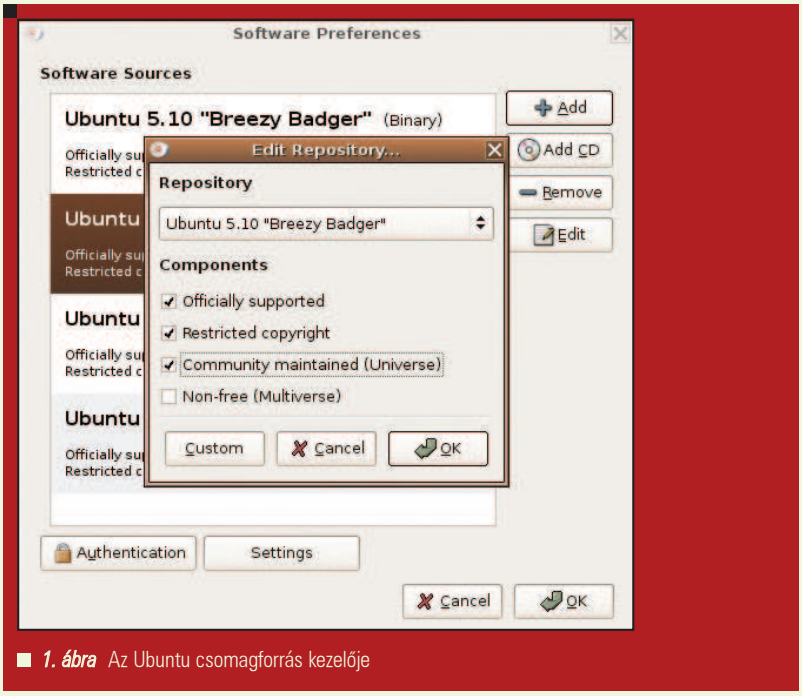
Tóth Virgil Zoltán
(m_v@c2.hu)

Segédeszközök tűzfal-szabályok beállításához

Internetezés közben Linux alatt sem vagyunk biztonságban, ezért érdemes valamilyen tűzfal alkalmazást használni, mely rendszerünk elsődleges védelmi vonala lesz. A számos elérhető konzolos és grafikus alkalmazás közül ebben a cikkben a Firestartert mutatom be.

A *Firestarter* a 2.4-es kerneltől meglévő *Netfilter* állapottartó csomagszűrő rendszerhez készült grafikus segédprogram. Segítségével könnyen és egyszerűen adhatunk meg tűzfal szabályokat, valós időben monitorozhatjuk a különböző hálózati eseményeket, nyithatunk vagy elrejthetünk portokat, megállíthatjuk *DoS* támadásokat.

A program több fajta csomagformátumban elérhető. Én *Ubuntu Linuxra* telepítettem, ahol először a hivatalos mellé fel kellett venni a közösség által karbantartott (*universe*) csomagforrást. Ez legegyszerűbben az *alkalmazások hozzáadása/beállítások/repository* alatt tehető meg. Ezután már csak ki kell adni az `apt-get install firestarter` parancsot. Ha *rpm* alapú csomagkezelővel rendelkező disztribúciónk van(több között *Red Hat, SuSE* vagy *Mandrake Linux*), akkor előbb a ➲ *www.fs-security.com/download.php* címről töltsük le a megfelelő csomagot. Majd rendszergazdaként adjuk ki az



1. ábra Az Ubuntu csomagforrás kezelője

```
rpm -Uvh firestarter*rpm
```

parancsot.

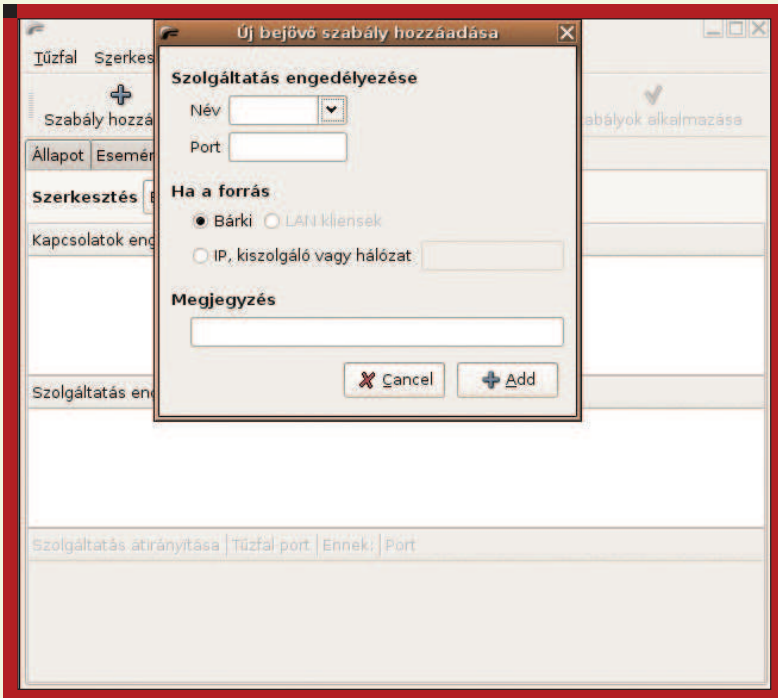
A programhoz csak rendszergazdai jogosultságokkal férhetünk hozzá. (Ha nem találjuk akkor az

Alt-F2 lenyomása után írjuk be, hogy `firestarter`) Első elindításakor egy varázsló segíti a tűzfalunk beállítását (ezeket megváltoztatni később sok más mellett a beállítások menüpont alatt lehet). Az üdvözlő képernyő után ki kell választanunk a használni kívánt hálózati eszközt. Ha modemet használunk vagy kábeles, *DSL* szolgáltatónk *PPPoE* protokollt használja akkor a *pppX*-t, egyébként *ethX*-t érdemes választani. (X=0,1...) Ezen kívül, ha betárcsázós *Internet* elérésünk van akkor választhatjuk, hogy a tűzfal csak a csatlakozás után induljon el. Ezt azonban nem javasolom mivel nem mindegyik tárcsázóval tud együtt működni a program. A következő képernyőn megadhatjuk, hogy ezen gépen keresztül más gép is el fogja érni a hálózatot, hanem akkor ne jelöljük be ezt az opciót. Utolsó lépésként mér csak a mentés gombra kell kattintani, ekkor alapértelmezett szabályokkal elindul a tűzfalunk. Ez azt jelenti,hogy korlátozni fogja a bejövő forgalmat, a kimenőt pedig engedélyezni. Tehát védve vagyunk a külső támadások elől, de korlátozás nélkül használhatjuk többek között a böngészőnket, levelező programunkat.
A megjelenő ablakon három fül található: állapot, események, és a szabályok. Az elsőn információkat kaphatunk a program aktuális állapotáról: le van e tiltva tűzfal, milyen volt a hálózati forgalom aktivitása(elküldött, fogadott *MB*-tok értéke), valamint, hogy mennyi a meggátolt behatolási kísérletek és nem engedélyezett kifelé irányuló próbálkozások száma. Ezekről részletes információt az események fül alatt kaphatunk pl.: mikor történt, iránya (belső hálózatról vagy az Internet felől), mi a forrás *IP* címe, mely szolgáltatást akarták használni, blokkolt csomag nagysága. Ezek nem mindegyike jelenik meg alapból, a hiányzóakat az ***események/oszlop mutatása*** menüpont alatt adhatjuk a többihez. A különböző támadásfajták színekkel is meg vannak különböztetve:

- fekete: Általános csatlakozási kísérlet egy találomra választott *port*-on. Ide kerülnek a lefülelt *port* szkennelési támadások.

- piros: Csatlakozási kísérlet egy nem publikus szolgáltatáshoz. (például: *FTP*-hez).

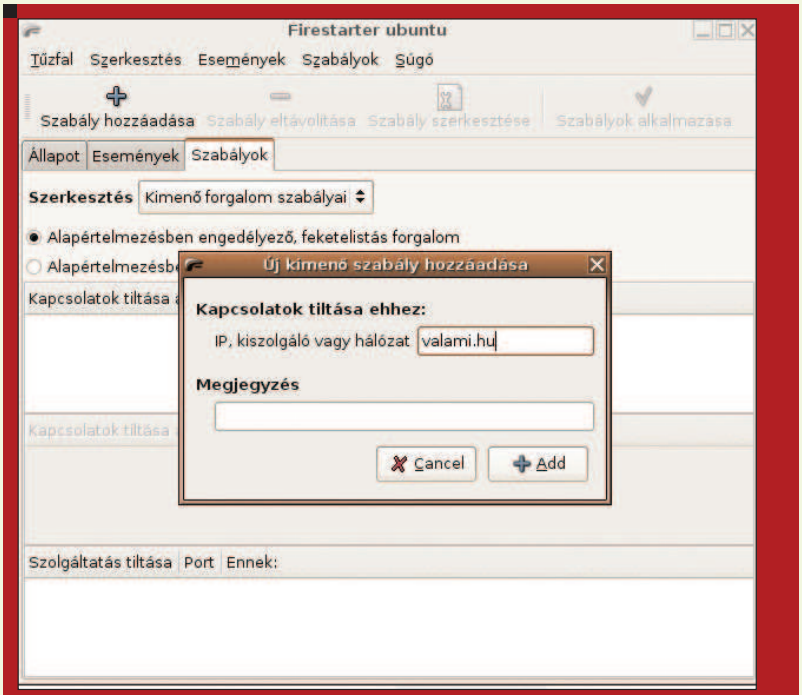
2. ábra A képen a tűzfal le van tiltva rossz hálózati eszköz választás miatt



3. ábra A szolgáltatás elérésnek engedélyezése

- szürke: Ártalmatlan kísérlet. Ebbe a csoportba elsősorban az üzenetszórási forgalom található meg.

Ha a jobb egérgombbal egy sorra kattintunk beállíthatjuk, hogy mit csináljon a program hasonló esemény esetén, tehát már itt megváltoztathatjuk



■ *4. ábra* A valami.hu felkerül a fekete listára

az alapértelmezett tűzfalszabályokat. Az esemény irányától függően a következőeket adhatjuk meg:

1. Ha befelé irányuló (csatlakozási kísérlet az Internet vagy helyi hálózat felől a tűzfalt futtató ***host***-hoz.)
   - A forrás ***IP***-nél megjelenő cím számára mindenfajta csatlakozás engedélyezése.
   - Használni kívánt szolgáltatás engedélyezése mindenkinek illetve csak a forrás címnek. Utóbbi esetben a szolgáltatás rejtve van más ***host***-ok esetében
2. Kifelé irányuló próbálkozás esetén:
   - Csatlakozás engedélyezése a célhoz
   - Adott külső szolgáltatáshoz való csatlakozás engedélyezése a helyi hálózaton adott/mindegyik kliens számára

Az utolsó fülön bejövő és kimenő hálózati forgalomra adhatunk meg szabályokat.
A bejövő forgalom alapértelmezetten tiltva van, s itt adhatunk meg ezt felülbíráló szabályokat, vagyis a legitim forgalom számára lyukat tehetünk a tűzfalba. Három fajta szabálycsoport megadására van lehetőség:

- Megadhatjuk, hogy mely távoli ***host***-ok felől érkező forgalom haladhat át legálisan a tűzfalon. Ezt megtehetjük ***IP*** cím vagy ***host*** nevének megadásával.
- Engedélyezhetünk különböző szolgáltatások elérését. Meg kell adni a szolgáltatás nevét (egy legördülő listából választható ki) vagy az általa használt ***port***ot. és, hogy kiknek legyen engedélyezve (Bárkinek, ***LAN*** vagy adott ***IP***-vel rendelkező kliensnek)

Lehetőség van, hogy a program szolgáltatást kérést közvetítsen a helyi hálózat felé. Ha több gép is osztozik egy adott ***Internet*** elérésen, akkor ezek csak egy gépnek látszódnak az ***Internet*** felől, és beállítható, hogy melyik helyi gép, melyik ***port***-jára küldje tovább a ***Firestarter*** az adott szolgáltatási kérést.
Kimenő forgalomra vonatkozó szabályok kezelésére a program kétfajta megközelítést kínál:

- ***Megengedő mód***: Alapértelmezetten így indul a program, a kifelé irányuló forgalom engedélyezve van Szabályozás eszköze: a fekete lista. Az erre felkerülő ***IP*** címhez, kiszolgálóhoz vagy hálózathoz, nem csatlakozhatnak a sem helyi hálózatról sem a ***Firestartert*** futtató gépről. Ezenkívül ezen fülön megadhatjuk, hogy lokális gépek közül melyik nem érheti el az ***Internetet***, illetve letilthatjuk a különböző szolgáltatások elérését.
- ***Korlátozó mód***: Minden olyan kifelé irányuló forgalom tiltva van, mely nincs engedélyezve az un. fehér listán. Maximális védelmet biztosít, de minden hálózati alkalmazásra külön szabályt kell alkotni. Először is megadhatjuk, hogy melyik ***IP*** cím, kiszolgáló vagy hálózat érhető el. Ez például jól jöhet, ha gyerekeink számára le akarjuk korlátozni a meglátogatható honlapokat. Megadhatjuk, hogy a helyi hálózat melyik gépe csatlakozhat az Internethez, és milyen szolgáltatás érhet el.

Mindkét módban az új szabályok csak a ***szabály alkalmazása*** gomb megnyomása után lépnek életbe.
A működő tűzfalunk további finomhangolásra a beállítások menüpont alatt van lehetőség. Az itt megjelenő opciók közül zárásként mindössze egyet emelnék ki. Az ***ICMP*** protokoll lehetővé teszi különböző hibaüzenetek, teszt csomagok küldését egy adott ***host***-nak. Sok ilyen küldésével egy adott gép elérést meghiúsítják, ez az úgynevezett ***DoS*** támadás. Megadhatjuk, hogy a tűzfal korlátozza az ***ICMP*** üzenetek létrehozását és fogadását, illetve, hogy közülük melyikre ne vonatkozzon ez.

Fekete Imre
(imre.fekete@gmail.com)

Programtervező-matematikusként végeztem a Debreceni Egyetemen. A Linuxtól kezdetben idegenkedtem, de ma már csak azt tudom mondani róla, hogy remek rendszer.

Dinamikus web – azoknak, akik nézni szeretik

Nincs dinamikusabb a filmeknél – ráadásul egyesek elképesztő dolgokat művelnek velük. Igen, tudom; vannak, akik inkább csak ülni és nézni szeretnek, ahelyett, hogy maguk csinálnának valamit. Ragadják meg tehát a távirányítót, foglaljanak helyet kényelmesen és élvezzék a borukat!

Igen, ***François***, gyakran így fest a helyzet. Manapság, minden illúziónk ellenére, a világháló nem elsősorban párbeszédre vagy a tudás megosztására szolgál, hanem a szórakoztatás irányába mozdult el. A dinamikus webfejlesztés netovábbja csúcsosodik ki az új film-házhoz-szállítási rendszerben. Tudjuk mindannyian, mennyit lehet tanulni a filmekből, és hogy több olyan webhely áll a széles tömegek rendelkezésére, melyen megoszthatják véleményüket egy-egy filmről, megjegyzéseket fűzhetnek hozzájuk a megfelelő fórumokon. Igaz ugyan, hogy amiket ilyen helyeken találhatunk, azok lényegében „húzd-és-ejtsd" módon lettek kiemelve a televízióból, de többen előnyösnek érzik ezt a viszonylag olcsó filmszerzési csatornát, mert itt megtornáztathatják kreatív izmaikat, megoszthatják gondolataikat a nagyvilággal. Véleményem szerint ez azt mutatja, hogy irdatlan mennyiségű tehetség halmozódik fel e webhelyek körül. Ez a technológia nem csak azt teszi lehetővé, hogy elérhetővé váljon a világ, hanem azt is, hogy a nézők kifejezésre tudják juttatni friss, fiatalos tehetségüket.

Úgy látom, közelednek a vendégeink, ***François***. Íme a mai menü. Olyasvalamit tartogatok, ami egy egészen új szintre emeli az egész internet-televíziózási kérdéskört. Á, már meg is érkeztek! Üdvözöllek benneteket, ***mes amis a Chez Marcelben***, a legfinomabb linuxos fogások és a világ legkitűnőbb borainak otthonában!

Épp arról beszélgettünk ***François***-val, hogy a filmek világhálóba robbanása hogyan fogja átformálni a tartalomszerkesztők és a fogyasztók gondolkodását. Foglaljatok helyet, míg elküldöm hűséges pincéremet a borért. Azt hiszem, még van két dobozzal a 2000-es ***Château La Tour Blanche***-ból a pincében. Mozogj!

Töméntelen mennyiségű tartalom halmozódott fel. Az internetes filmközvetítés („vodcasting") igen élénk, nyüzsgő világot teremt, melyben néha nem egyszerű tájékozódni. Nemrégiben botlottam egy remek programba, mely nagymértékben leegyszerűsíti a filmkeresési zűrzavart, melynek neve: Democracy.

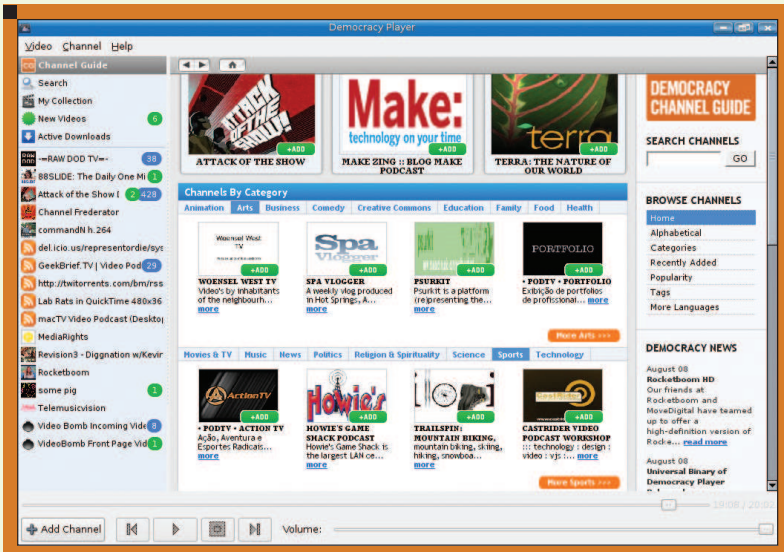
Ez egy igen érdekes küldetéssel útjára bocsátott nyílt forrású internet-televíziózási program. A mögötte álló nonprofit csoport magát *„**Résztvevői Kulturális Alapítványnak**"* (*„**Participatory Culture Foundation**"*) hívja. Sok más efféle csoportosuláshoz hasonlóan őket is az a tapasztalat mozgatja, hogy médiánkat nagyrészt óriáscégek határozzák meg, az ő szűrőiken keresztül jut el hozzánk szinte minden. Úgy érezték, hogy egy nyílt szabvány, a résztvevői internet-televíziózás formáját támogatva lehet leginkább szembeszállni a központosított médiaszerkesztéssel és -felügyelettel. Megoldáskeresésükből született a ***Democracy*** internet-televíziós médialejátszó.

A program lehetővé teszi internet-televíziós programok keresését, letöltését, mentését, manipulálását és megnézését. Ha valakinek van otthon ***TiVo*** (vagy más) digitális felvevője, bizonyára könnyen átlátja e szemlélet szépségét. Alapértelmezetten öt napig maradnak meg a letöltött filmek, utána törlődnek. Természetesen megvan a lehetőség a végleges mentésre is.

A Democracy ismeri a teljes képernyős üzemmódot, így lehet élvezni például a 21 hüvelyk képátlójú lapos képernyő előnyeit. A beépített csatornakalauzzal, közösségileg működtetett szavazórendszerrel, valamint a szükséges interaktív közzétételi eszközökkel megspékelt ***Democracy*** program jóval több, mint egyszerű tévénéző eszköz. Úgy sejtem, hogy bennetek sokkal több van, mintsem hogy csak nézzétek a közzétett tartalmakat, úgyhogy rögtön mindent el mesélek is ezekről a többletszolgáltatásokról... de most úgy látom, ***François*** visszatért. Kérlek, ***mon ami***, győződjön meg mindegyikőtök arról, tele van-e a pohara.

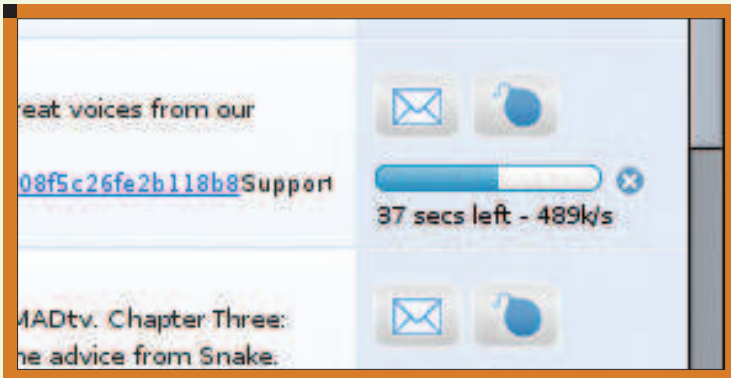
A ***Democracy*** program saját példányának beszerzéséhez látogassatok el a honlapjára (lásd a cikkhez tartozó online forrásokat). Található itt ***Ubuntu***, ***Fedora*** és teljesen egyszerű ***Debian*** csomag is, valamint a forráskód is elérhető, ha egyik csomag sem használható saját disztribúciótokban. Nem nagy művészet a ***Democracy TV*** telepítése, de annyit tudni kell róla, hogy szüksége van a ***Mozillára*** és

annak függőségeire. Amiatt említem meg ezt külön, mert többen *Firefoxot* használunk *Mozilla* helyett.
Ha a *Mozillát* csomagként töltitek le, győződjetek meg arról, hogy a fejlesztői és a *PSM* csomagok is letöltődtek-e.
A *Democracy* első futtatásakor elindul egy „*Hogyan kezdjük*” bemutató („*How to Get Started*”). A második indítástól már az online csatornakalauzt mutatja meg először a lejátszó (*1. ábra*), amikor is különböző kategóriák véletlenszerűen választott csatornái jelennek meg a képernyőn. Ha a kép alatti rövid leírás nem kielégítő, rá lehet kattintani a „*tovább*”-ra („*more*”) a további információkért. Ha jól hangzik az összefoglalás, vegyük fel a csatornát kedvenceink közé a zöld „*Hozzáad*” („*Add*”) gombra kattintva.
A csatornakalauztól jobbra eső sávban egyéb navigációs lehetőségeket találhatunk. Lehet kulcsszavak alapján csatornát keresni, (mint például a „*Linux*” vagy az „*open source*”) vagy más módokon böngészni a csatornákat: névsorba rendezve, különböző kategóriák szerint bejelölve (vagy a kategóriák kulcsszavai alapján), vagy akár a népszerűségi lista szerint is haladhatunk.
A felhasználói felülettel igen könnyű dolgozni, nem vonja el a figyelmet a műsorok nézésétől. A baloldali menüsáv tetején egy kis menürendszer látható, alatta sorakoznak a kedvencként bejelölt (azaz „*előfizetett*”, „*subscribed*”) csatornák. A *Democracy* egy tucatnyi „kedvenc” megmutatásával indul – természetesen nem kell őket megtartani, de ezekből jól megérezhető a program stílusa. Ha hozzám hasonlóan egyből meg szeretnétek nézni valamit, akkor kezdjünk is neki. Kattintsatok valamelyik csatornára: megjelenik a pillanatnyilag elérhető sorozatrészek listája a főablak bal oldalán (*2. ábra*). Ez a főablak ugyanaz, mint ahol a műsorok is nézhetőek.
Az oldalon felsorolt filmek mindegyikét egy kis bélyegkép jelzi a baloldalon, és egy összefoglalás is olvasható róla jobboldalt. Itt gyakran egy link is található, mely arra a weboldalra vezet, ahonnan az adott film származik. Vizsgáljuk csak meg ezt a bélyegképet! A jobb alsó sarokban látható

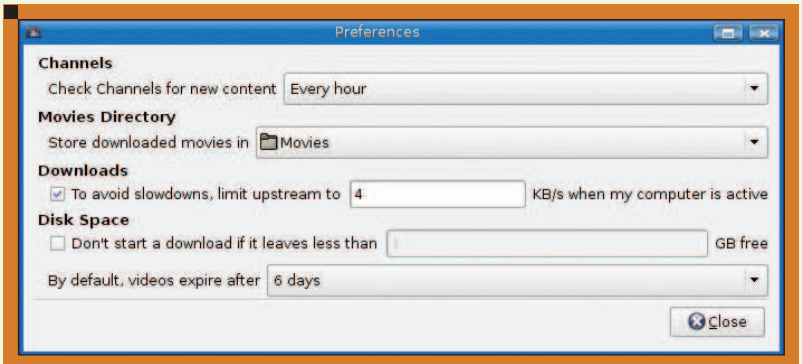


1. ábra Könnyű megszeretni a Democracy beépített csatornakalauzát. Az érdekes műsorok egy-egy kattintással összegyűjthetők.

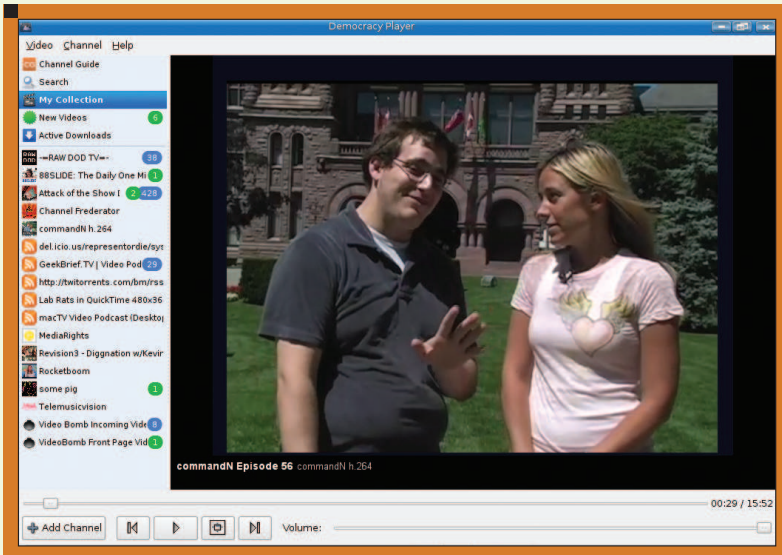
2. ábra Egy csatorna kiválasztása előhozza a pillanatnyilag letölthető sorozatrészek listáját



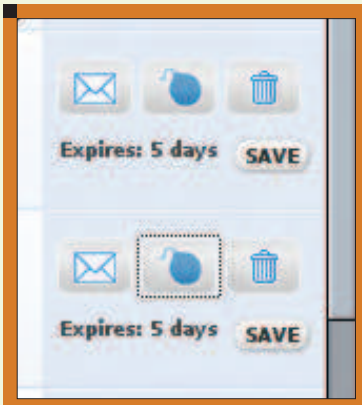
3. ábra A műsorok letöltésének folyamatát grafikusan is nyomon követhetjük



4. ábra A legtöbb beállítási lehetőség a (tár)hellyel és az idővel (lejárati időintervallummal) kapcsolatos



5. ábra Ezt nyugodtan hívhatjuk úgy, hogy „akcióban a demokrácia"



6. ábra Oszd meg barátaiddal legjobb filmélményeidet – „bombázd" őket!

egy lefelé mutató fehér nyilacska kék körben (mint egy fejre állított egyirányú utca tábla). Erre kattintva elkezdődik a film letöltése és gyűjteményünkben való eltárolása. Ne feledjük, hogy nem folyamatosan töltődő (***streaming***) formátumot igénylő alkalmazással van dolgunk, hanem egy digitális felvevővel. Folyamatjelző sáv mutatja baloldalt, hogy hol tart a film letöltése (*3. ábra*).

Ahogy az ábrán is látható, a letöltési sebesség és a becsült hátralévő idő is leolvasható a folyamatjelző sáv mellett. Természetesen nem kell megvárni, míg teljesen letöltődik az adott film. Annyit lehet elindítani, ahányat csak akartok. A letöltések a háttérben zajlanak. Ahogy megérkeznek a letöltött filmek, bekerülnek a gyűjteménybe, melyet a baloldali sáv felső részén levő „csapó" ikonra kattintva lehet megtekinteni.

A program első indításakor létrejön egy ***.democracy*** nevű könyvtár a saját könyvtárunkban. Ebben a könyvtárban keletkezik egy másik is, melynek ***Movies*** (***Filmek***) lesz a neve. Amint az sejthető, a teljes filmek letöltése és tárolása az idők folyamán elképesztő mennyiségű tárhelyet fel fog emészteni. A partíciók méreteitől függően elképzelhető, hogy érdemes valahol máshol tárolni a filmeket. Ennek beállítására kattintsunk felül a Video menüpontra, és válasszuk a ***Preferences (Beállítások)*** almenüt. Másodperceken belül megjelenik a megfelelő párbeszédpanel (*4. ábra*).

A ***Democracy*** rendszeresen végignézi a kedvencekként bejelölt csatornákat, van-e ott újabb film. Az alapértelmezett óránkénti figyelés átállítható a felül levő mezőben. A tárolásra vonatkozó részleteket szeretném kiemelni. A ***„Movies Directory"***-ra (***„Mozikönyvtár"***-ra) kattintás után fájlválasztóban lehet kijelölni a megfelelő tárhelyet. Még ne zárjátok be a panelt! Más érdekes beállítási lehetőségek is elérhetőek itt. A fájlok elérési útján kívül ide tartozik az is, hogy mekkora helyet szeretnétek a letöltött filmek rendelkezésére bocsátani.

Ha semmit nem választotok, két dolog fog történni. Az egyik: a ***Democracy*** annyi filmet fog letölteni és tárolni a kívánságoknak megfelelően, amennyi csak elfér az adott merevlemez-helyen. A másik: a letöltés utáni hatodik napon nyomtalanul törlődnek a filmek. Ha ehelyett három napra szeretnétek állítani ezt az időkorlátot, itt megtehetitek. Végül ki szeretném emelni a ***„Disk Space"*** (tárhely) alapértelmezetten üresen tátongó jelölőmezőjének fontosságát. Ha ez be van jelölve, akkor megadható, hogy a fájlok letöltésének megkezdése előtt mekkora szabad tárhely létéről kell megbizonyosodnia a ***Democracy*** programnak. Ha minden szükséges változtatás készen van, zárjátok be a párbeszédpanelt.

E cikk végül is az internet-televízió nézéséről szól, úgyhogy térjünk vissza oda. Miután végeztetek a letöltésekkel, következhet a ***„My Collection"*** (***„Gyűjteményem"***) gomb, amivel meg tudjuk nézni, mely filmek lettek letöltve a helyi gépre. Minden bélyegképhez tartozik egy kis „lejátszás" gomb: zöld körben fehér háromszög.

Rákattintva a beépített médialejátszó elindítja a filmet (*5. ábra*). Csúszkán látható, mennyi telt már el a filmből – mellette található a hangerő-beállító. A ***Democracy***-nek szép nagy lejátszóablaka van, de miért ne használnánk ki a teljes képernyős üzemmódot? A kép alján (a lejátszás/szünet gombtól jobbra) levő gomb használatával átkapcsolhatunk erre.

Az ember egyik filmet nézi a másik után, és egyszer csak kezdi úgy érezni, meg tudja különböztetni a jó filmeket (amit érdemes lenne végleg elmenteni) a gyengéktől (amikre egy bitet sem érdemes pazarolni). Némelyiket jó volna ajánlani másoknak. A ***Democracy*** lehetőséget ad minderre. A sorsszerű törlődés elkerüléséhez lehetőség van mentésre (*„Save”* gomb a leírás mellett, *6. ábra*), a törléshez a kuka ikon használható. Végül, ha egy filmet annyira élveztetek, hogy ezt világgá kell kürtölni, akkor az alábbiak közül lehet választani: a boríték ikonra kattintva e-mailben lehet erről hírt adni az ismerősöknek, vagy választható a jobboldali bomba ikon is (*6. ábra*). Nem, ez nem fogja felrobbantani a készüléket.

Egy film bombázása nagyon jó dolog – mindjárt kiderül, miért. Hatására tűzbe jön az alapértelmezett böngésző (például a ***Konqueror*** vagy a ***Firefox***) és elvisz a szóban forgó film oldalára, a ➲ ***videobomb.com*** weboldalra. Itt a szavazatotok (mint egy bomba) hozzáadódik a többiekéhez, ezáltal növelve eme film sikerességének mutatóját. Ehhez előzőleg létre kell hozni egy azonosítót az említett webhelyen, ahol nemcsak efféle szavazásra van lehetőség, hanem alkalom nyílik beszélgetésre, beszélgetésekre ismerősökkel, barátokkal: megbeszélhetőek a „bombázott” filmek.

A fiatal hölgy a 12-es asztalnál azt súgja, hogy van más efféle társasági webhely is, és igaza van. A szóban forgó honlapot azonban éppen a ***Democracy***-vel való szoros köteléke különbözteti meg a többitől. Ilyen értelemben ez a webhely teszi teljessé a képet az élményről: csatornák böngészése, filmek rögzítése és nézése, végül megtárgyalása. Ez olyan, mint amikor cigarettaszünetben a munkatársaiddal az esti filmről beszélgetsz, csak ez a lehetőség itt azonnal elérhető.

Mon Dieu! Már ennyi az idő? Az óra ismét azt mondja, hogy be kell zárnunk kapuinkat, ***mes amis***. Gondolom, nem ez lesz az első alkalom, hogy órákat töltöttünk el a televízió előtt, késő éjszakába nyúlóan. Csak ritkán van ilyen gazdag választék előttünk. Talán François lesz olyan kedves és újra teletölti poharainkat egy utolsó koccintásra. Emeljük poharainkat, ***mes amis***, és igyunk egymás egészségére! A ***votre santé! Bon appétit!***

Linux Journal 2006., 151. szám

Marcel Gagné
(mggagne@salmar.com)
Mississaguában, Ontario államban él.
Ő a szerzője a Kiskapu kiadásában tavaly szeptemberben megjelent Linux-rendszerfelügyelet (ISBN 96-9301-40) című könyvnek.

A CIKK FORRÁSAI

➲ www.linuxjournal.com/article/9259

Terítéken a podcast

Akit nem szédül el a bortól, biztosan megrészegül ezektől a podcast-letöltő eszközöktől.

Nyugodj meg, ***François***! Nem akartalak megijeszteni. Azért húztam ki a fülhallgatódat, mert nemsokára megérkeznek a vendégeink, és úgy tűnt, máshogy nem tudom elérni, hogy rám figyelj. Egyébként mit hallgatsz? A ***New Scientist*** podcastot? Ez igen, ***mon ami***. Nem tudtam, hogy érdekelnek a tudományos történetek, be kell vallanom, hogy ez a ***podcast*** nekem is az egyik kedvencem. Néhány műsort rendszeresen letöltök, és ez is köztük van. Időnként azon a Linux rendszeren hallgatom ezeket, ahová letöltöttem őket. Máskor a kézi számítógépemre másolom, és magammal viszem őket. A ***podcastok*** remek időtöltésül szolgálnak vonaton vagy váróteremben ülve. A legjobb az egészben, hogy rengeteg kitűnő műsör létezik, amatőröktől és hivatásosoktól egyaránt, szinte minden elképzelhető témában.
Quoi? Azt mondod, sokáig tart? De ugye nem kézzel töltöd le az összes ***podcastot***, vagy igen? Jobbgomb, és ***Save As (Mentés másként)*** a webhelyen? Komolyan, ***François***, ennél sokkal jobb módszerek léteznek. Szerencsére ma éppen ezek a hasznos programok szerepelnek az étlapon. Most viszont látom, hogy megérkeztek a vendégeink, és azonnal velük kell törődnünk. Isten hozott mindenkit ***Marcelnál***, a remek borok, mennyei Linux fogások és természetesen a legkiválóbb törzsközönség otthonában! Engedjék meg, hogy az asztalukhoz kísérjem önöket. Helyezzék kényelembe magukat! ***François***, a pincébe! Menj az északi szárnyba, és hozd ide a 2002-es új-zélandi ***Pinot Noir***-t, kérlek, a ***Marlbourough***-t.

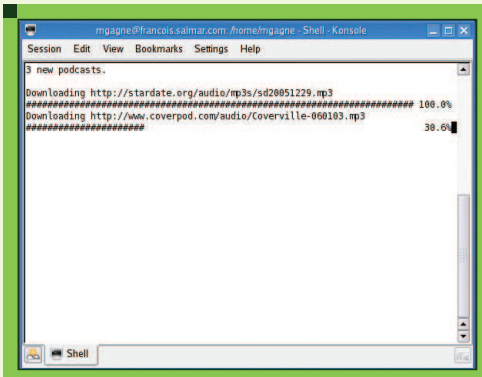
François és én éppen a ***podcastokról*** beszélgettünk, és azt taglaltuk, hogy szükség van egy olyan módszerre, amellyel hatékonyan lehet letölteni és gyűjteni azokat. A ***podcastok*** felkutatása, az új műsorok keresése és ezek letöltése rengeteg időt vehet igénybe. Ezért olyan nagyszerű találmány a ***podcast***-gyűjtő, és ezért létezik olyan sok ügyfél (kliens). Vannak a borhoz hasonlóan nehéz és összetett grafikus ügyfelek – nemsokára ilyet és látunk. Vannak olyanok, amelyek sokkal egyszerűbbek. ***Linc Fessenden BashPodder*** programja az egyik legegyszerűbb gyűjtő, amit valaha láttam. A ***BashPodder*** nem más, mint egy ***Bash*** héjprogram, és annak is apró. Menjünk a ***BashPodder*** webhelyre (lásd a kapcsolódó címeket), és töltsük le a ***bashpodder.shell*** parancsfájlt, majd a ***parse_enclosure.xsl*** fájlt. Amikor megvan mind a kettő, másoljuk őket egy tetszőleges könyvtárba. Ezután vessünk egy pillantást a parancsfájlra. Kisebb, mint 1 KB, és hihetetlenül egyszerű.
Kitűnő, ***François***, nagyon hamar visszaértél. Kérlek tölts a vendégeinknek. Egészségükre, ***mes amis***.
A ***BashPodder*** webhelyen egy példabeállítást, a bp.conf fájlt is le lehet tölteni. A példafájlban megtalálhatók Linc kedvenc podcast tartalmai, így ha valakit érdeklődik az ő ízlése iránt, letöltheti, de a formátum annyira egyszerű, hogy nem lesz rá szükség. A bp.conf csupán tartalomhivatkozások gyűjteménye. Példaként lássunk egy három tartalomhivatkozást tartalmazó fájlt. Az egyik a ***New Scientist*** podcast, a másik a ***Coverville Brian Ibbottól*** származik, míg a harmadik az ***In Our Time*** a BBC-től:

```
http://www.newscientist.com/
➥podcastfeed.ns
http://www.coverville.com/
➥index.xml
http://www.bbc.co.uk/radio4/
➥history/inourtime/mp3/
➥podcast.xml
```

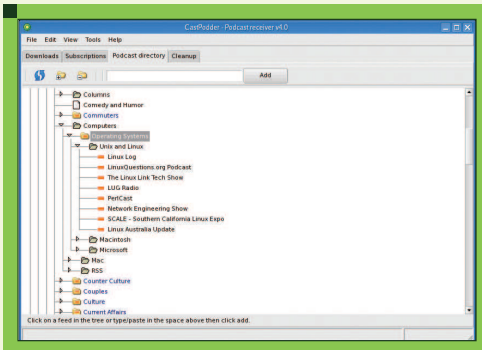
Ezt a fájlt ugyanabban a könyvtárban tárolom, mint amelyikből a ***bashpodder.shell*** parancsfájlt futtatom:

```
sh bashpodder.shell
```

A parancsfájl létrehoz egy könyvtárat, amely a futás dátumát kapja névként (például 2006-01-04). A ***BashPodder*** csendes üzemmódban fut, és letölti az összes aktuális ***podcastot*** ebbe a könyvtárba. Állítsuk be a programot, hogy ***cron*** feladatként fusson, így mindig naprakészek leszünk. Ne feledjük azonban, hogy a ***BashPodder*** az adott tartalom összes aktuális letöltését menti, válogatás nélkül.
A ***GoldenPod (Eskild Hustvedt)*** szintén parancssor-vezérelt program, amelyet könnyű használni, és könnyű ***cron*** feladatként beállítani az önműködő letöltéshez. Ez a program ***Bash*** helyet ***Perlt*** használ, és olyan hasznos szolgáltatásokat is biztosít, amelyek a sokkal egyszerűbb ***BashPodderben*** nem szerepelnek. A ***GoldenPod*** csendes üzemmódban is tud futni (a `-s` parancssori kapcsolóval), de arra is képes, hogy működés közben jelentést küldjön a folyamatairól. Megadhatjuk, hogy egy sorozatból csak a legfrissebb ***podcastot*** töltse le, és ne az egész sorozatot (ami időnként hatalmas lehet). A ***GoldenPod*** ezen kívül képes letöltés nélkül felsorolni az elérhető ***podcastokat***, hordozható lejátszóra



1. ábra A GoldenPod egy egyszerű, parancssor-vezérelt podcast-gyűjtő



2. ábra Nem kell podcast-jegyzékekben kutatnunk, mert a CastPodder széles körű, saját listát biztosít

másolni a fájlokat, törölni a régi ***podcastokat***, valamint statisztikát készíteni arról, hogy a ***podcastok*** mennyi helyet foglalnak. Mindezt kevesebb, mint 32 KB-ban.

Töltsük le a programot ***GoldenPod*** webhelyről (lásd kapcsolódó címek), majd tömörítsük ki a .tar csomagot egy tetszőleges könyvtárba.

Ha ebben a könyvtárban futtatjuk a `./goldenpod` parancsot, a program azt jelzi, hogy a ***podcast-lista*** üres, viszont egyúttal létrehoz két új könyvtárat a saját könyvtárban. Az első neve ***.goldenpod*** – ez a beállításfájlokat és a ***podcast-listát*** tartalmazza. A második a ***Podcast***, ide töltődnek a ***podcastok***. A ***~/goldenpod*** könyvtárban két fájl található: a ***goldenpod.conf*** és a ***podcasts.com***. A ***podcasts.conf*** fájl – egy szövegfájl, amelyet a kedvenc szerkesztőprogramunkkal szerkeszthetünk – formátuma megegyezik a ***BashPodder bp.conf*** fájljával, az-az gyakorlatilag ***podcast*** tartalomcímek listája.

A tartalmak hozzáadása után nem kell mást tenni, mint újból futtatni a programot a letöltés elindításához. Érdemes kipróbálni a

```
goldenpod --dry-run
```

parancsot, hogy lássuk, mire számíthatunk, illetve a

```
goldenpod -f
```

parancsot, hogy csak a legfrissebbeket töltsük le. Ezután a terminálon láthatjuk a letöltések folyamatát (*1. ábra*). Miután a letöltés elkészült, a ***podcast*** könyvtárakban egy-egy, a tartalomnak megfelelő m3u lejátszólista-fájlt találunk (például ***~/Podcasts/catalog/showname/showname.m3u***). Igaz, hogy minden a parancssorban történik, de megismétlem: ez egy remek program, amelyet ***cron*** feladatként futtathatunk a háttérben, csendesen, talán éppen amikor alszunk. Ezen kívül időnként érdemes a `--stats` beállítással futtatni a programot, hogy lássuk, hány műsorunk van, és hogy mennyi helyet foglalnak.

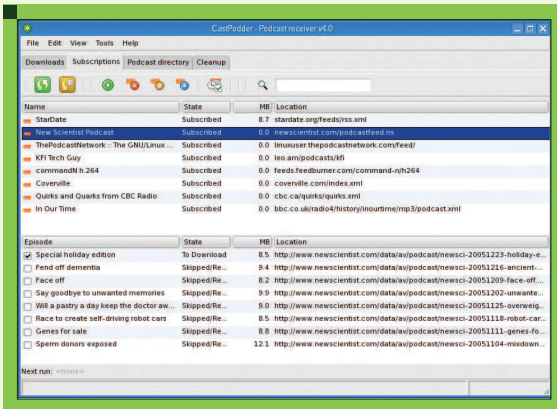
Természetesen grafikus podcast-gyűjtők is léteznek a Linux alatt, amelyek jellegüknél fogva kibővítik az imént említett két program szolgáltatásainak körét. A tökéletes podcast-gyűjtőnek először is saját podcast-könyvtárat kell tartalmaznia, amelyeket könnyen kereshető témákba csoportosít. Ehhez vegyünk egy egyszerű, egygombos módszert a tartalmak hozzáadására, egy új műsorokat pásztázó szolgáltatást, hogy ne nekünk kelljen megkeresni a legfrissebb adásokat, egy automatizált, időzített, háttérben futó letöltésrendszert, és máris minden adott egy nagyszerű programhoz. Úgy tűnik, ***Scott Grayban CastPodder*** alkalmazása képes mindenre. Tegyük meg az első lépést a podcast-nirvána felé, és látogassunk el a ***CastPodder*** webhelyre (lásd a kapcsolódó címeket), majd töltsük le a program legfrissebb változatát. Itt ***Mandriva*** és ***Debian*** csomagokat, továbbá tar és gzip forráscsomagokat találunk.

A ***CastPodder*** az első indításkor (parancsnév: ***CastPodder***) létrehoz egy ***iPodderData/downloads*** nevű könyvtárat és alkönyvtárat a saját könyvtárban. A ***File*** menü ***Preferences*** elemének kiválasztásával megváltoztathatjuk ezt az alapértelmezett beállítást. Maga a ***CastPodder*** kezelőfelülete a szokásos, felül húzódó menüsávot kínálja, amelyben elérhetjük a program összes szolgáltatását. Négy fő lap áll rendelkezésre: ***Download (Letöltés)***, ***Subscriptions (Feliratkozások)***, ***Podcast Directory (Podcast-jegyzék)*** és ***Cleanup (Törlés)***. Ha az indításkor nincs kijelölve, kattintsunk a ***Podcast Directory*** lapra, és megjelenik egy könyvtárakat tartalmazó lista, amely különböző elérhető jegyzékeket és a népszerű podcast-gyűjteményeket jelenít meg. Mielőtt elkezdenénk érdekes műsorokat keresgélni, kattintsunk az ikonsáv bal szélén található ***Refresh (Frissítés)*** gombra. Így biztosan naprakész listát követünk.

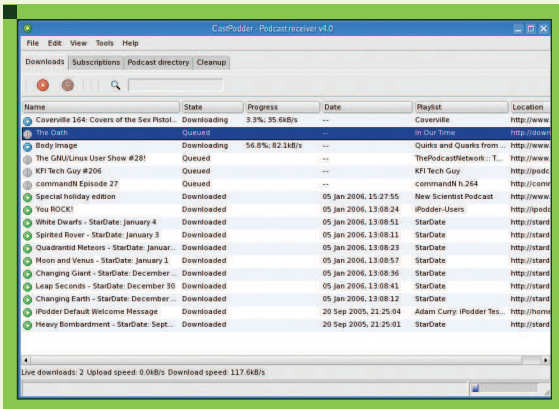
Most térjünk vissza a podcast-jegyzékekhez. Talán az ***iPodder.org*** : ***Podcasting Central*** könyvtár a legérdekesebb, mert az ***iPodder.org*** terjedelmes podcast-listáját tartalmazza. A kategóriák, alkategóriák, illetve a kategóriákhoz tartozó különböző podcastok megjelenítéséhez kattintsunk a mappa melletti kis nyílra.

Ha fel akarunk iratkozni valamelyik podcastra, kattintsunk a címére, majd az ***Add (Hozzáadás)*** gombra. Ha a kívánt podcast itt nincs felsorolva, a menüsáv ***Tools (Eszközök)***, ***Add a Feed (Tartalom hozzáadása)*** menüjét választva kézzel hozzáadhatjuk azt.

Amikor feliratkozunk a különböző podcastokra, az adott podcastok nevét viselő könyvtárak jönnek létre az ***~/iPodderdata/downloads*** könyvtáron belül. Nekem például egyebek mellett ***Coverville***, ***New Scientist Podcast***, valamint ***Quirks and Quarks from CBC Radio*** nevű könyvtáraim vannak. Az egyes könyvtárakban az adott műsorok podcastjai találhatók, tehát



3. ábra A feliratkozás-lista segítségével önálló epizódokat választhatunk ki a letöltéshez



4. ábra A Downloads lap megjeleníti az egyes podcast-letöltések állapotát

gondoskodjunk arról, hogy sok hely legyen a merevlemezen.
Térjünk vissza a ***Subscriptions*** lapra (*3. ábra*). A fő ablak felső felében látható az összes olyan podcast listája, amelyre feliratkoztunk. Ha a műsorok valamelyikére kattintunk, az alsó részen megjelennek a különböző epizódok, az adás mérete és a webcíme. Az epizódcímtől távolabbra, balra láthatunk egy jelölőnégyzetet. A négyzetre kattintva letöltésre jelöljük ki az adott epizódot. Ha egy kis zöld nyíl szerepel az epizódcím mellett, akkor az adott rész már letöltődött a ***Linux*** rendszerre.
Amikor kiválasztottuk a kívánt epizódokat, kattintsunk az ikonsáv bal szélén található első gombra, amelyiken a ***Scan for new Feeds (Új tartalom keresése)*** felirat jelenik meg, ha fölé visszük az egeret. A gomb szürkére változik, és megkezdődik a ***podcastok*** letöltése.
A ***Download*** lapra kattintva megtekinthetjük a letöltések folyamatát. Itt látjuk a letöltés aktuális állapotát, a sebességet, valamint az átvitel befejezettségének százalékértékét (*4. ábra*).
Végül elérkezünk a műsorok keresésének és letöltésének valódi értelméhez: a hallgatáshoz. Vessünk ismét egy pillantást a letöltött epizódok bal oldalán szereplő kis zöld nyílra (ez a ***Download*** és a ***Subscriptions*** lapra egyaránt érvényes). Kattintsunk duplán a bejegyzésre, vagy kattintsunk a jobb gombbal, majd a felugró menüből válasszuk a ***Play episode in media player (Epizód lejátszása a médialejátszóban)*** lehetőséget. Alapesetben az ***XMMS*** kezdi lejátszani a kiválasztott műsort (*5. ábra*). Megváltoztathatjuk a felkínált médialejátszót, de csak az ***XMMS***-t, a ***Beep Media Playert***, vagy a médialejátszó hiányát választhatjuk. A módosítás elvégzéséhez a kattintsunk a menüsor ***File***, ***Preferences (Tulajdonságok)*** menüpontjára, ezután pedig a ***Preferences*** ablak ***Player (Lejátszó)*** lapjára.
A ***CastPodder*** kellemes, sárgás kisalkalmazássá zsugorodik a rendszertálcán, így a program tovább futhat a háttérben. Ez különösen a ***CastPodder*** időzítő szolgáltatása miatt hasznos.
A ***Subscriptions*** lapon kattintsunk az ikonsáv ***Scheduler (Időzítő)*** gombjára a podcast-letöltések automatizálásához. Beállíthatjuk, hogy a jegyzett ***podcastok*** frissítése és letöltése a nap egy meghatározott időpontjában vagy a nap folyamán rendszeres időközönként fusson. Így nem kell időt szánnunk az új epizódok figyelésére. Engedjük, hogy a ***CastPodder*** dolgozzon helyettünk.
Úgy tűnik, ***mes amis***, nemsokára záróra. Azért még mielőtt befejezném, szót kell ejtenem egy utolsó ***CastPodder*** szolgáltatásról, ami a takarítással kapcsolatos. Korábban említettem, hogy jó, ha sok szabad helyünk van, arra az esetre, ha rászoknánk a sok ***podcast*** letöltésére. Ezen kívül érdemes folyamatosan törölni a régi műsorokat, miközben az újak töltődnek. A ***Cleanup*** lapra kattintva a ***CastPodder*** lehetővé teszi, hogy a ***podcast*** tartalom neve alapján egyes epizódokat jelöljünk ki, és könnyedén töröljük azokat. Ha minden epizódot törölni akarnánk, egyetlen kattintással kijelölhetjük az összeset.
Elérkezett a záróra, de látom, hogy sokan éppen a podcast-éter számos érdekes műsorát böngészik és feliratkoznak azokra. Van még egy kis idejük, ***mes amis***. Biztos vagyok benne, hogy ***François*** nagyon szívesen újratölti a poharukat még utoljára, mielőtt elbúcsúznánk. Hmm... talán valaki megnézhetné, hogy van-e borokkal foglalkozó podcast. Ha már itt tartunk, emeljük poharunk, ***mes amis***, és igyunk egymás egészségére!
A votre santé! Bon appétit!

5. ábra A CastPodder alapértelmezésben az XMMS-t indítja el a podcastok lejátszásához

Linux Journal 2006., 144. szám

Marcel Gagné
(mggagne@salmar.com)
Mississaguában, Ontario államban él.
Ő a szerzője a Kiskapu kiadásában tavaly szeptemberben megjelent Linux-rendszerfelügyelet (ISBN 96-9301-40) című könyvnek.

KAPCSOLÓDÓ CÍMEK

- www.linuxjournal.com/article/8829

Linux lemezrészek elérése Windows alól

Windows operációs rendszerünk alapból nem látja másik kedvelt rendszerünket, a Linuxot, mivel csak egy ismeretlen partícióként tünteti fel. Erre kell megtanítani, hogy gond nélkül képes legyen olvasni, és írni is azt.

■ Sok program létezik már ***Ext2*** fájlrendszer olvasására, ezek közül most négyet mutatunk be bővebben. Szerencsére ezek a szoftverek ingyenesen letölthetőek, és használhatóak. A programok funkciói megegyeznek, csak néhánynál le van tiltva az ***Ext2*** fájlrendszerre való írás, hogy kárt ne tehessünk ***Linux*** operációs rendszerünkben.

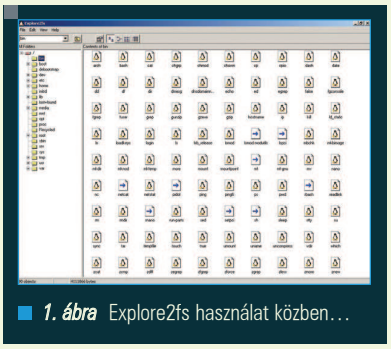
Csak olvasási képességgel rendelkező csomagok

Explore2fs

Ingyenes program, mely letölthető a ➲ ***http://uranus.it.swin.edu.au/~jn/linux/*** oldalról. Legfrissebb stabil verziója ***1.07***, letölthető forráskód és ***Windows*** alkalmazás alakban is.
Ha a programot használni szeretnénk, akkor az utóbbit válasszuk. Miután letöltöttük, és kicsomagoltuk a tömörített állományt, szemügyre vehetjük, hogy a programot nem kell telepíteni, egyből futtatható.
Ha elindítjuk a programot, kicsit olyan érzésünk van, mintha a ***Windows Intézőben*** lennénk, hasonló az elrendezés, de itt csak a ***Linux*** partíciónkat tudjuk használni. A program nagyon egyszerűen kezelhető, nincsenek felesleges menüpontok, csak ami szükséges. A fejlécben a szokásos ***Windows*** menüpontok, úgymint a ***Fájl***, a ***Szerkesztés***, a ***Nézet*** és a ***Segítség*** gombok foglalnak helyet. Ezek alatt található a pontos helyzetünk, majd egy ***Vissza***, egy ***Tulajdonságok*** és ***Nézet*** gombok. Végül pedig a legnagyobb részt a böngésző tölti ki, baloldalon a mappák szerkezete fa struktúrában, jobboldalon pedig az adott mappa tartalmát látjuk. A ***Tulajdonságok*** gomb nagyon részletes információt szolgáltat egy mappára vagy egy fájlra rákattintva, leolvashatunk szinte minden fontos információt róla, attribútum, blokkméret, létrehozási/módosítási dátum stb.
Ami nagyon fontos, hogy nem írhatjuk a ***Linux*** partíciónkat. A program szerzője kivette az ***1.05*** verziótól az írási funkciót, tehát nem törölhetünk, módosíthatjuk fájljainkat, és nem hozhatunk létre új könyvtárakat. Mindössze olvashatjuk, és másolhatjuk a ***Linux*** partíció tartalmát a ***Windows*** rendszerünkre, vagy ahova szeretnénk.

Total Commander Ext2+Reiser bővítménnyel, avagy fájlkezelőnk fejlesztése

A ***Total Commandert*** talán senkinek nem kell bemutatni, a legnépszerűbb fájlkezelő program ***windowsos*** körökben. Sok jó kiegészítő van hozzá, köztük van egy ***ext2*** fájlrendszer olvasó, ami ***Ext2+Reiser*** nevet viseli, nézzük, mivel leszünk gazdagabbak telepítése után. Ha már rendelkezünk ***Total Commanderrel***, akkor csak a bővítményt (plugin) kell letöltenünk hozzá a ➲ ***www.ghisler.com/index.htm*** weboldalról, de az említett fájlkezelőt is innen szedhetjük le. Telepítése nem okoz nagy nehézséget, kicsomagolás után helyezzük el a tartalmát bárhova a merevlemezünkre, majd indítsuk el a ***Total Commandert***. Itt kattintsunk a ***Beállítások*** ikonra (***Configuration***), majd az ***Általános beállításokat*** válasszuk ki (***Options***). Egy új ablak tárult elénk, itt a ***Működést*** (***Operation***) és végül



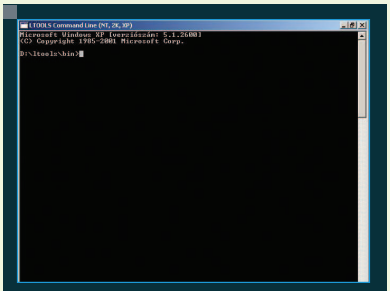
■ ***1. ábra*** Explore2fs használat közben…

az ***FS beépülőkre*** kattintsunk.
Itt hozzáadjuk a letöltött állományt, és készen is vagyunk a telepítéssel. ***Linux*** partíciót a hálózati kapcsolatoknál találjuk meg ***Linux-drives*** néven. Biztonsági okokból a fájlrendszerre való írást ebben a programban sem találjuk meg, be kell érnünk tehát a már megszokott olvasással. A mappákat elég gyorsan jeleníti meg a program, és stabilitásra sincs panasz. ***Szétkapcsolás*** ikonra kattintva választhatjuk le a partíciót.

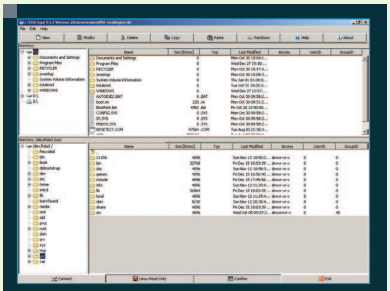
Akik olvasnak, és írnak is...

LTOOLS

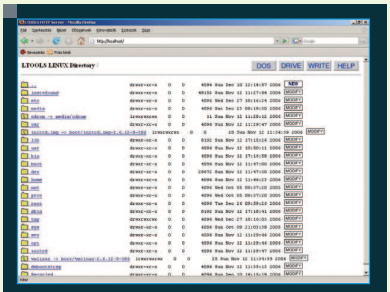
Az *LTOOLS* programot, vagy régebbi nevén ***LREAD*** a ➲ ***http://www.it.fht-esslingen.de/~zimmerma/software/ltools.html*** internetes oldalon tölthetjük le. Telepítés után ez a program már nagyobb helyet foglal, mint testvérei, de funkcióban gazdagabb is náluk. Ha elindítjuk a programot, egy parancsértelmező tárul elénk, innen hívhatjuk elő a többi funkciót. A parancsértelmező hasonló a ***Windowsban*** már megszokotthoz, annyi különbséggel, hogy a prompt



2. ábra LTOOLS parancsértelmezője



3. ábra LTOOLS grafikus környezetben



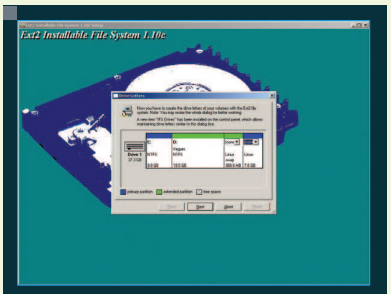
4. ábra LTOOLS Firefox alatt

sose változik, legalábbis nekem mindig a program helyét mutatta, ahova telepítve lett, nem az aktuális könyvtárat, amiben éppen jártam. A parancsok listáját megtalálhatjuk a súgóban, vagy a *readme* fájlban, hasonlóak a *Linuxban* használt parancsokhoz, és szintén rengeteg kapcsolóval könnyíthetjük meg például a listázást, vagy a másolást. Amennyiben nem a parancssort akarjuk használni, választhatjuk a grafikus módot is, illetve a webböngészőnkben való használatot is. A grafikus mód elindításhoz elsősorban szükségünk lesz *Java Runtime environment* minimum *1.2-es* verziójára, vagy a *Microsoft .NET Framework 1.1-es* verziójára. A két mód között sok különbséget nem vettem észre, csak az indításukban.

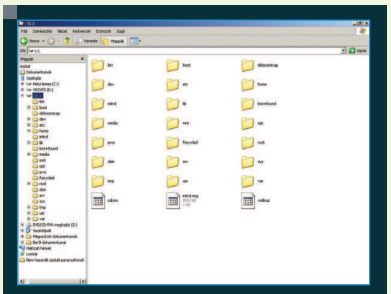
`Ltoolsgui` parancs a *Java*-s módhoz kell, a másikhoz pedig az `ltoolsnet` parancs szükséges.
Egy nagyon jól használható felületet kapunk, a *Linux* partícióra való írást is be lehet kapcsolni, ha a *Fájl* menüben kivesszük a pipát a *Linux read only* felirat mellet. A grafikus mód két nagy részre van osztva, alul a *Linuxos* partíciónk, felül pedig a többi partíciót láthatjuk. Ezen belül baloldalon a szokásos mappaböngésző fa struktúrában megjelenítve, jobboldalon pedig az aktuális könyvtár tartalma helyezkedik el. Amit én hiányoltam ebben a módban, az az állapotjelző csík volt, amikor másolunk vagy áthelyezünk egy fájlt vagy mappát. Ez különösen akkor zavaró, ha egy nagyobb méretű dolgot akarunk másolni, áthelyezni, mert nem tudjuk követni, hogy a program még dolgozik, vagy esetleg lefagyott. Ha böngészőnkben akarjuk használni a programot, akkor az `lreadsrv` parancsot adjuk ki. Ha van tűzfal telepítve a számítógépen, akkor ez riasztani fog, és blokkolni szeretné azt, mivel a program egy portot nyit meg. Engedélyezzük a használatot, és indítsuk el a böngészőnket, majd a böngészősávba gépeljük be a `localhost` utasítást. Egy *FTP-hez* hasonló dolgot kapunk, *Linux* partíciónk tartalmát láthatjuk, és letölthetünk bármit, amit akarunk.
Én egy kicsit lassúnak éreztem ezt a módot, és kényelmetlennek, de ha nincs kéznél *Java*, vagy *Microsoft. NET*, akkor tökéletes is lehet.

EXT2 IFS – EXT2 Installable File System for Windows

Egy kicsi, de annál hasznosabb program, melyet a ➲ *www.fs-driver.org/download.html* oldalról lehet letölteni, legújabb elérhető verziója *1_10c* nevet viseli. Miután feltelepítettük a programot, megkérdezi tőlünk, hogy melyik betűjelhez csatolja a *Linux* partíciókat, ez a legfontosabb pontja a telepítésnek, mert ha nem állítjuk be a kívánt betűjeleket, nem fogjuk látni a partíciókat.
Akár a *Swap* partíciót is lehetőségünk van felcsatolni, de felesleges, mert *Windows* rendszerünk továbbra is formázatlan lemezként látja.
Ez a program annyiban különbözik az előbb bemutatottaktól, hogy nem kell indítani, beépül operációs rendszerünkbe, tehát ne keressük a *Start* menüben. Az *Intézőben* megjelenik Linux rendszerünk betűjele, felismeri a fájlrendszert, olvashatjuk és írhatjuk is azt. Olyan érzés, mintha a *Windows* mindig is tudta volna ezt lehetőséget. Nekem ez a program nyerte meg legjobban a tetszésemet, mert nincs alkalmazáshoz kötve, ha például csak *Intézőben* szeretünk dolgozni, azzal is tökéletes, mint bármi mással is.



5. ábra EXT2 telepítés közben



6. ábra Intézőben Linuxunk…

Végszó

Ha csak böngésszük a Linux fájlrendszerünket, kárt nem okozhatunk benne. Más a helyzet azonban, ha írunk rá, vagy esetleg módosítani akarunk fájljainkon. Ilyenkor legyünk nagyon óvatosak, ha valamit változtatunk rajta, utána mindig indítsuk is el a Linux rendszerünket, meggyőződve, hogy károsodás nem érte a fájlrendszert. Jó böngészést kívánok!

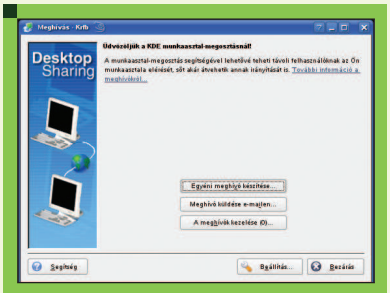
Berei László
(bereil@invitel.hu)
Berei László 22 éves, a GAMF műszaki informatika szak másodéves hallgatója. Ha nem a notebookja előtt ül, akkor vagy a barátaival van, vagy a konditeremben tölti edzéssel az idejét.

KDE alkalmazások (8. rész)

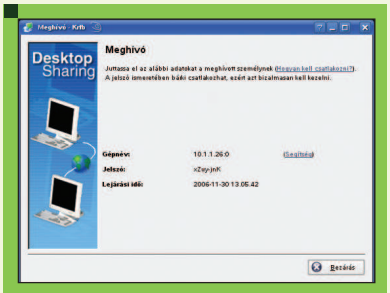
A munkaasztal megosztása, átvétele

A népszerű VNC programcsomag KDE környezetbe integrálva biztosítja, hogy képesek legyünk a munkaasztalunkat megosztani, illetve mások megosztott munkaasztalát átvenni.

Ehhez telepítenünk kell a ***krdc*** és a ***krfb*** programokat, amelyek általában a ***kdenetwork*** csomagban találhatók. Ha itt nem találnánk meg, akkor a disztribúcióban szétkapták a ***kdenetwork*** csomagot több kisebb csomagra, így talán a ***kdenetwork-vnc*** vagy a ***kdenetwork-rfb*** csomagban lesz. Mindenképpen meg kell találunk, mert a 3.1-es verzió óta a ***KDE*** szerves része.



1. ábra Munkaasztal megosztása

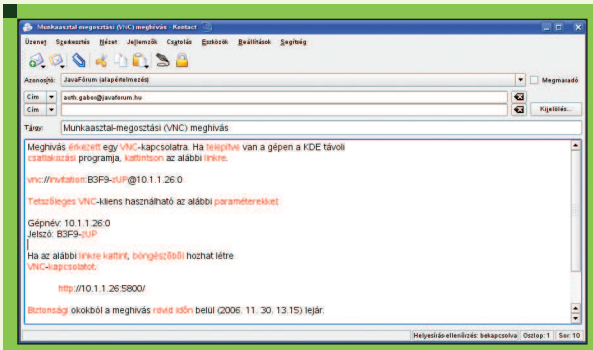


2. ábra A megosztás adatai

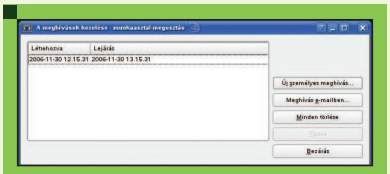
Munkaasztal megosztása

Munkaasztalunk megosztására a ***krfb*** programot használhatjuk, amelyet (***OpenSuSE*** esetén) a ***Rendszer -> Távoli elérés -> Munkaasztal megosztás*** menüpont alatt találunk meg.
A legegyszerűbb módszer egy meghívó készítése, ekkor a program megadja azt a három adatot, amely szükséges a megosztott munkaasztal (mások általi) távoli eléréséhez.
Ha valakit be szeretnénk engedi a gépre, akkor elég megadni a gépnevet és a jelszót, a meghívás élettartama csak nekünk szól: legkésőbb ekkor lehet belépni a megadott jelszóval.
A program külön ablakban jelzi, hogy csatlakoznának a gépünkre, amely adatok között a látogató ***IP*** címét vagy ***DNS*** nevét láthatjuk. A kapcsolatot ekkor még elfogadhatjuk, vagy akár vissza is utasíthatjuk, illetve megakadályozhatjuk, hogy a csatlakozni kívánó felhasználó átvehesse az irányítást az egerünk és a billentyűzetünk felett. Ha kivesszük a pipát (vagy az x jelet), akkor csak a képernyőképünk kerül a csatlakozók gépére, de nem tudnak beavatkozni a gépen történő eseményekbe. Ez utóbbi oktatásnál hasznos, az előbbi pedig rendszergazdáknak okoz kisebb cipőtalpkopást.
Közvetlenül levélben is meghívhatunk felhasználót a gépünkre, ekkor a ***KMail*** hívódik meg, egy kitöltött formalevéllel, amelyben a megfelelő helyen a megfelelő adatokat tartalmazza.
Az élő meghívások kezelése alatt általában a meglévő meghívók törlését szoktuk érteni, hiszen ha a munka

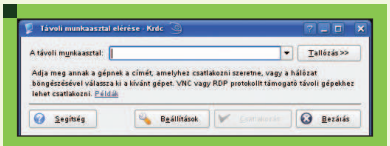
3. ábra Látogató érkezett



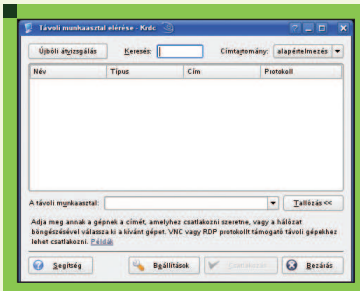
4. ábra Meghívás levélben



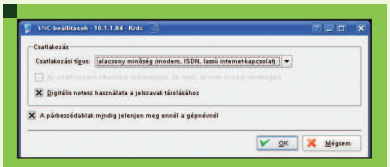
5. ábra Meghívások kezelése



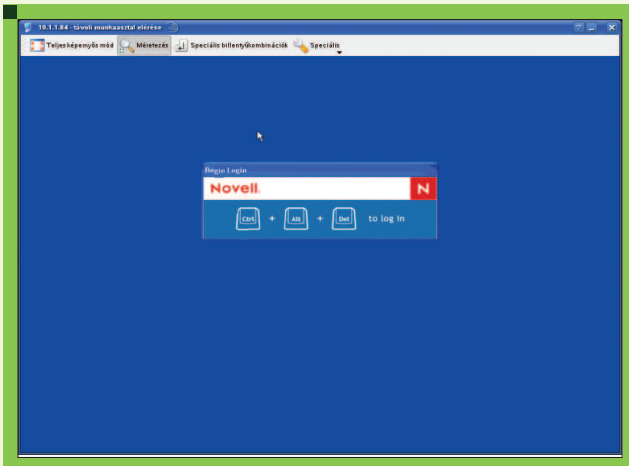
6. ábra Távoli számítógép elérése



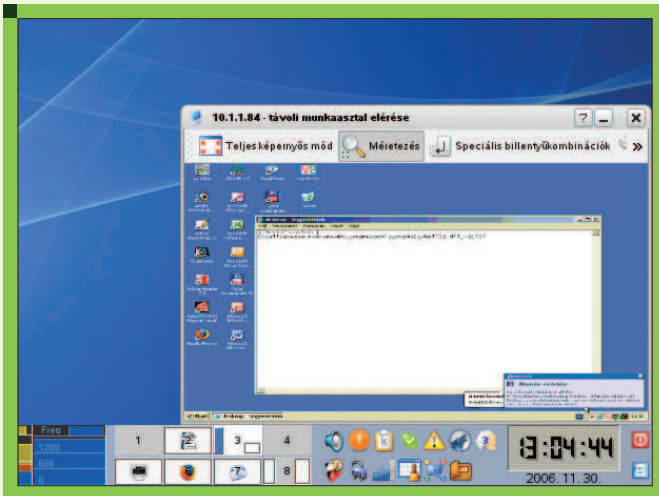
7. ábra Számítógépek keresése



8. ábra Minőség beállítása



9. ábra Egy átvett Windows képernyője



10. ábra Windows a sarokban

elvégeztetett, akkor jobb, ha töröljük az élő meghívót. De természetesen létrehozhatunk újat is.

Távoli munkaasztal elérése

Távoli munkaasztalt a ***krdc*** programmal tudunk elérni, amely (***OpenSuSE*** esetén) a ***Rendszer -> Távoli elérés -> Távoli munkaasztal*** elérése menüpont környékén található.
Ha kaptunk meghívást, akkor elég az ott található gép nevét beírnunk, ha nem kaptunk meghívást, akkor tallózhatunk is a hálózaton, hogy mely gépeken fut ***VNC*** szolgáltatás. Ha megtaláltuk a keresett számítógépet, esetleg tudunk egy közvetlen gépcímet, akkor a program megkérdezi, hogy milyen minőségben szeretnénk a megosztást használni. Ha kis sebességet választunk (ISDN), akkor a kapott kép minősége nagyon gyatra lesz, ellenben élvezhető sebességet kaphatunk ISDN vonalon is.
Ezt követően a kért jelszót kell megadnunk, és már nálunk is van azon gép képernyője, amelyet irányítani szeretnénk. Ha ***Windows***-t futtató gépre jelentkeznénk fel, akkor problémát jelenthet a ***Ctrl-Alt-Del*** kombináció bevitele, amely ***Linux*** esetén újraindulást jelent, ezért erre a speciális billentyűkombinációk menüpontot kell használnunk, különben a gépünk újra fog indulni... :)
A program ezen túl képes teljes képernyős működésre, illetve az ablak méretéhez igazítani a kapott képernyőképet, amely egy kissé elmosódottá teszi a kapott képet, de akár a munkaasztalunk sarkában tudjuk figyelemmel kísérni, hogy mi történik az adott gépeken, ez oktatás esetén jól jön, hogy a tanulók 12-16 gépét táblázatszerűen kiteszem a munkaasztalra, így látom azonnal, hogy ki mivel foglalkozik...

Auth Gábor
(auth.gabor@enaplo.hu)

Egy pécsi középiskolában informatikát és programozást oktat. Tíz éve botlott először a UNIX rendszerekbe, 7 év Linux használat után kapta el a FreeBSD lázat, amiből máig nem tudott kigyógyulni.

KAPCSOLÓDÓ CÍMEK

KDE projekt oldala:
➲ http://www.kde.org

Házi telefonrendszer Skype kiszolgálóval

Spórolni tudni kell! Skype kiszolgálóval hagyományos telefonkészüléken a hét minden napján, napi 24 órás telefonszolgáltatást biztosíthatunk otthonunkban.

A *Skype*-nak igen bosszantó az a tulajdonsága, hogy telefonhívások kezdeményezéséhez és fogadásához futnia kell egy számítógépen. Tehát, ha gépünk ki van kapcsolva, a *Skype* sem működik. Sőt, ha gépünk megy is éjjel-nappal, a *Skype* teljesítménye (hívások minősége, megbízhatóság, stb.) változik attól függően, hogy egyéb folyamatok mennyire fosztják meg a számára szükséges erőforrásoktól.

Az én megoldásom erre az volt, hogy építettem egy *Skype* kiszolgálót, amely napi 24 órás telefonszolgáltatást biztosít nagyon kevés ráfordítással.

Ha szakítunk a hagyományos telefonszolgáltatónkkal és magunk vesszük kezünkbe az ügyet egy *Skype* kiszolgáló segítségével, nemcsak közel ugyanolyan – sőt, néha jobb – minőségű telefonrendszerre teszünk szert, de rengeteget meg is spórolhatunk vele. Jómagam megközelítőleg 700 dollárt takarítok így meg évente (idén, jövőre és azután ...), vagy másképp: régi telefonszámlámnak körülbelül 82%-át.

Ha bekötünk egy *Skype* kiszolgálót otthonunk már meglevő rézvezetékes telefonhálózatába, a kagylót felemelve hallhatjuk a szokásos tárcsahangot. *Skype* felhasználóktól vagy hagyományos telefonvonalról érkező bejövő hívásoknál a készülékek ugyanúgy csörögnek az egész lakásban. Tehát kevés ráfordítással a *Skype*-ból egy hagyományosként viselkedő telefont varázsolhatunk.

A *Skype* kiszolgáló építéséhez három út áll előttünk: veszünk egy új számítógépet, összerakunk egy új gépet, vagy a szekrényünk mélyén lapulót alakítjuk át. A cikkben leírom, hogyan építhetünk szinte a semmiből egy *Skype* kiszolgálóként működő gépet. Bármelyik utat is választjuk, a beállítások ugyanazok lesznek, és a cikkben ismertetem őket.

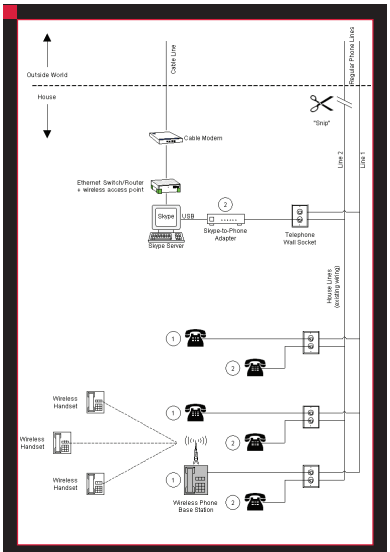
A *Skype* nem egy mindent vagy semmit dolog, mivel összeilleszthető a már meglevő telefonrendszerünkkel, és a kettő együtt működtethető. Így kényelmesen bonyolíthatunk helyi hívásokat, miközben az ingyenes *Skype*-ok közötti hívások lehetősége is fennáll, továbbá a *Skype*-pal nemzetközi beszélgetést is kezdeményezhetünk igen kedvező áron. A beállításokat, amelyeket majd az *1. ábrán* láthatóhoz hasonlóan kell megoldani, a cikkből megismerhetjük. Ha megtartjuk az egyik hagyományos vonalunkat, áthidalhatjuk a segélyhívó számok hívásával kapcsolatos problémákat, megmarad a faxunk és a riasztórendszerünk is (persze lehetőleg azt a vonalat tartsuk meg, amelyiken a riasztó működik).

Az *1. ábrán* egy egészen egyszerű konfigurációt láthatunk. Futtathatunk egyszerre több *Skype*-ot is *Linuxon*, így több telefonvonalunk is lehet. Ez azonban már egy másik cikk témája lenne...

Építsünk Skype kiszolgálót

Akár új gépet veszünk a célra, akár meglevő alkatrészeinkből szereljük össze, figyelembe kell vennünk a következő szoftver- és hardverkritériumokat:

- *Fedora Core 3* (A *Skype* támogatja még a *SUSE 9*-et, a *Mandriva 10.1*-et és a *Debian 3*-at, vagy ezek újabb változatait. *Linuxon* viszont a *Skype* kiegészítő hardvereszközök támogatása jelenleg roppant korlátozott. A cikkben is szereplő *SkypeMate* program miatt csak a *Fedora Core 3* jöhet szóba.)
- 400 MHz processzor.
- 128 MB RAM.
- 10 MB szabad lemezterület.
- *OSS*-kompatibilis hangeszköz (vagy *ALSA* annak *OSS*-kompatibilitási rétegével).
- Szélessávú internetkapcsolat.



■ *1. ábra* Hagyományos telefonvonal helyettesítése Skype kiszolgálóval

1. táblázat *Egy vadonatúj Skype kiszolgáló (hozzávetőleges) költségei*

Alkatrész	Árak (US)
IN-WIN BT610P.180BFU2 Black steelMicroATX ház, 180W tápegység	$39.99
BIOSTAR M7VIG400 MicroATX alaplap AMD Duron 800-as mobil CPU	$69.00
OCZ value sorozatú 512MB (2 x 256MB) 184-pin unbuffered PC 2700 DDR SDRAM	$43.75
10GB HDD (használt darab)	Ingyenes
Skype USB adapter (*2. ábra*)	$43.90
Linux operációs rendszer	Ingyenes
Összesen	$196.64

Azt különösen vegyük figyelembe, hogy ezek egyetlen telefonvonal minimális hardverkövetelményei. Ha arányosan növeljük őket a tervezett ***Skype*** kiszolgálós vonalak számának megfelelően, nem biztos, hogy jól járunk, hiszen nem valószínű, hogy ekkora fejlesztésbe akarnánk kezdeni. A ***Skype*** hihetetlen gyorsan fejlődik, és minden kiadás újabb és újabb funkciókat hoz magával. Az új funkciók

■ *2. ábra* Egy szokványos USB Skype adapter

pedig nyilvánvalóan a hardverigények növekedésével járnak. Saját ***Skype*** kiszolgálómnak egy új gépet terveztem venni, amely mind méretét, mind pedig fogyasztását tekintve kicsi. (Gondoljunk bele, az év 365 napján napi 24 órán keresztül be lesz kapcsolva.) A műszaki leírást, illetve a költségeket az *1. táblázatban* foglaltam össze. Ne feledjük, a ***Skype*** kiszolgáló egeret, billentyűzetet, monitort, ***CD-ROM***-ot vagy floppyt csak a beállításokhoz igényel, egyébként nem. A ***Skype*** kiszolgáló építését nyilván az alkatrészek összerakásával kezdjük. A gyakorlati részleteket itt a cikkben nem tárgyalom, az interneten rengeteg segítséget találunk ehhez. Ott van például a ***PCMechanic***, ahol egy gép összeszerelését lépésről-lépésre bemutatják.

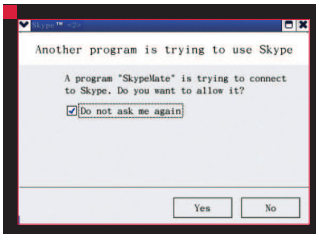
A Linux beállítása a Skype-hoz

Feltételezem, hogy a ***Linux Journal*** (és a ***Linuxvilág***) olvasóinak nem okoz gondot a ***Fedora Core 3*** telepítése és futtatása a ***Skype*** kiszolgálón. Azt viszont jó tudni, hogy a ***Skype*** egy ***Qt*** alkalmazás (bár van belőle a ***Qt***-val statikusan kapcsolt változat is) és a ***Skype API***-ja ***D-BUS***-t használ. A képernyőkímélőt kapcsoljuk ki (végül is úgysem kell semmilyen képernyőt kímélni...) és tiltsuk le a készenléti állapotba váltást, nehogy zavarják a ***Skype***-ot. Az alábbi útmutató lépésről lépésre ismerteti, hogyan állítsuk be ***Linuxot*** a ***Skype***-hoz (feltételezzük, hogy be van már állítva ***skype*** néven egy linuxos felhasználói fiók).

1. A ***Skype*** kiszolgálót kapcsoljuk ki, majd egy ***USB*** kábellel kössük össze a ***Skype*** adaptert a kiszolgálónkkal, és próbaképpen a ***TEL*** feliratú csatlakozójába dugjunk egy hagyományos telefonkészüléket.
2. Kapcsoljuk be a kiszolgálót, és a ***Linuxon*** jelentkezzünk be ***skype*** néven.
3. Töltsük le és telepítsük a *„**Skype Linuxhoz**”* csomagot. Ha nem ***RPM*** csomagból telepítjük, a */etc/dbus-1/system.d/skype.conf* fájlt manuálisan kell létrehoznunk:

```
<!DOCTYPE busconfig PUBLIC "-
➥//freedesktop//DTD D-BUS Bus
➥Configuration 1.0//EN"
 "http://www.freedesktop.org/
 ➥standards/dbus/1.0/
 ➥busconfig.dtd">
<busconfig>
  <!- skype.conf ->
  <policy context="default">
    <allow own=
    ➥"com.Skype.API"/>
    <allow send_destination=
    ➥"com.Skype.API"/>
    <allow receive_sender=
    ➥"com.Skype.API"/>
    <allow send_path="/com/
    ➥Skype"/>
  </policy>
</busconfig>
```

4. Indítsuk el a ***Skype***-ot és jelentkezzünk be. A következő két lépés (5. és 6.) a ***Skype*** beállítása.
5. Győződjünk meg róla, hogy a ***Skype*** automatikusan indul (a ***Skype->Tools->Options->Privacy*** menüpontban jelöljük be a *„**Remember my password**”* melletti jelölőnégyzetet.).
6. Mivel napi 24 órás ***Skype*** telefonszolgáltatást tervezünk, státusunknak mindig ***Online***-nak kell lennie, hogy a többi ***Skype***-használó elérjen bennünket. A ***Skype->Tools->Options->General*** menüpontban a *„**Show me as “Away” when inactive for ... minutes**”*-et állítsuk nullára. (Állapotom legyen „Távol”, ha ...



■ *3. ábra* Engedélyezzük, hogy a SkypeMate az API-ján keresztül kapcsolódjon a Skype-hoz

percig nem nyúlok a géphez.)
A *„Show me as "Not Available" when inactive for ... minutes"*-et szintén nullára állítsuk be. (*Állapotom legyen „Nem elérhető", ha ... percig nem nyúlok a géphez.*)
A nulla ezekben az estekben a végtelenséget jelöli.

7. Váltsunk át rendszergazdai módba a `su` paranccsal és a *root* jelszóval.
8. Töltsük le az *install-SkypeMate.zip*-et és csomagoljuk ki, hogy kinyerjük az *install-SkypeMate* fájlt. (*Skype* adapterünknek kompatibilisnek kel lennie a *SkypeMate* programmal – ellenőrizzük vásárlás előtt!)
9. Változtassuk meg az *install-SkypeMate* jogosultságait, hogy futtatható legyen (a `skype@fc3:~$` a prompt, a többi a begépelendő parancs):

```
skype@fc3:~$ chmod +x install
➥-SkypeMate
```

10. Futtassuk a *SkypeMate* telepítőprogramját:

```
skype@fc3:~$ ./install
➥-SkypeMate
```

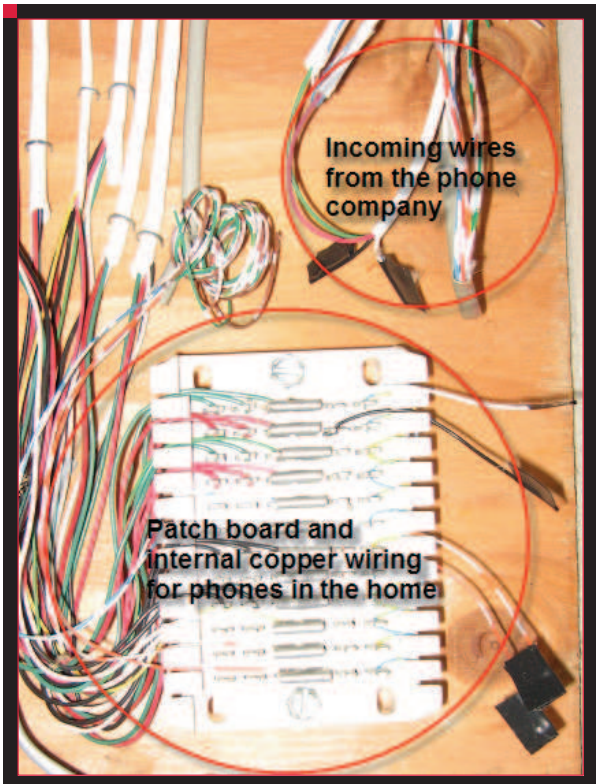
11. Váltsunk vissza normál felhasználói módba és indítsuk újra a *Skype* kiszolgálót. Ismét *skype*-ként jelentkezzünk be.
12. Kattintsunk duplán a *SkypeMate* ikonra (ez a */usr/bin/SkypeMate*-re mutat). Felbukkan egy ablak, amely engedélyt kér, hogy a *SkypeMate* az *API*-ján keresztül kapcsolódjon a *Skype*-hoz (*3. ábra*). Jelöljük be a *„Do not ask me again"* melletti jelölőnégyzetet, majd kattintsunk a *Yes* gombra (ily módon nem kell többször engedélyezni).
13. A hívások hangeszközeként az *USB Skype* adaptert jelöljük meg. (A *Skype->Tools->Options->Hand/Headsets* menüpontban az *„Audio Devices"* alatt válasszuk ki a megfelelő eszközt a legördülő listából.)
14. Kényelmi okokból esetleg szeretnénk gyorshívószámokat beállítani a névjegyzékünkhöz (contact list), hogy mondjuk egy egyszerű 10# tárcsázással máris egyik barátunkat hívhassuk.
15. Teszteljük le a *Skype*-ot az *echo123*-mal, a hívásellenőrző szolgáltatással.

Ha vezetékes- vagy mobiltelefonokat szeretnénk hívni, fel kell töltenünk a SkypeOut egyenlegünket. Ha fogadni akarunk hívásokat ezekről, egy SkypeIn számot kell vásárolnunk. Mindkét szolgáltatás a Skype alacsony tarifáival működik.

Skype kiszolgálónk telepítése

Hol és hogyan kell egy *Skype* kiszolgálót tárolni? Egy *Skype* kiszolgáló akkor érzi jól magát, ha van energiaellátása, szellőzési lehetősége, internetkapcsolata, hozzáférése a hagyományos telefonvezetékekhez

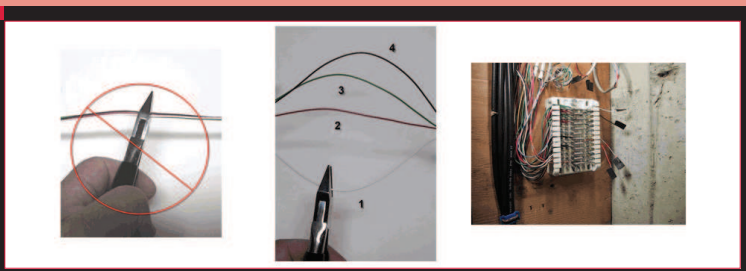
■ *4. ábra* Napi 24 órás telefonszolgáltatást biztosító Skype kiszolgáló. Maga a gép a fénykép felső részén látható, a kábelmodem és a vezeték nélküli router balra, a házi telefonvezetékek kapcsolótáblája pedig jobbra. A kép közepét a villanyóraszekrény foglalja el.



■ *5. ábra* Telefonvezeték-kapcsolótábla

Óvatosan vágjuk el a telefonvezetékeket!

A bejövő telefonvezetékek biztonságos elvágásához szigetelt markolatú vezetékvágó fogóra (vagy nagyon éles ollóra) és szigetelőszalagra van szükségünk. És tartsuk be a vezetékvágás műveletének alapvető szabályait (*6. ábra*). Telefonvezetékeinket a telefonszolgáltató látja el árammal, ez az oka annak, hogy áramszünet idején is tudunk telefonálni. Viszont, ha a telefonszolgáltató ki is kapcsolt minket a hálózatból, attól még a vezetékek áram alatt lehetnek. Tehát legyünk nagyon óvatosak, nehogy rövidre zárjuk őket a vágáskor. Még egy pillanatnyi rövidzárlat is problémás lehet. Ilyesmi akkor történik, ha két vagy több vezetéket vágunk el egyszerre (*1. lépés*). A megoldás kézenfekvő: egyszerre csak egy vezetéket vágjunk el (*2. lépés*). A vezetékek szabadon levő végeit tekerjük be szigetelőszalaggal (*3. lépés*).
Ha esetleg ódzkodnánk az efféle tevékenységtől, akár a telefontársaság szakemberét is megbízhatjuk a feladattal. Még ha ez némi kiadásunkba is kerül, a ***Skype***-pal megspórolt összeg rövid időn belül kárpótol minket.



6. ábra A telefonvezetékek kettévágásának lépései: 1) Ne egyszerre vágjuk el őket, 2) hanem egyesével. 3) A végeket menet közben szigeteljük le.

(***RJ-11***-es csatlakozó), és nincs szem előtt (vagy láb alatt). Én a pincét választottam (*4. ábra*), ami valószínűleg ideális környezet, de mások lehetőségei ennél korlátozottabbak is lehetnek. Ha ez utóbbi csoportba tartozunk, az még egy ok arra, hogy jól megfontoljuk a döntést a ***Skype*** kiszolgáló helyét illetően.
Az alábbi útmutató lépésről lépésre ismerteti, hogyan telepítsünk otthonunkban ***Skype*** kiszolgálót:

1. Szüntessük meg az egyik hagyományos telefonvonalunkat (ne azt, amelyiken a riasztó működik).
2. Vágjuk el a megszüntetett vonal bejövő vezetékét.
3. Kössük össze a ***Skype*** adaptert (TEL csatlakozó) a megszüntetett vonal összes telefonkészülékével (a falon levő csatlakozó). A célra hagyományos telefonvezetéket használjunk, melynek mindkét végén egy ***RJ-11***-es csatlakozó van.
4. Teszteljük újra a ***Skype***-ot a megszüntetett vonalra bekötött telefonkészülékekkel.
5. Kapcsoljuk ki a gépet, és pakoljuk ki belőle azokat a kölcsönvett alkatrészeket (***CD-ROM***, floppy meghajtó), amelyekre csak a beállításhoz volt szükség, de a működéshez nem kellenek.
6. Helyezzük el a gépet a végleges helyén. Dugjuk bele a kábeleket és csatlakozókat, majd kapcsoljuk be.
7. Jelentkezzünk be és teszteljük a ***Skype***-ot még egyszer.
8. Távolítsuk el a monitort, egeret, billentyűzetet.
9. Ha minden rendben ment, most már van nonstop telefonszolgáltatásunk, amit kizárólagosan a ***Skype*** működtet.

Azok a kedves olvasók, akiknek a biztonság a mindenük, és nem bíznak a ***Skype***-ban mint alkalmazásban, a ***Skype*** kiszolgáló használata további előnyökkel jár. Ha a ***Skype*** kiszolgálót a tűzfalon kívülre helyezzük, biztosak lehetünk abban, hogy ha egy illetéktelen behatoló bejut a kiszolgálónkig és valahogy megrongálja a ***Skype***-unkat, azután, mivel a kiszolgáló felülete nem több, mint a tűzfalunkon belülre jövő rézvezeték, bármilyen károsodás a tűzfalon kívüli ***Skype*** kiszolgálóra fog korlátozódni. Legrosszabb esetben is csak a kiszolgáló újbóli telepítése szükséges, és felülvizsgálata biztonsági szempontból, hogy máskor képes legyen megvédeni magát. Ha a tűzfalunk elég korlátozó ahhoz, hogy azon belül a ***Skype*** ne működjön, csak kívülre helyezve használható ki a ***Skype*** telefonszolgáltatás minden előnye.

Mennyit nyerünk egy Skype kiszolgálón?

A telefonszámlák és az adók közös tulajdonsága, hogy mindkettő zsibbasztóan bonyolult, sokat visznek el és keveset hoznak a konyhára. A ***Skype***-pal megspórolt összeg kiszámolására írtam egy táblázatkezelő programot, amely leegyszerűsíti a számolást. Letölthető az ***Elpis*** weboldaláról (lásd a forrásokat). Az elszámolásnál figyelembe kell venni a ***Skype*** kiszolgáló megépítésének és a működtetésének költségeit. A működtetési költség attól függ, milyen gépet választunk a célra. Egy régi gép elég sokat fogyaszthat, így hosszútávon egy új géppel jobban járunk. Osszunkszorozzunk a nagy döntés előtt. Számoljuk ki, milyen költséggel jár, ha ***Skype*** kiszolgálónk az év minden napján 24 órán keresztül be van kapcsolva. Monitor és más energiazabáló perifériák nélkül egy kis modern gép fogyasztása 50 és 100 W között van. Ha a villany kWh-ként 0,1 dollárba kerül, éves költsége 44-88 dollár. Ezek csak megközelítő értékek, ne vegyük szentírásnak. Azt viszont kitűnően jelzik, hogy a 24 órás Skype-működtetés nem ró ránk jelentős anyagi terheket. (Költség = felhasznált energia kW-ban x 24 óra x a kWh egységára, ahol az 50W energiafogyasztás 0,05 kW illetve a 100 W = 0,1 kW.)

Linux Journal 2006., 141. szám

Andrew Sheppard
shep@elpispublishing.com

KAPCSOLÓDÓ CÍMEK

A cikk forrása:
➲ http://www.linxjournal.com/article/8644

Hívófél-azonosítás Asterisk és Ajax segítségével

Asteriskkel és Ajaxszal egyszerűen megjeleníthetjük a bejövő és kimenő telefonhívásaink információit.

Asterisket használok a telefonhívásaim kezelésére már körülbelül egy éve. Eközben jöttem rá, hogy nagyon sok érdekes dolgot meg lehet csinálni az *Asterisk*, a *VoIP* és egyéb technológiák bevetésével. Az egyik ilyen trükkös dolog például a bejövő hívások telefonszámának valós idejű kiíratása a webböngészőmben. Ehhez nem kell más, csupán *Asterisk*, *Perl*, *CGI*, *HTML*, *CSS*, *SQL*, *XML* és *aszinkron JavaScript*, vagy röviden *Ajax*. Sok különböző alkotóelemet kell összehozni, de sokszor ez teszi érdekessé a megoldandó feladatot.

A programom működése dióhéjban: ha valaki keres bennünket, akkor az *Asterisk* megvárja a hívófél telefonszámát. Ezután az információ a */tmp*-be kerül pár egyéb információ társaságában. Mindezt az *Asterisk* végzi a *tárcsázási szabályok* (*dial plan*) alapján. A böngészőben megnyitott weboldal másodpercenként lefuttat egy *JavaScript* programot.

Ez *XMLHttpRequest* objektumként lekéri az esetleges új hívások információit. Ezt a szerveren futó *CGI szkript* küldi el *XML* formátumban.

A *JavaScript* feldolgozza a kapott eredményt és megjeleníti. Létrehoztam egy *CSS* állományt is, hogy a feldolgozott információ úgy nézzen ki a megnyitott weboldalon, mint egy *vizuális emlékeztető* (*sticky note*).

Ha vége a hívásnak, az *Asterisk* szerver *SQL* adatbázisba lementi a *hívásrekordot* (*CDR – Call Detail Record*). A *CGI* szkript minden kliens oldali lekérdezéskor ellenőrzi az adott híváshoz tartozó hívásrekord meglétét. Ha létezik, akkor befejeződött a hívás és törli az információs állományt a */tmp*-ből. Ennek megfelelően a vizuális emlékeztető is eltűnik, ha a hívó fél letette a kagylót.

Ráadásul a program egyszerre négy hívást képes kezelni és a kimenő hívásokat is jelzi. Jópofa dolog látni, ki van a vonal túlsó végén – legyen az hívó vagy hívott. Mindezt anélkül, hogy az aktuális hívást megszakítva meg kellene kérdezni. Ha a fiaim idősebbek lesznek, ez még jól jöhet.

Ahhoz, hogy működjön a dolog, az *Asterisk*-et megfelelően be kell állítani, hogy a kapott információkat *SQL* adatbázisba mentse le. Alap beállításokkal az *Asterisk* vesszővel elválasztva adja át az információkat.

A probléma az, hogy az egyszerű szöveges hívásrekord nem tartalmazza a hívás egyedi azonosítóját, amellyel a hívás befejezését lehetne észlelni. Az *SQL*-ben tárolt adatsor azonban tartalmazza. Ez azonban nem lehet probléma. Emlékeim szerint az *Asterisk* hívásinformációk *Postgres* adatbázisba mentése elég egyszerű és jól dokumentált (`cdr_pgsql.conf`). Természetesen használhatunk *MySQL*-t vagy *ODBC*-t is.

Az első és legkönnyebb rész: módosítjuk az *Asterisk* tárcsázási szabályát, hogy létrehozzon egy egyszerű fájlt a bejövő és kimenő hívások esetén. Csupán egy sort kell beszúrni, ha megtaláltuk, hogy hova is kell (az alábbi részletet egy sorba írjuk!):

```
exten => s, n, system(echo
➥"IN#${CALLERID(name)}
➥#${CALLERID(number)}#${UNIQUE
➥ID}" >/tmp/panels/
➥cid/${UNIQUEID})
```

Ez a sor létrehozza egy állományt a */tmp/panels/cid* könyvtárban négy mezővel, a *#*-et használva elválasztónak. Természetesen létre kell hozni a */tmp/panels/cid* könyvtárat megfelelő jogosultságokkal, hogy *Asterisk* tudja írni, a *CGI* szkript pedig olvasni és törölni.

Az első mező a *IN* vagy *OUT*, annak megfelelően, hogy bejövő vagy kimenő hívásról van szó. A következő két mező a hívó nevét és telefonszámát tartalmazza, míg az utolsó a hívás egyedi azonosítóját. Ezt a sort a tárcsázási szabályrendszer megfelelő helyére kell beszúrni. Oda ahol már megkaptuk a hívófél telefonszámát, de még nem csörgetjük a telefont.

Ha szeretnénk a ugyanezt megoldani a kimenő hívásokra is, használjuk az alábbi sort:

```
exten => s, n, system(echo
➥"OUT##${EXTEN}#${UNIQUEID}"
➥> /tmp/panels/cid/$
➥{UNIQUEID})
```

Kimenő hívás esetén nyilván nem kapjuk meg a hívó fél számát, így a második mező üres. Tudjuk viszont a hívott számot, melyet az `${EXTEN}` változó tartalmaz a harmadik mezőben.

Mindkét esetben biztosnak kell lennünk, hogy a mellék és a prioritás mezők frissültek (példánkban s és n).

A lényegre koncentrálva lecsupaszítottam a weboldalt az alapokig, ahogy az az első listán látható. Noha egyszerűnek látszik a *HTML* oldal, mégis minden lényeges benne van. Először is betölti a `cid.js` állományban található

1. Lista Leegyszerűsített weboldal

```
<html>
<head>
  <title>CID Test</title>
  <script language=javascript
➥src=http://hostname/cid.js>
  </script>
  <style type="text/css">
    @import "cid.css";
  </style>
</head>
<body>
  <div id="phone1"></div>
  <div id="phone2"></div>
  <div id="phone3"></div>
  <div id="phone4"></div>
  <script>
    start_cid();
  </script>
  Your Content Would Go Here.
</body>
</html>
```

JavaScript forráskódot. Majd az oldalhoz tartozó stíluslapot a `cid.css` fájlból. A stíluslappal rugalmasan konfigurálhatjuk a vizuális emlékeztetők megjelenését. Utána létrehozunk négy `div` részt, `phone1`...`phone4` nevekkel. Ezek a szekciók a hívásinformációk megjelenítésekor válnak majd láthatóvá. Végül az oldal időnként meghívja a `start_cid()` eljárást. Erre később térünk ki.

Habár a ***CSS*** tudásom nem tökéletes, mellékeltem egy példát, amivel már lehet kísérletezni. (*2. Lista*).

Ez a ***CSS*** fájl tovább ésszerűsíthető a közös tulajdonságok összevonásával, ezt azonban az Olvasóra bízom. A stíluslap négy egyenlő távolságra elhelyezett sárga színű vizuális emlékeztetőt eredményez egy kis árnyékkal kiegészítve (*1. ábra*).

Most pedig lássuk a ***CGI*** szkriptünket (*3. Lista*).

Ez a ***Perl*** szkript a ***/tmp/panels/cid*** könyvtárban keres állományokat, természetesen figyelmen kívül hagyva a `.` és `..` bejegyzéseket. Minden egyéb fájlt megnyit és beolvas. A végeredmény egy ***XML*** formátumú állomány lesz, ahogy az a *4. Listán* is látszik.

2. Lista Példa stíluslap (cid.css)

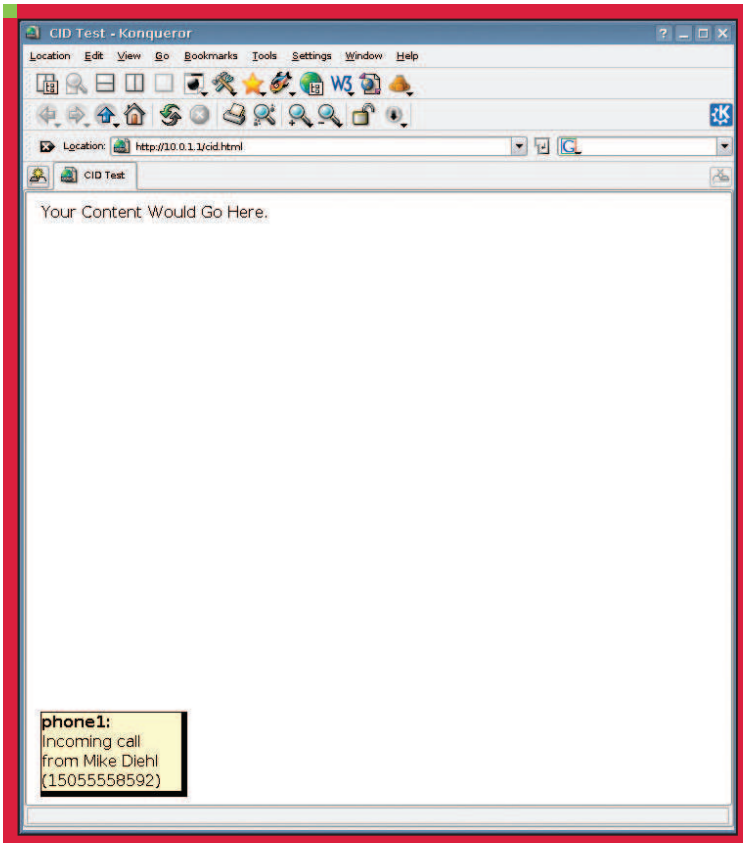
```
div#phone1{
  background: #FFFFCC;
  display: none;
  position: absolute;
  border-top: thin solid
  ➥black;
  border-left: thin solid
  ➥black;
  border-right: 6px solid
  ➥black;
  border-bottom: 6px solid
  ➥black;
  top: 85%;
  left: 2%;
  width: 20%;
  height: 5em;
}
div#phone2{
  background: #FFFFCC;
  display: none;
  position: absolute;
  border-top: thin solid
  ➥black;
  border-left: thin solid
  ➥black;
  border-right: 6px solid
  ➥black;
  border-bottom: 6px solid
  ➥black;
  top: 85%;
  left: 27%;
  width: 20%;
  height: 5em;
}
div#phone3{
  background: #FFFFCC;
  display: none;
  position: absolute;
  border-top: thin solid
  ➥black;
  border-left: thin solid
  ➥black;
  border-right: 6px solid
  ➥black;
  border-bottom: 6px solid
  ➥black;
  top: 85%;
  left: 52%;
  width: 20%;
  height: 5em;
}
div#phone4{
  background: #FFFFCC;
  display: none;
  position: absolute;
  border-top: thin solid
  ➥black;
  border-left: thin solid
  ➥black;
  border-right: 6px solid
  ➥black;
  border-bottom: 6px solid
  ➥black;
  top: 85%;
  left: 77%;
  width: 20%;
  height: 5em;
}
```

3. Lista CGI szkript forrása

```
#!/usr/bin/perl
use DBI;
$dbh = DBI->connect("dbi:Pg:
➥dbname=database",
➥"postgres", "password")
  || die "Can't connect to
  ➥database.\n";
print "Content-type:
➥text/xml\n\n\n";
print "<panels>\n";
check_cid("/tmp/panels/cid");
print "</panels>\n";
exit;
sub  check_cid {
  my($dir) = @_;
  my(@a, $a, $file, $count,
  ➥$top);
  local(*FILE, *DIR);
  opendir DIR, "/tmp/panels/
  ➥cid";
  while ($file = readdir(DIR)) {
    if ($file eq ".") {
    ➥next; }
    if ($file eq "..") {
    ➥next; }
    open FILE, "/tmp/panels/
    ➥cid/$file";
    chomp($line = <FILE>);
    close FILE;
    ($dir, $name, $number,
    ➥$uid) = split("#",
    ➥$line);
```

3. Lista folytatás

```
    $count++;
    if ($dir eq "IN") {
      $html = "Incoming call
      ➥from $name ($number)";
    } else {
      $html = "Outgoing call
      ➥from $name ($number)";
    }
    expire_call($uid);
    print <<EOF
<panel>
  <name>phone$count</name>
  <content>$html</content>
</panel>
EOF
;
  }
}
sub  expire_call {
  my($id) = @_;
  my($sth, $count);
  $sth = $dbh->prepare("select
  ➥count(*) from cdr where
uniqueid=\'$id\'");
  $sth->execute();
  ($count) = $sth->
  ➥fetchrow_array();
  if ($count) {
    unlink("/tmp/panels/
    ➥cid/$id");
  }
}
```



1. ábra A bejövő hívás a weboldalon vizuális emlékeztetőként

4. Lista A CGI szkript által legenerált XML állomány

```
<panels>
<panel>
  <name>phone1</name>
  <content>Incoming call from
  ➥Mike Diehl
(15055558592)</content>
</panel>
</panels>
```

Természetesen az ***XML*** állomány maximum négy `<panel>` blokkot tartalmazhat `phone1`-től `phone4`-ig. A `<content>` blokk a megjeleníteni kívánt szöveget tartalmazza. Minthogy nem egyszerű dolog ***HTML*** kódot XML-be ágyazni, így a `<content>` blokk formázásával nem foglalkoztam túlzottan. Megfigyelhető továbbá a bejövő és kimenő hívások különálló kezelése.

Ha az ***XML*** adott részének legenerálással végeztünk és elküldtük a kliensnek, meghívjuk a `expire_call()` függvényt is. Ezzel leellenőrizzük a hívásrekord meglétét az adatbázisban, hogy véget ért-e a hívás? Az ***Asterisk*** bejegyzi, így ha találatot kapunk, a hívás befejeződött és törölhetjük a fájlt a */tmp/panels/cid* könyvtárból.

A ***JavaScript*** komponens végzi a munka oroszlánrészét és talán ezt a legnehezebb megérteni is. (*5. Lista*). Korábban már volt szó róla, hogy a rendszert a `start_cid()` indításával érjük el. Az eljárás lényege csupán annyi, hogy másodpercenként meghívja a `update_cid()` eljárást. Az `update_cid()` meghívja a `get_from_server()` eljárást, amely böngészőfüggetlen módon kap egy ***XMLHttpRequest*** objektumot. Ezt fogjuk később feldolgozni. Később az `update_cid` meghívja a `clear_panels()` eljárást, mely minden vizuális emlékeztetőt kezdetben üressé tesz és elrejt.

5. Lista JavaScript komponens

```
function start_cid () {
  setInterval("update_cid()",
  ➥1000);
}
function update_cid () {
  var req;
  var xml;
  var panels;
  var count;
  var name;
  var div;
  req = get_from_server();
  clear_panels();
  xml =
req.responseXML.getElementsBy
➥TagName("panels")[0];
  panels = xml.getElementsBy
  ➥TagName("panel");
  for (count=0 ; count <
  ➥panels.length ; count++) {
    panel = panels[count];
    name = panel.getElementsBy
    ➥TagName("name")[0];
    name = name.firstChild.
    ➥nodeValue;
    content = panel.get
    ➥ElementsByTagName
    ➥("content")[0];
    content = content.first
    ➥Child.nodeValue;
    div = document.getElement
    ➥ById(name);
    div.style.display="block";
    div.innerHTML = "<b>" +
    ➥name + ": </b>" +
    ➥content;
    if (div.innerHTML == "") {
      div.style.display=
      ➥"none";
    }
  }
}
function get_from_server () {
  var req;
  if (window.XMLHttpRequest) {
    req = new
    ➥XMLHttpRequest();
  } else if (window.ActiveX
  ➥Object) {
    req = new ActiveXObject
    ➥("Microsoft.XMLHTTP");
  }
  req.open("GET", "/cgi-bin/
  ➥cid.pl", false);
  req.send(null);
  return req;
}
function clear_panels () {
  for (count=1 ; count < 5 ;
  ➥count++) {
    document.getElement
    ➥ById("phone" +
    ➥count).innerHTML = "";
    document.getElement
    ➥ById("phone" +
    ➥count).style.display=
    ➥"none";
  }
  return;
}
```

Láthatóvá akkor válnak, ha lesz valami tartalmuk.
A program további része kicsit bonyolultabb lesz. Felhasználva a korábbi objektumot és a `getElementsByTagName()` eljárást, kapunk egy *XML* objektumot `<panels>` blokkal.
A `getElementsByTagName()` újabb meghívásával különálló `<panel>` blokkokból álló tömböt kapunk. Végül minden, a tömbben szereplő `<panel>` blokkot feldolgozunk. Minden feldolgozott blokk egy folyamatban lévő hívás. Ennek megfelelően az új hívások vizuális emlékeztetőként jelennek meg. Minden `<panel>` blokk egy `<name>` és egy `<contents>` blokkot tartalmaz, melyet megfelelő változókban tárolunk.
A `getElementById()` eljárással lekérdezzünk a panel névvel egyező `<div>` elem azonosítóját (*ID*). Most már minden szükséges információnk megvan, név, tartalom, hely. Nincs más hátra, mint az adott `<div>` láthatóvá tétele és a tartalom hozzárendelése egyszerűen az `innerHTML` attribútum alkalmazásával. A ciklus végén visszatérünk az eljárás elejére és elölről kezdjük.
A bemutatott (választ kér és megjelenít) eljárás másodpercenként és felhasználói beavatkozás, illetve oldal újratöltés nélkül történik.
A felhasználó ebből csak annyit vesz észre, hogy telefoncsörgéskor megjelenik egy vizuális emlékeztető, ha pedig vége a beszélgetésnek, eltűnik.
Amint látható, jól használható nyelv a ***JavaScript***. Sajnos azonban a böngésző oldali támogatása és a fejlesztőeszközök hiánya érezhető. A program fejlesztése közben gyakran omlott össze a böngészőm. Sokszor pedig a tévedésből ***cache***-elt adatokkal és titokzatos hibaüzenetekkel akadtak problémáim. Miután azonban működésre bírtam, meg kellett győződnöm, hogy az általam használt böngészőkben – ***Konqueror*** és ***Firefox*** – hibátlanul működik. Úgy vélem más böngészők alatt is fut, noha nem teszteltem. Minthogy a szoftverfejlesztéseknél jellemzően ***vi***-t használok, járatlan vagyok az integrált fejlesztő környezetekkel (***IDE – Integrated Development Environments***) kapcsolatban. Ennek ellenére ha hallott az Olvasó olyanról, amely ***JavaScript***-et is támogat, szeretettel várom e-mailben.
Most, hogy a program működik, itt az ideje bővíteni. A legnyilvánvalóbb változtatás egy link lesz, mely lehetővé tenné további információk lekérését a hívóról. Ezt persze a címjegyzékemből vagy egy külön adatbázisból. Talán egy kép is megjelenhetne a hívóról, bár elég hosszú időbe telne családtagoktól, barátoktól, ismerősöktől összegyűjteni a képeket. Jó lenne az is, ha egy gombnyomással üzenetrögzítőre irányíthatnám a bejövő hívást. Nem lenne nehéz megvalósítani azt sem, hogy a rendszer jelezze, ha van új üzenetem vagy a barátaim ráérnek csevegni.
Itt van tehát egy jópofa elfoglaltság, amelyben több különféle eszköz és technika van jelen. Vajon mennyit számítana fel a ***Qwest*** egy ilyen webes felület havi díjaként, ha csak a hívófél azonosítás havi 6 dollár?

Linux Journal 2006., 151. szám

Mike Diehl a Sandia National Laboratories-nál dolgozik és munkaidejében hálózatfelügyelő programot fejleszt. Feleségével és két kisfiával Albuquerque-ben, Új-Mexikóban él. Mike a mdiehl@diehlnet.com email címen érhető el.

A rendszerfelügyelet megkönnyítése wikik és blogok alkalmazásával

A TWiki és a WordPress ismertetésével megmutatjuk, hogy a wikik és a blogok milyen hasznosak lehetnek a rendszerfelügyelet és a dokumentáció során.

A rendszerfelügyelet időnként olyan, mintha egy hajót irányítanánk. A motoroknak egyenletes teljesítménnyel kell működniük, folyamatosan tájékoztatni illetve értesíteni kell a legénységet és a kikötőket, továbbá a kapitány naplóját is vezetni kell. A látóhatáron kell tartani a szemünk, hogy lássuk, mi következik. Az elmúlt évben két olyan technológia is megjelent, amely segíthet tartani az irányt: ez a ***wiki*** és a ***blog***.

A jó dokumentáció

Meglátásom szerint a rendszerfelügyelet egyik legnehezebb része a dokumentáció pontos és naprakész vezetése. Ha leírjuk, hogyan javítottunk egy idegesítő problémát az egyik nap, segíthet felidézni a megoldást hónapokkal később, amikor újra megjelenik. Aki dolgozott már másokkal, bizonyára felismeri, hogy mennyire fontos a jó dokumentáció. Ha valaki egyedüli rendszergazda, akkor is élvezheti a jó dokumentáció előnyeit, különösen, ha egyszer újabb rendszergazda érkezik a fedélzetre.
A jó dokumentációs rendszerekhez többek között az alábbi célokat kell szem előtt tartani:

- Könnyítse meg a fontos adatok keresését a magunk és munkatársaink számára.
- Könnyítse meg a gyors felzárkózást az új alkalmazottak számára.
- Könnyítse meg a dokumentáció létrehozását, szerkesztését és kivonását.
- Kövesse nyomon a változtatásokat, valamint, hogy ki végezte azokat.
- Hitelesítő rendszerrel korlátozza, hogy ki láthatja és szerkesztheti a dokumentációt.

Sajnos a dokumentáció folyamatos frissítése önmagában kitöltheti a teljes munkaidőt. Igaz,hogy a dokumentálás nem valami felemelő feladat, hosszú távon mégis kifizetődő.

Miért éppen wiki?

Itt jön a képbe a ***wiki***. Idézet a ***Wikipédiából***: ***„a wiki olyan webhely, amely lehetővé teszi, hogy a felhasználók tartalmat adjanak hozzá és szerkesszék azokat, továbbá különösen alkalmas a tevékeny, együttműködő alkotásra.”***
Ez azt jelenti, hogy a ***wiki*** segítségével egy központi helyen tárolhatjuk és szerkeszthetjük a dokumentációt. Az általunk használt platformtól függetlenül érhetjük el és szerkeszthetjük azt, mindössze egy webböngészőre van szükségünk. Vannak olyan ***wikik***, amelyek képesek nyomon követni a módosított dokumentumok minden egyes változását, így ha téves módosításokat végzünk egy dokumentumon, visszaállíthatjuk az előző változatot. Az új felhasználóknak csupán egyetlen akadályt kell leküzdeniük: meg kell ismerniük az adott ***wiki*** jelölőnyelvét, de időnként még ez sem feltétlenül szükséges.
A ***wiki*** egyik szolgáltatása egyben az egyik hátránya is. A ***wikik*** meglehetősen kötetlenek, és amellett, hogy ez lehetővé teszi, hogy a dokumentáció gyors megírására összpontosítsunk, egyúttal azt is okozhatja, hogy a ***wiki*** szerkesztése folyamatosan kicsúszik a kezünk közül. Alaposan át kell gondolni a ***wiki*** szerkesztését, hogy a témák nehogy elsikkadjanak vagy elvesszenek. Tapasztalataim szerint igen célszerű, ha a főoldalon az összes téma tartalomjegyzéke szerepel. Nem számít, végül hogyan szerkesztjük a ***wikit***, csak arra kell ügyelni, hogy mindenki számára jól érthető legyen. Sőt, az első dokumentum lehetne a ***wiki*** szervezését leíró szabályrendszer.

TWiki

Számos nyílt forráskódú ***wiki*** létezik, például a ***MediaWiki*** és a ***MoinMoin***, amelyek saját elképzeléseket valósítanak meg a jelölés és a külalak terén, de most a ***TWikire*** összpontosítunk. Lássuk a ***TWiki*** néhány előnyét:

- Hálózatok fogalma, ami lehetővé teszi a wikigazda számára, hogy az együttműködés területeit önálló területekké csoportosítsa, amelyek mindegyike saját hitelesítési szabályokkal és témákkal rendelkezik.
- Biztos alapokon álló felhasználói és fejlesztői réteg.

- *RCS* alapú változatkezelés.
- *Perl* alapú – *mod_perl* vagy *FastCGI* alkalmazható.
- A hitelesítés kezelése a *wikin* kívül történik, olyan eljárások segítségével, mint az *Apache htpasswd*.

Jelenleg a legfrissebb stabil változat a *Cairo*, azaz *Twiki20040904*. Ahogy a név is jelzi, 2004. szeptember 4-én adták ki, és nagyon stabilnak bizonyult. Ennek ellenére hiányzik belőle a jelenlegi béta kiadás, a *Dakar* néhány szolgáltatása, amelyeket nagyon hasznosnak találok.
A cikkben használat *Dakar* változat a *TWikiRelease2005x17x783beta*.
A *TWiki* telepítése viszonylag könnyű, de azért dolgozni kell vele. Remélhetőleg a béta változat fejlődésével könnyebbé válik a telepítés és a frissítés, egyúttal a dokumentáció is világosabb lesz.
Először létre kell hoznunk azt a könyvtárat, amelybe a *TWikit* szeretnénk telepíteni, legyen mondjuk */var/www/wiki*. Ezután ebbe a könyvtárba csomagoljuk ki a *TWiki* terjesztés .tar állományát. Következőnek meg kell győződnünk, hogy az a felhasználó, akinek joga van *CGI* parancsfájlokat futtatni (általában *apache* vagy *www-data*), birtokolja az összes fájlt, valamint képes a fájlok írására:

```
# install -d -o apache /var/
# www/wiki
# cd /var/www/wiki
# tar zxf /path/to/TWikiRelease
# 2005x12x17x7873beta.tgz
# cp bin/LocalLib.cfg.txt
# bin/LocalLib.cfg
# vi bin/LocalLib.cfg
# lib/LocalSite.cfg
# chown -R apache *
# chmod -R u+w *
```

Most másoljuk a *bin/LocalLib.cfg.txt*-t a *bin/LocalLib.cfg*-be, majd szerkesszük azt. A `$twikiLibPath` változót módosítani kell, hogy a *TWiki lib* könyvtár abszolút elérési útvonalára mutasson, ami a mi esetünkben a */war/www/wiki/lib*. Ezen kívül létre kell hozni a *lib/LocalSite.cfg*-t, amely a meghatározott webhely adatait tartalmazza. Például az alábbiak kerülhetnek egy *LocalSite.cfg* fájlba:

```
# Ez a Localsite.cfg, amely
# a helyi TWiki
# webhely összes beállítását
# tartalmazza
$cfg{DefaultUrlHost} =
➥"http://www.example.com";
$cfg{ScriptUrlPath} =
➥"/wiki/bin";
$cfg{PubUrlPath} = "/wiki/pub";
$cfg{DataDir} =
➥"/var/www/wiki/data";
$cfg{PubDir} =
➥"/var/www/wiki/pub";
$cfg{TemplateDir} =
➥"/var/www/wiki/templates";
$TWiki::cfg{LocalesDir} =
➥'/var/www/wiki/locale';
```

Íme a *wiki* futását engedélyező *Apache* beállítófájl megfelelő részlete:

```
ScriptAlias /wiki/bin/
➥"/var/www/wiki/bin/"
Alias /wiki "/var/www/
➥localhost/wiki"
<Directory "/var/www/wiki/bin">
    Options +ExecCGI -Indexes
    SetHandler cgi-script
    AllowOverride All
    Allow from all
</Directory>
<Directory "/var/www/wiki/pub">
    Options FollowSymLinks
    ➥+Includes
    AllowOverride None
    Allow from all
</Directory>
<Directory
"/var/www/wiki/data">
    deny from all
</Directory>
<Directory "/var/www/wiki/lib">
    deny from all
</Directory>
<Directory "/var/www/wiki/
➥templates">
    deny from all
</Directory>
```

A *TWikihez* tartozik egy beállító parancsfájl, amelyet a *TWiki* telepítéséhez futtatunk. Ez a parancsfájlt nem csak az először telepítéskor alkalmazzuk, hanem később is, amikor bővítményeket akarunk engedélyezni.
Most készen állunk a *TWiki* beállítására, tehát adjuk meg a böngészőben a *TWiki* beállító parancsfájl címét: *http://www.pelda.com/wiki/bin/configure*. A *Security (Biztonság)* rész különösen érdekes lehet, de ezt nemsokára meg is látogatjuk. Amíg nem regisztráltuk az első felhasználót, érdemes minden beállítást az eredeti állapotban hagyni. Ha a beállító parancsfájl figyelmeztetést vagy hibaüzenetet küld, először azokat kell orvosolni, majd újraindítani a parancsfájlt. Miután a *Next (Következő)* gombra kattintunk, meg kell adnunk egy jelszót. A jövőben ezt a jelszót kell használni a beállító parancsfájl minden egyes futásakor, az illetéktelen hozzáférés megakadályozása érdekében.
A beállítás sikeres elvégzése után ideje belépni a wikibe. Irányítsuk a böngészőt a *http://www.pelda.com/wiki/bin/view* címre, és megjelenik a *Main (Fő)* hálózat. A lap közepén található hivatkozás a regisztrációra szolgál. Regisztráljuk magunkat felhasználóként. Mindenképpen érvényes e-mail címet adjunk meg, mivel a program ennek segítségével érvényesíti a fiókunkat. A felhasználói fiók megerősítése után adjuk magunkat a *TWikiAdminGroup-hoz* (*TWikigazdacsoport*). Térjünk vissza a *Main* hálózatra, majd kattintsunk a bal oldalon látható *Groups (Csoportok)* hivatkozásra, ezután pedig válasszuk a *TWikiAdminGroup*-ot. Szerkesszük ezt az oldalt, és módosítsuk a `GROUP` változót úgy, hogy az új felhasználónevet tartalmazza:

```
Set GROUP =
➥%MAINWEB%.TiLeggett
Set ALLOWTOPICCHANGE =
➥%MAINWEB%.TWikiAdminGroup
```

A sorok elején szereplő három szóköz elengedhetetlen.
Ez a két sor hozzáadja a felhasználót a *TWikiAdminGroup*-hoz, és csak a *TWikiAdminGroup* tagjai számára engedélyezi a csoport megváltoztatását. Itt az ideje, hogy engedélyezzük a hitelesítést a wikiben, tehát menjünk vissza a *http://pelda.com/wiki/bin/configure* oldalra. A Security rész számos hasznos lehetőséget kínál. Győződjünk meg, hogy a `{UseClientSessions}` (*Ügyfélmunkamenetek alkalmazása*), valamint a `{Sessions}{UseIPMatching}` (*Munkamenetek, IP megfeleltetés*) lehetőség engedélyezett. Ezen kívül a `{LoginManager}` (*Bejelentke-*

zés-kezelő) beállítása legyen `TWiki::Client::TemplateLogin`, a `{PasswordManager}`-é (*Jelszókezelő*) pedig `TWiki::Users::HtPasswdUser`. Ha a kiszolgáló támogatja, állítsuk a `{HtPasswd}{Encoding}` (kódolás) értékét sha1-re. Mentsük a változásokat, majd térjünk vissza a wikibe. Amennyiben az oldal nem léptet be automatikusan, a bal felső sarokban található hivatkozás segítségével léphetünk be.

Most, hogy a hitelesítés működik, érdemes korlátozni a *wikit*, hogy illetéktelen személyek ne változtathassák tiltott adatok tárhelyévé a dokumentáció tárhelyét. A *TWiki* hitelesítő rendszere meglehetősen kifinomult, ami az egész webhely webhelyen érvényes tulajdonságoktól egészen a meghatározott témákig szűkül. A *Main* hálózat lezárása előtt még el kell végeznünk néhány feladatot. Ha csak néhány felhasználó tudja módosítani a Main hálózatot, az új felhasználók regisztrálása sikertelen lesz. Ez azért van, mert a felhasználói regisztrációs folyamat létrehoz egy témát az adott felhasználó számára a *Main* hálózaton belül. A *Dakarban* ezt a *TWikiRegistrationAgent* felhasználó hajtja végre. A Main hálózaton balra fent található *Jump* (ugrás) doboz segítségével ugorjunk a *WebPreferences* témára. Módosítsuk a témát úgy, hogy az alábbi négy sort tartalmazza, majd mentsük a változtatásokat:

```
Set ALLOWTOPICRENAME =
➥%MAINWEB%.TWikiAdminGroup
Set ALLOWTOPICCHANGE =
➥%MAINWEB%.TWikiAdminGroup
Set ALLOWWEBRENAME =
➥%MAINWEB%.TWikiAdminGroup
Set ALLOWWEBCHANGE =
➥%MAINWEB%.TWikiAdminGroup,
 -->;%MAINWEB%.TWiki
 ➥RegistrationAgent
```

Ez kizárólag a *TWikiAdminGroup* tagjai számára teszi lehetővé a Main hálózat módosítását vagy átnevezését, valamint a Main hálózat tulajdonságainak frissítését.

A *TWikiRegistrationAgent* felhasználó számára továbbá lehetővé teszi hogy új saját témát hozzon létre a felhasználóknak, amikor új tagok regisztrálnak. Beépítettem egy foltot is, amelyet mindenkinek alkalmaznia kell a *lib/TWiki/UI/Register.pm* fájlon. A folt kódja alább látható, de a *Linux Journal FTP*-helyéről is letölthető (lásd a kapcsolódó címeket):

```
-- lib/TWiki/UI/
➥Register.pm.orig        2006-
➥01-04 01:34:48.968947681
➥-0600
+++ lib/TWiki/UI/Register.pm
➥2006-01-04 01:35:48.999652157
➥-0600
@@ -828,11 +828,12 @@

     my $userName = $data->
➥{remoteUser} || $data->
➥{WikiName};
     my $user = $session->
➥{users}->findUser(
➥$userName );
+    my $agent = $session->
➥{users}->findUser( $twiki
➥RegistrationAgent );
     $text = $session->expand
➥VariablesOnTopicCreation
➥( $text, $user );

     $meta->put( 'TOPICPARENT',
     ➥{ 'name' => $TWiki::cfg
     ➥{UsersTopicName}} );

-    $session->{store}->
➥saveTopic($user, $data->
➥{webName},
+    $session->{store}->
➥saveTopic($agent, $data->
➥{webName},

$data->{WikiName}, $text,
➥$meta );
     return $log;
 }
```

Máskülönben nem lehet létrehozni az új felhasználók saját könyvtárát, és az új felhasználó regisztrációja sikertelen lesz. Miután ellenőriztük, hogy a *Main* hálózat le van zárva, ugyanezt kell tennünk a *TWiki* és a *Sandbox (Homokozó)* hálózattal. Amikor elkészültünk a *TWiki* beállításával, biztonságossá kell tenni a fájlok hozzáférését:

```
# find /var/www/wiki/ -type d
# -exec chmod 0755 {} ';'
# find /var/www/wiki/ -type f
# -exec chmod 0400 {} ';'
# find /var/www/wiki/pub/ -type
# f -exec chmod 0600 {} ';'
# find /var/www/wiki/data/
# -type f -exec chmod 0600 {}
# ';'
# find /var/www/wiki/lib/
# LocalSite.cfg -exec chmod
# 0600 {} ';'
# find /var/www/wiki/bin/ -type
# f -exec chmod 0700 {} ';'
# chown -R apache
# /var/www/wiki/*
```

Ahogy korábban említettem, a *TWikiben* rendelkezésre áll egy bővítményrendszer. Számos bővítmény érhető el a *TWiki* webhelyen. Használat előtt győződjünk meg, hogy a kiválasztott bővítményeket frissítették a *Dakar* számára.

A felhasználók tájékoztatása

A rendszerfelügyelet egyik fontos – gyakran hanyagolt – eleme a felhasználók tájékoztatása. A felhasználók nagy része szeretne értesülni arról, ha új szolgáltatás érhető el vagy az erőforrások leálltak, illetve nem hozzáférhetők. Nem csak a felhasználókat teszi boldogabbá a folyamatos tájékoztatás, hanem a mi életünket is megkönnyítheti. A központi fájlkiszolgáló leállása esetén a legkevésbé azzal szeretnénk foglalkozni, hogy a felhasználók kérdéseire válaszoljunk, amikor azt szeretnék tudni, hogy miért nem tudják elérni a fájljaikat. Ha megtanítjuk a felhasználókat, hogy először egy központi helyen ellenőrizzék az eszközök állapotát, a hiba észlelése után nem kell mást tennünk, mint egy központi helyen közzétenni, hogy valamilyen probléma adódott. A levelező listák is jók erre, de mi van olyankor, ha a levélkiszolgáló nem működik? Bizonyos emberek – például a felettesünk vagy a vállalat alelnöke – esetleg azonnal szeretnének értesülni a helyzetről. Ezek hírek nem feltétlenül alkalmasak arra, hogy naponta mindenkinek e-mailben küldjük szét. Létrehozhatnánk egy újabb levelező listát az ilyen értesítések számára, de érdemes megfontolni egy blog létrehozását. Azok számára, akik nem ismerik a blogokat, hadd idézzek ismét a *Wikipédiából*: *„a blog egy olyan webhely, amelyben rendszeresen naplóbejegyzéseket tesznek közzé, fordított időrendi sorrendben megjelenítve.”* A *blog* fogalma a napló formájában

már évszázadok óta jelen van, a ***blogok*** azonban mostanában gombamód szaporodnak az interneten. A ***blogokat*** gyakran személyes naplóként vagy hírek közlésére indítják, ugyanakkor a hihetetlenül hasznosak lehetnek a rendszergazda számára is. A rendszergazda a blog segítségével percről percre tájékoztathatja a felhasználókat, hogy éppen min dolgozik, valamint a rendszer állapotáról. Ha szorgalmasan frissítjük a ***blogot***, könnyen visszatekinthetünk a végzett eredményekre, így megindokolhatjuk a régóta áhított fizetésemelést. Ezen kívül segít nyomon követni a munkatársak tevékenységét. Ráadásul, mivel számos ***blog*** programcsomag biztosít ***RSS*** tartalmat, a felhasználók feliratkozhatnak a ***blogra***, hogy értesítést kapjanak az új bejegyzésekről.

WordPress

Manapság számos ***blog*** programcsomag létezik, most azonban a ***WordPress***-szel foglalkozunk. A ***WordPress*** gyors, és kellemes bővítmény illetve bőr felülettel rendelkezik, ami lehetővé teszi a tetszés szerinti testreszabást. A ***WordPress*** futtatásához csupán ***Apache***-ra, ***MySQL***-re és ***PHP***-ra van szükség. Nem részletezném a ***WordPress*** telepítésének mikéntjét, mert az interneten elérhető dokumentáció eléggé világos és könnyen követhető. Ehelyett onnan kezdem, ahol a telepítés véget ér, és bemutatok néhány hasznos bővítményt. Annak ellenére, hogy már megjelent a ***WordPress v2.0***, azt javaslom, hogy az v1.5.2-es változattal kezdjünk. Az első v2.0 kiadásnál felmerültek bizonyos problémák, ami miatt érdemes a v2.0.1. változatra várni. Ráadásul a bővítmények jó részét még nem frissítették az új rendszerhez.

A ***WordPress*** telepítése után az legyen az első dolgunk, hogy rendszergazda felhasználóként bejelentkezünk. A bejelentkezés után megjelenik a ***Dashboard (Műszerfal)***. A lap tetején látható menüben a ***Write (Írás)***, ***Manage (Vezérlés)***, ***Links (Hivatkozások)*** és a többi lehetőség található. Először a ***Users (Felhasználók)*** lehetőségre kattintva létre kell hoznunk egy saját fiókot. Amikor ez betöltődött, a főmenüben elérhetővé válik a ***Your Profile (Saját profil)*** és az ***Authors & Users (Szerzők és felhasználók)*** lap. Kattintsunk az ***Authors & Users*** lapra, gördítsük le az ***Add New User (Új felhasználó hozzáadása)*** részhez, majd töltsük ki a szövegmezőket. A felhasználó a hozzáadás után a följebb található ***Registered Users (Regisztrált felhasználók)*** részben jelenik meg. A több elérhető adatoszlop közül kattintsunk a ***Promote***-ra (***Előléptetés***). Ha előléptetünk egy felhasználót, szerzővé válik, és a szintjétől függően további jogokat kaphat. A felhasználó szintje az előléptetés után 1. A szint két oldalán egy plusz és egy mínusz jel látható – ezekkel változtathatjuk a felhasználó szintjét. Növeljük 9-re, ami a nem rendszergazda felhasználók számára elérhető legmagasabb szint. Ha valamikor esetleg szerzővé előléptetett felhasználót kell törölnünk, nem kell mást tenni, csak 1 alá csökkenteni a szintjét, majd törölni a felhasználót. A kapcsolódó címekben szerepel egy hivatkozás, amelyen részletesebben tárgyalják a felhasználói szintekhez tartozó jogokat.

Van még néhány beállítás, amit esetleg megváltoztathatunk. A ***General Options (Általános beállítások)*** részben található jelölőnégyzetekkel engedélyezhetjük, hogy mindenki, aki regisztrál, ***blog*** felhasználóvá váljon, valamint, hogy a megjegyzések hozzáadásához a felhasználóknak be kelljen jelentkezniük. Ezeket a lehetőségeket a biztonsági tényezőktől és a ***blog*** nyilvánosságától függően engedélyezhetjük vagy tilthatjuk. A ***Linux Journal*** webhelyén a felhasználók nem regisztrálhatnak, de bárki hozzáadhat megjegyzéseket anélkül, hogy bejelentkezne. Érdemes böngészni a menüket és azok lehetőségeit, hogy a webhelynek megfelelően állítsuk be azokat.

WordPress bővítmények

A ***WordPress*** meglehetősen moduláris bővítményrendszerrel rendelkezik, és rengetegen írtak hozzá bővítményeket. A ***WordPress*** ezen kívül a kategória fogalmát alkalmazza. A kategóriákat számos célra lehet használni, többek között mini-blogokat lehet létrehozni a különböző felhasználói közösségek számára, vagy csoportosítani lehet a rendszer különböző elemeiről szóló bejegyzéseket. Viszont nem szeretnénk, ha minden felhasználó láthatná az összes kategóriát.

A ***Userextra*** és a ***Usermeta*** bővítmény együttes használata éppen az ilyen kérdések kezelését teszi lehetővé. Ha követjük a bővítmény telepítési utasításait, két további menü jelenik meg az ***Options*** menüben, valamint egy a ***Manage*** menüben, amelyekkel lehetővé válik a hozzáférések finomítása. A ***HTTP Authentication*** bővítményt is hasznosnak találhatjuk. Ennek segítségével külső hitelesítő eljárást – például ***Apache BasicAuth*** eljárást – alkalmazhatunk a ***WordPressben*** történő hitelesítéshez. Ez remek lehetőség, ha már rendelkezünk olyan ***LADP*** könyvtárral vagy ***Kerberos*** tartománnyal, amelyet hitelesítésre használunk, valamint a ***mod_auth_ldap*** vagy a ***mod_auth_kerb*** éppen fut.

A ***WordPresshez*** számos további bővítmény érhető el a ***WordPress Codex*** és a ***WordPress Plugin DB*** webhelyen. Ha valaki úgy érzi, hogy egy bizonyos szolgáltatás hiányzik, a ***WordPress*** webhelyen számos példa és dokumentáció található, valamint a bővítmény tárhelyek segíthetnek a saját bővítmények megírásában.

Összegzés

Remélem, hogy a ***wikik*** és a ***blogok*** világában tett forgószélszerű kirándulás segített felismerni, hogy miként használhatjuk ezeket a hajó simább útjának biztosítására, valamint arra, hogy a felhasználók minden kívánt információt megkapjanak. Ugyanúgy, ahogy a hajót számos különféle vitorla képes előre vinni, sok különböző wiki és blog programcsomag létezik. Azt a csomagot kell választani, amelyik boldoggá és hatékonnyá teszi a felhasználóinkat.

Linux Journal 2006., 144. szám

Ti Leggett (ti@daleggetts.com) főállású rendszergazda. Amikor éppen nem dolgozik, valószínűleg a Gibson B-25-ösén játszik, barkácsol a ház körül vagy asztalosmunkát végez.

KAPCSOLÓDÓ CÍMEK

➲ http://linuxjournal.com/article/8832

A MediaWiki telepítése és beállítása

Telepítsünk saját MediaWiki rendszert

Évekkel ezelőtt ismertem meg a wiki-ötletet. Egyik kollégám hívta fel a figyelmem egy webhelyre, ahol a látogatókat arra buzdítják, legyenek a webhely írói és szerkesztői. Úgy gondolta, a wiki idővel rengeteg tudást foglal majd magába a közös tapasztalatoknak köszönhetően anélkül, hogy szükség volna bármiféle koordinálásra vagy szerkesztői munkára. Első reakcióm ambivalens volt. Egyrészt egy jóhiszemű, fegyelmezett felhasználói közösségnek a wiki tényleg nagyon hatékony eszköze lehet. Ugyanakkor viszont káoszba is süllyedhet – kétségkívül így is történt néha.

Az évek során a wikik megszokottá váltak. Tucatnyi különböző programcsomag áll rendelkezésünkre szinte az összes programozási nyelven, majdnem minden web/adatbázis keretrendszerhez. Többé már nem a wikik hozzáférhetősége és a telepítés bonyolultsága a kérdés, hanem az, hogy a wiki megfelelő eszköz-e számunkra és melyik csomag passzol leginkább igényeinkhez.

A döntés talán nem egyszerű, bár a kitűnő ***Wikipedia Project*** bizonyítja, hogy igenis elérhetők nagyszerű eredmények. Egy, a cikkem megírása előtt pár héttel közzétett tanulmány szerint, ha tudománnyal kapcsolatos cikkekről van szó, a ***Wikipedia*** precízsége csak hajszálnyival marad el az ***Encyclopaedia Britannicáétól***. Tudvalevő, hogy az ***Encyclopaedia Britannicát*** író és szerkesztő szakértők pénzt kapnak munkájukért, míg a ***Wikipedia*** önkéntes alapon működik (persze az önkéntesek is lehetnek szakértők), nyilvánvaló, hogy előzetes félelemmel ellentétben a wikik nem feltétlenül süllyednek káoszba.

Mindezek után, szilárdan meg vagyok győződve arról, hogy a ***Wikipedia*** népszerűsége nem a wikik napjainkban tetőző divatjából ered, hanem abból, hogy egy lelkes csapat áll mögötte, akik a színfalak mögött a tartalom használhatóságáért dolgoznak.

Említettem, hogy számtalan lehetőségünk van saját wiki létrehozására webkiszolgálónkon. Az egyik legismertebb csomag a ***MediaWiki***, amely a ***Wikipediát*** is működteti. A cikkből megtudhatjuk, hogyan telepíthetjük kiszolgálónkra, hogyan kell a statikus tartalmat létrehozni és szerkeszteni, és hogyan hozhatók létre akár dinamikus speciális lapok is.

A MediaWiki telepítése

Annak ellenére, hogy a ***Wikipedia*** hatalmas mennyiségű szöveget tartalmaz, az azt működtető ***MediaWiki*** program meglepően kisméretű, könnyen érthető és könnyen telepíthető. Maga a program ***PHP***-ben íródott, és ***MySQL*** adatbázist használ a tartalom tárolására és indexelésére. Így a ***MediaWiki*** webhely egyik oldala sem lemezen levő fájl, hanem a ***PHP*** által futásidőben létrehozott tartalom.

A ***MediaWiki*** program telepítéséhez tehát mindössze egy ***PHP***-t és ***MySQL***-t futtató kiszolgálóra van szükség. A ***PHP*** számos platformon működik, de cikkünkben feltételezzük az ***Apache*** webszerver használatát a ***PHP*** egy újabb változatával (legalább 4.1.2-vel, de a 4.3 a preferált) és ***MySQL***-lel (legalább 3.2.x-vel, de a 4.0.x a preferált). A legtöbb ***Linux*** terjesztés vagy tartalmazza már a ***PHP***-t és a ***MySQL***-t, vagy könnyen elérhetővé teszi online tárolóiból.

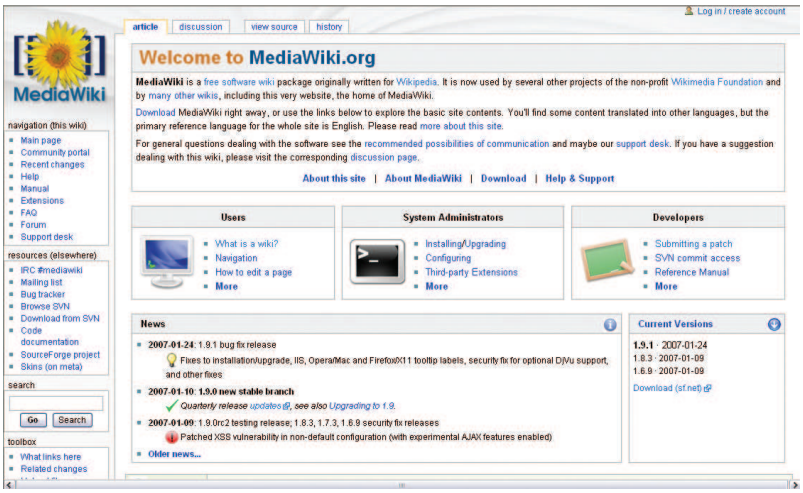
A ***MediaWiki*** telepítéséhez az Apachehoz és a ***MySQL***-hez is adminisztrátori jogok kellenek. Ha nincs saját kiszolgálónk, valószínűleg a rendszeradminisztrátor segítségét kell kérnünk a megfelelő beállításfájlok módosításához.

Bár a ***MediaWiki*** program közvetlenül a ***SourceForge***-ról is letölthető, ajánlott a fő ***MediaWiki*** helyen megkeresnünk, a ➲ ***www.mediawiki.org*** címen. Kattintsunk az oldal jobb oldalán levő ***Versions&Download*** menüpontra, majd a megfelelő hivatkozásra az aktuális stabil változat letöltéséhez. Csak akkor töltsünk le korábbi vagy fejlesztői stádiumban levő változatot, ha kezelni tudjuk a felmerülő problémákat.

A programot egy, a változatszámmal jelölt ***.tar.gz*** fájl tartalmazza. Az 1.5.3-as változat például a ***mediawiki-1.5.3.tar.gz*** nevű fájlban található, amely 2005. december 21-én jelent meg. A fájlt helyezzük el valahol a fájlrendszerünkben (jómagam egy ***/downloads*** nevű könyvtárat tartok fenn hasonló célokra). Ehhez rövidesen visszatérünk, de előbb foglalkozzunk egy kicsit az ***Apache*** beállításával. Ha a webhelyünk csak a ***MediaWiki*** programot fogja futtatni, valószínűleg nincs okunk a (hagyományosan ***httpd.conf***-nak nevezett) ***Apache*** beállításfájl módosítására. Ilyenkor csomagoljuk ki a ***.tar.gz*** fájlt a webhelyünk htdocs könyvtárába a ***MediaWiki*** oldal útmutatása alapján. Ha a terjesztésünk más könyvtárt használ dokumentumgyökérként, akkor nyilván annak megfelelően kell eljárnunk.

Virtuális kiszolgálók

Ha a ***MediaWiki***-t egy ***Apache*** szerver virtuális kiszolgálójára szeretnénk telepíteni, módosítanunk kell az ***Apache***



beállításfájlokat, hogy létrehozzunk egy új virtuális kiszolgálót (***virtual host***) a rendszeren. A ***HTTP*** újabb változatai megkövetelik, hogy a böngészők a dokumentumokat egy bizonyos gazdagépnévről kérjék le. Az ***Apache virtual host*** funkciója lehetővé teszi, hogy egyetlen kiszolgáló több gazdagépnév alapján szolgáltasson dokumentumokat akkor is, ha ezek a nevek egyetlen ***IP*** címhez tartoznak. Ha a ***MediaWikit*** csak ki szeretnénk próbálni anélkül, hogy ez befolyásolná a többi webhelyet, illetve, ha az oldalaink többsége a ***MediaWikivel*** nem kompatibilis keretrendszeren fut (ilyen például a ***Zope*** vagy a ***Ruby on Rails***), egy virtuális kiszolgáló létrehozása igencsak ajánlott. Az ilyen virtuális hostokat én gyakran használom új programok kipróbálására és beállításaikkal való kísérletezésre, így a többi oldal működését nem veszélyeztetem. Az ***Apache VirtualHost*** beállításainál megadhatunk egy vagy több ilyen virtuális kiszolgálót, mindegyiket saját beállításokkal. Saját rendszeremen a ***MediaWiki*** beállításához a következő ***VirtualHost*** részt adtam hozzá a ***httpd.conf***-hoz:

```
<VirtualHost 69.55.225.93>
ServerName wiki.lerner.co.il
    ServerAlias
➥mediawiki.lerner.co.il
➥wikipedia.lerner.co.il
ServerAdmin reuven@lerner.co.il
DocumentRoot /usr/local/
➥apache/v-sites/
➥wiki.lerner.co.il/www/
DirectoryIndex index.php
CustomLog
/usr/local/apache/v-sites/
➥wiki.lerner.co.il/logs/
➥access-log combined
CustomLog
/usr/local/apache/v-sites/
➥wiki.lerner.co.il/logs/
➥referer-log referer
ErrorLog /usr/local/apache/
➥v-sites/wiki.lerner.co.il/
➥logs/error-log
</VirtualHost>
```

A `VirtualHost` rész kódjának első két sora meghatározza az új virtuális kiszolgáló nevét és álneveit. Ha `alias`-okat, azaz álneveket is megadunk, azzal lehetővé tesszük, hogy más címmel is el lehessen jutni a webhelyre. (A szolgáltatómnál minden gépnév a ***lerner.co.il*** tartományban a kiszolgálóm ***IP*** címére mutat, így akárhány virtuális kiszolgálót létrehozhatok, vagy éppen törölhetek anélkül, hogy a ***DNS*** beállításokon módosítanom kellene.) A `ServerAdmin` opcióhoz a saját e-mail címemet adom meg, így biztosítva, hogy a hibaüzenetek a látogatókat hozzám irányítsák.
Következő lépésként megadjuk a gyökérkönyvtárat a ***PHP*** fájlok számára. Nálam ez a könyvtár a ***/usr/local/apache/v-sites*** alatt található, mindig ebben helyezem el a virtuális kiszolgálókat. Mindegyik oldal kap egy alkönyvtárat a nevének megfelelően, benne további két alkönyvtárral: www és logs. A fenti beállításban tehát a `DocumentRoot` a ***v-sites/wiki.lerner.co.il/www*** lett, a hozzáférési, a hivatkozási és a hibanaplókat pedig a ***v-sites/wiki.lerner.co.il/logs***-ba tettem.
A fenti beállítások befejeztével újraindítjuk az ***Apache***-ot.
A ***http://wiki.lerner.co.il*** megnyitásakor hibaüzenetet kapunk, hiszen még nem telepítettük magát a programot. A ***MediaWiki*** program telepítéséhez letöltöm a legutóbbi ***.tar.gz*** fájlt, és kicsomagolom a ***v-sites/wiki.lerner.co.il*** könyvtárba:

```
cd /usr/local/apache/v-sites/
➥wiki.lerner.co.il/
tar -zxvf /downloads/
➥mediawiki-1.5.3.tar.gz
mv mediawiki-1.5.3 www
```

Ezek után a böngészőmben a ***http://wiki.lerner.co.il***-t megnyitva egy üzenetet kaptam, miszerint még szükség van a program beállítására is. A jogosultságok sem voltak még rendben, tehát a ***MediaWiki*** nem tudta írni saját beállításkönyvtárát. És ő ahelyett, hogy leállt volna egy nehezen érthető hibaüzenettel, kedvesen arról értesített hogy probléma van a fájlrendszerbeli jogosultságokkal, és még javaslatot is tett arra, hogy hogyan használjam a `chmod`-ot a probléma kiküszöbölésére. Miután a jogosultságokat megváltoztattam, a ***MediaWiki*** megjelenítette a kezdő konfigurációs képernyőt. Ezután nem sok tennivalóm maradt: beírtam a wikim nevét (***ReuvenWiki***), és szinte mindenhol meghagytam az alapbeállítást. Magában a wiki rendszerben megadtam egy jelszót a ***WikiSysop*** felhasználónak, majd beírtam az adatbázis-felhasználó és a ***MySQL*** adminisztrátor jelszavait. Itt háromféle felhasználónévvel dolgozunk, s ez némileg bonyolult. ***Linux*** rendszerünknek és a ***MySQL***-nek is megvannak a maga felhasználói és jogosultságai és ugyanez áll a ***MediaWiki*** programra is. És bár ez túlzásnak tűnik, ez biztosítja a nagyfokú rugalmasságot, és lehetővé teszi, hogy a ***MediaWikit*** biztonságos környezetben futtassuk.
Ha kitöltöttem az összes fenti információt, az ***Install*** gombra kattintottam. Más webes alkalmazásoktól eltérően, amelyek kimenetet csak akkor adnak, ha készen vannak a munkájukkal, a ***MediaWiki*** telepítő folyamatosan informál a telepítésről,

így képet alkothatunk arról, hogy mit csinál éppen és az mennyi ideig fog még tartani. A végén (remélhetőleg) látunk egy üzenetet, hogy a telepítés sikeresen megtörtént, s egy figyelmeztetést, hogy a ***config/LocalSettings.php*** fájlt a szülő könyvtárba kell helyeznünk (a *www* könyvtár):

```
mv -v config/LocalSettings.php .
```

Miután ezzel is végeztem, újra beírhatom az *URL*-t (***http://wiki.lerner.co.il***) a böngészőmbe. A *MediaWiki* kezdőoldala fogad, tudtomra adja, hogy a program sikeresen települt, és két oldalra javasol, az egyiken a felületet lehet testre szabni, a másik segítséget nyújt a használathoz és a beállításokhoz.

Alapvető navigáció és szerkesztés

Ha használtunk már valaha *Wikipediát*, akkor a *MediaWiki* oldalak közötti navigálás igencsak ismerős lesz számunkra. Ha még nem, az alapvető szabályok a következők: ha az oldalon egy hivatkozásra kattintunk, akkor azzal a *MediaWiki* webhely egy másik oldalára jutunk. Ez alól kivétel az olyan hivatkozás, amelyet egy nyíl ikon követ, mert ilyenkor a hivatkozás egy másik kiszolgálón levő oldalt nyit meg.
A webhely oldalai nem csak olvashatók a nagyközönség számára, hanem szerkeszthetők is. Ha meg szeretnénk változtatni egy oldal tartalmát, kattintsunk az ***Edit (Szerkesztés)*** hivatkozásra (ez mindig a *MediaWiki* oldalak tetején található), módosítsuk a tartalmat belátásunk szerint, majd, ha végeztünk, kattintsunk a *Save* gombra.
A *MediaWiki* igyekszik a lehető legkönnyebbé tenni a felhasználók részvételét anélkül, hogy szükségük lenne a technikai háttér megértésére. Mert hát nem kell egy végfelhasználónak ismernie a *HTML*-t; csak gombokat kell nyomkodniuk, illetve egy egyszerűsített formázási rendszert használva formázni a bejegyzéseket. Például, ha azt írjuk be: `[[Foo]]`, az létrehoz egy hivatkozást a *Foo*-ra függetlenül attól, hogy az már létezik. Majd, ha beírjuk a következő formájú hivatkozást: `[[Foo | Hivatkozás a Foo-ra]]`, az elhelyez egy hivatkozást a *Foo* oldalra a ***„Hivatkozás a Foo-ra”*** szöveggel.
A *MediaWiki* egyszerű leírónyelvénél fontosabb az a tény, hogy minden, a rendszeren „elkövetett” változtatás egy könnyen használható változat-követő rendszerben tárolódik. Az adott oldal ***History*** hivatkozására kattintva még egy nem számítástechnikai beállítottságú felhasználó is megtalálja az oldal korábbi változatait, láthatja, hogy ki eszközölte a változtatást vagy összehasonlíthatja egy bizonyos dokumentum két változatát. A felhasználók figyelőlistákat is beállíthatnak, hogy e-mail értesítést kapjanak az oldal minden egyes módosítása után.
Így lehetőség nyílik egy adott oldal frissítéseinek követésére, és felügyelhető a tartalom, a potenciális károkat elkerülendő.
Alapértelmezésként minden lap bal oldalán egy rövid menü mutatja a leggyakrabban felkeresett hivatkozásokat, beleértve a wiki főoldalát (kezdőlap), a legújabb módosítások listáját, egy véletlenszerűen kiválasztott oldalt és a dokumentációt. Ugyanitt egy kereső űrlap is helyet kapott.
Ha beírunk ide egy szövegrészletet és a *Search* gombra kattintunk, felbukkan az olyan dokumentumok listája, amelyek címe és/vagy tartalma a keresési feltételnek megfelel. Mivel a *MediaWiki* tartalmak mind *MySQL* adatbázisban tárolódnak, és mert a *MySQL* újabb változataiban beépített szövegindexelő funkció található, az ilyen keresések viszonylag gyorsak.
A *MediaWiki* webhely minden oldalának *URL*-je a webhely *URL*-jével kezdődik (esetemben ez a ***http://wiki.lerner.co.il***), amelyet az ***index.php/*** követ, majd az oldal címe. Az oldal címében előfordulhat szóköz is, ezeket az *URL* egy aláhúzás karakterrel jelöli. Tehát a kiszolgálómon található *Reuven Lerner* nevű oldal teljes *URL*-je így festene: ***http://wiki.lerner.co.il/index.php/Reuven_Lerner***.
Kicsit zavaró, de a *MediaWiki URL*-ek betűállás-érzékenyek, így egy nem megfelelő betűállású szót begépelve egy új oldal jön létre. Ne feledjük, a wiki sosem ad igazi „az oldal nem található” hibaüzenetet, mert mi, látogatók esetleg pont azt az oldalt szeretnénk létrehozni.
A *MediaWiki* webhelyeken nincs oldal-hierarchia. Egyetlen névtér van, amely bizony összevisszasághoz vezet. A félreérhetőséget csökkentendő, az oldalak további, zárójelek közé zárt információkat tartalmazhatnak (ami így az *URL* részét képezi). Másik megoldás, ha a felhasználók létrehoznak egy úgynevezett ***„disambiguation”*** oldalt, amely a hasonló vagy azonos nevű oldalak hivatkozásait tartalmazza.
Ha lehetővé szeretnénk tenni a felhasználóknak képek feltöltését, módosítanunk kell a *LocalSettings.php* fájlt, amelynek már léteznie kell kiszolgálónk `DocumentRoot` könyvtárában.
Az általam telepített változatban a *LocalSettings.php* 73. sora (a *PHP* kód idevágó megjegyzése alatt) tartalmaz egy `true`-ra állított `$wgEnableUploads` megjegyzéssel érvénytelenített változót. A sor megjegyzés-jeleit törölve lehetővé tesszük a felhasználók számára képek feltöltését a kiszolgálóra. Ezek a feltöltések alapértelmezetten az ***images (képek)*** alkönyvtárba kerülnek (***www/images***), amelynek az *Apache*-folyamat által írhatónak kell lennie.
Ha engedélyeztük a képek feltöltését, a bejelentkezett felhasználók két lépésben feltölthetik azokat. Első lépés: a felhasználó a wiki-oldal szerkesztésekor rákattint a képek beillesztése gombra, vagy kézzel beilleszt egy `[[Image:Example.gif]]`-formátumú címkét. (*MediaWikiben* minden képnek `Image:` az előtagja, de létezik egy egyszerű névtér a képekhez, ugyanúgy, mint a szöveges tartalmú oldalakhoz.) Ha a felhasználó végzett az oldal szerkesztésével, rákattint a hiányzó kép hivatkozására. Ekkor előbukkan egy HTML űrlap, lehetővé téve a hiányzó kép feltöltését a felhasználó böngészőjén keresztül. Más típusú dokumentumok feltöltése is hasonlóképpen zajlik.

Speciális lapok

A wikik a tartalomkezelő rendszerek (***CMS, content-management system***) egy speciális formája. S mint ilyen programok, céljuk az, hogy megjelenítsék egy általában statikus dokumentum legújabb változatát.
Ez ellentétben áll a legtöbb más kiszolgálóoldali programmal, melyek kimenete minden felhasználónál és lehívásnál más. Természetesen a *MediaWikinek* be kell olvasnia minden egyes dokumentumot úgy,

ahogyan az megjelent, hogy a szükséges különböző *URL*-eket létre tudja hozni a szövegtörzsben említett már meglevő és az új hivatkozásokhoz. A *MediaWiki* legnagyobb része – a programozott oldalakkal ellentétben – a könnyű létrehozásra és a statikus tartalom terjesztésére szakosodott. A *MediaWiki* fejlesztői azonban felismerték, hogy ők és mások is talán dinamikusan generált oldalakat szeretnének létrehozni, így született meg a `Special:` megnevezés. Minden oldal, melynek neve `Special:`-lal kezdődik, az egy, nos, másként kezelt oldal, amely a dokumentum törzsét egy *PHP* függvény kimenetéből tölti fel. Saját speciális lap létrehozásához írnunk kell egy *PHP* függvényt, majd bejegyezni azt a *MediaWikibe*. Az első lépés egy új *MediaWiki* kiterjesztés létrehozása, melyet behelyezünk közvetlenül a *MediaWiki* `DocumentRoot` alatt levő kiterjesztés könyvtárba. (Az alap *MediaWiki* terjesztésben megvan ez a könyvtár, de általában üres a telepítés után.) Hozzuk létre a következő *extensions/SpecialHello.php* fájlt:

```
<?php
$wgExtensionFunctions[] =
➥"wfExtensionSpecialHello";
function
➥wfExtensionSpecialHello()
{
     global $wgMessageCache;
     $wgMessageCache->
     ➥addMessages(array('hello'
     ➥=> 'Hello page'));
     require_once('includes/
     ➥SpecialPage.php');
     SpecialPage::addPage
     ➥(new SpecialPage
     ➥('Hello'));
}
?>
```

Az első sor a függvény nevét (`"wfExtensionSpecialHellov`) hozzáfűzi a globális `$wgExtensionFunctions` tömbhöz, s így a függvényt a kiterjesztés könyvtárba helyezi. Sok *MediaWiki* kiterjesztés megváltoztatja bizonyos címkék kimenetét; ez esetben egy teljes oldalt szeretnénk létrehozni, s nem egy címke viselkedését megváltoztatni. Mindazonáltal egy kiterjesztést hozunk létre.

Ezután meghatározzuk a függvényünket. Először is módosítjuk a `$wgMessageCache`-t, így a kiterjesztésünk inkább egy speciális lapra fog hasonlítani, semmint egy címkemódosító kiterjesztésre. Ha elfelejtjük ezeket a kezdő sorokat beírni, a speciális lap akkor is működni fog, de a `Special:Specialpages`-en látható listája furcsa lesz, mert a cím < és > jelek között jelenik meg, mintha címke volna. Fontos, hogy az oldal neve kisbetűvel kezdődjön az `addMessages` meghívásában még akkor is, ha a `SpecialPage::addPage` hívásban nagybetűvel szerepel. Ha nem figyelünk a betűállásra, igen fura címeket kapunk végeredményül.
A két utolsó sor importálja a speciális lapok kódját, majd létrehoz egy speciális lap példányt és a könyvtárhoz adja.
Az, hogy mit csinál a speciális lapunk, egy ugyanolyan nevű másik fájltól függ (ez a *SpecialHello.php*). Ez a fájl a kiterjesztés könyvtárral párhuzamosan az *includes* könyvtárban van, közvetlenül a *MediaWiki* `DocumentRoot` alatt. Ebben a könyvtárban számos, a *MediaWikivel* együtt járó szabványos speciális lap található, például *SpecialNewpages.php*, *SpecialUserrights.php* és *SpecialImagelist.php*. Ezek a függvények hozzáférnek a kiszolgálóoldali *MySQL* adatbázishoz, számításokat végeznek és elérnek külső oldalakat, majd az végeredményt lekérik és megjelenítik egy szabványos *MediaWiki* kimeneti oldalon.
Íme egy egyszerű *includes/SpecialHello.php* fájl:

```
<?php
function wfSpecialHello() {
  global $wgOut;
  $wgOut->addHTML('Hello,
  ➥world');
}
?>
```

A fenti függvény, amely a `Special:Hello` oldal megnyitásakor meghívódik, a *„Hello, world” HTML*-t a kimenethez adja. A felhasználónak küldött *HTML* fájlt nem kell elkezdenünk, sem lezárnunk, sem máshogyan módosítanunk.
A kinézet-választás vagy a menük beállítása és más hasonló részletek miatt sem kell aggódnunk. Ügyelnünk kell viszont a szóközökre, mert a *PHP* programokban a kezdő `<?php` címke előtti le nem vágott szóköz gyakran fura, a módosított fejlécek miatti hibaüzeneteket okoz.
Végül a következő sorral regisztráljuk a kiterjesztésünket és speciális lapunkat a *LocalSettings.php* fájlban:

```
require_once("extensions/
➥SpecialHello.php");
```

Ha a fenti sort a helyére rakjuk, webhelyünknek lesz egy `Hello`-ként megjelenített `Special:Hello` lapja a `Special::Specialpages` alatt.

Összefoglalás

A *Wikipedia* komoly figyelmet kapott az elmúlt hónapokban, eredményei pedig egy megbízható, semleges, önkéntes alapon fejlesztett enciklopédia létrehozása terén továbbra is megkérdőjelezhetetlenek. A rendszer hatékonyan helytállhat olyan szervezeteknél is, ahol nagy számú résztvevő közreműködésével kell információkat összegyűjteni. Ha egy centralizált információtárolót akarunk létrehozni decentralizált módon, a wiki mindenképpen kitűnő megoldás.
A *MediaWiki*, bár messze nem az egyetlen wiki megoldás, de letisztult, megbízható és mind az adminisztrátorok, mind a felhasználók számára könnyen kezelhető. Az új dinamikus oldalak, azaz az úgynevezett speciális lapok létrehozása persze a *MediaWiki* működésének alaposabb ismeretét igényli, azonban a kezdet sem nehéz, és igenis használnát vehetjük a *MediaWiki* nyújtotta lehetőségeknek.

Linux Journal 2006., 144. szám

Reuven M. Lerner hosszú ideje web- és adatbázis-tanácsadóként dolgozik, emellett PhD-s oktatástudomány (Learning Sciences) hallgató a Northwestern Universityn (Illinois állam, Evanston). Feleségével és Amotz David névre hallgató kisfiával él.

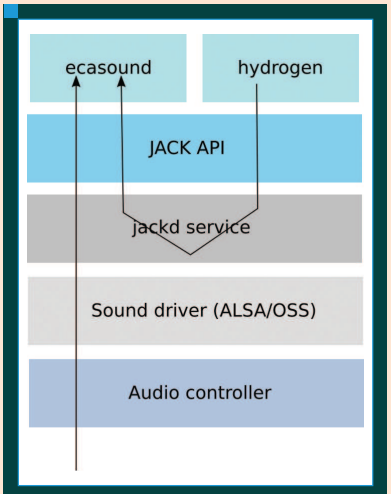
Hangstúdió – akár konzolon is

A dobozos megoldás túl drága? Nem elég rugalmas? Nem elég érdekes? Miért ne használhatnánk Linuxot hangstúdiónkban?

Legyen akár metál, jazz, barokk vagy bármi más, egyre jellemzőbb, hogy a zeneszerzők és az előadók egyben hangmérnöki, produceri és terjesztői munkát is ellátnak.
A linuxos zenei alkalmazások minősége és képességei már most is nagyon jók és egyre fejlődnek. Az alsó és felső kategóriás hangeszközök támogatottsága is egyre jobb az utóbbi időkben. Akik értékelik a nyílt forráskód és a ***Linux*** rugalmasságát, azoknak számára lényegtelen hogy ezen a téren domináns lesz-e. Mindenesetre a ***Linux*** is alkalmas jó minőségű felvételek készítésére.
A cikkben szó esik arról, hogyan készíthetünk egyszerűen többsávos felvételeket. A linuxos hagyományoknak megfelelően számos apró programot fogok bemutatni, melyek egy-egy részfeladat megoldását végzik el. Ennek végeredményeként egy digitális hangfeldolgozó környezetet fogunk kapni. Az eszközöket a cikkben leírtak szerint használva minimálisra csökkenthető a hibalehetőség.
Az ***Ecasound***-ot és ***JACK***-et mutatom be, de a ***Hydrogen*** dobgépről is esik szó, igaz csak nagy vonalakban. Semmilyen ***ALSA*** vagy ***OSS*** specifikus dolgot nem használunk, így bármelyiket használhatjuk. Egyébként ezek a technikák és programok működnek ***MacOSX***-es ***CoreAudio*** alatt is.
Mindezt csak azok kedvéért említem, akik letértek a megvilágosodáshoz vezető útról (csak viccelek).
Az *1. ábra* az adatáramlást mutatja a különböző részek között.

Felszerelés

A cikk példáihoz bármilyen hangkártya megfelel. Még a linuxos laptopom integrált ***Intel i8x0*** hangkártyájával is egész elfogadható felvételeket készítettem. Természetesen az alsó és a közép illetve felsőkategóriás eszközök között érezhető a különbség.
Szükségünk van egy ***Linux*** disztribúcióra is. Ha gondot okoz a disztribúciónkhoz a ***JACK*** és az ***Ecasound*** beszerzése, próbáljuk ki az ***AGNULA***-t. Persze a legtöbb mai disztribúció tartalmazza ezeket a programokat.
Keverő használata ajánlott. Egy olcsó keverő rugalmasabbá teszi a stúdiónkat és talán a minőség is javul. Valószínűleg szükség lesz egy mikrofon előerősítőre vagy egy impedancia illesztőre is.
Érdemes a linuxos gépbe dugni a fejhallgatót (*2. ábra*). A legtöbb keverő persze rendelkezik olyan bemenettel is, ahova a hangkártya kimenetét csatlakoztathatjuk. Ezáltal lehetővé válik feldolgozás előtt és után belehallgatni. Ez a megoldás is megfelelő.

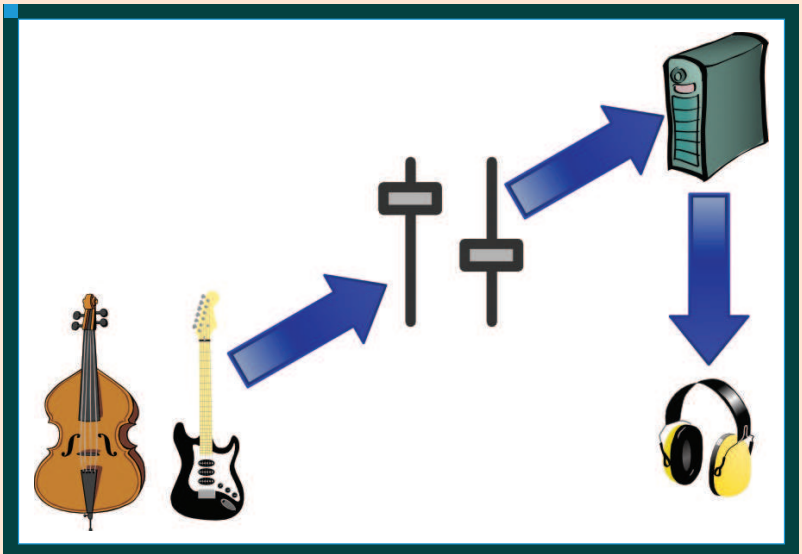


1. ábra Logikai adatáramlás

Szükséges ismeretek

Nem feltételezek komoly tudást, azonban az alábbiakra szükség lesz:

- ***Linux***, beállított és kipróbált hangkártyával.
- Tudjon az olvasó forráskódot letölteni és lefordítani, illetve az esetleges függőségeket feloldani.
- Bármilyen zajkeltő eszköz ismerete (például gitár, cselló, macska, vagy bármi más).



2. ábra Eszközök közötti adatáramlás

Jackd indítása

A *JACK* a *JACK Audio Connection Kit* rövidítése, amely *API* és szolgáltatás egyben. Lehetővé teszi számos *POSIX* kompatibilis rendszeren a hangfeldolgozó programok virtuális összekötését. A *JACK* tervezésekor a kis jelkésleltetésen volt a hangsúly.
A cikk példái közül pár *JACK* nélkül is ugyanolyan jól megy. Személyes tapasztalatom azonban az, hogy a valósidejű feldolgozás miatt jóval kisebb a jelkésleltetés, mint anélkül.
Ez a különböző programok összekapcsolásánál jön jól, mint például az *Ecasound* és a *Hydrogen*.
Ahhoz, hogy egy alkalmazás képes legyen a *JACK*-et használni, a *JACK API* jelenlétében kell lefordítani, illetve a program futtatása előtt el kell indítani a `jackd`-t. Amelyik disztribúció tartalmazza a *JACK*-et, annak valószínűleg az alkalmazásai a *JACK API* jelenlétében kerültek lefordításra. Ha mégse így történt, úgy fordítsuk le újra, az alkalmazás leírását követve.
A *JACK* szolgáltatás futtatásához futtassuk le az alábbi parancsot:

```
jackd -R -d alsa
```

Az `-R` opció valós idejű privilégiumot biztosít, a `-d alsa` hatására pedig az *ALSA* rendszert fogja használni. Akik még mindig az *OSS*-t használnak, azok `-d oss` opcióval indítsák, a *MacOSX* tulajdonosok pedig `-d coreaudio`-val.

Minden hangrendszer tovább konfigurálható, az opciók a `-d alsa --help` paranccsal kérdezhetőek le.

Hangpróba és jelszintek beállítása

Mielőtt túl mélyre ásnánk és elkezdenénk a hangfélvételt, állítsuk be a rendszert és a jelszinteket. A jó hír, hogy csupán csatlakoztatni kell a hangszert a keverőhöz vagy leülni a mikrofon elé és játszani.
Kezdésként állítsuk be a keverőt, hogy átlagos jelszint 0 környékén legyen és lehetőleg kerüljük a gyakori túlvezérlést (átmegy a pirosba).
Ha végre sikerült beállítani a keverőt, kössük össze a számítógéppel és ellenőrizzük itt is a be és kimeneti jelszinteket:

```
ecasound -i jack_auto -o null
➥-ev
```

A `-i jack_auto` opció hatására az *Ecasound* a *JACK*-től kapja a bemenetet. Minthogy még nem futtatunk más *JACK*-et használó programot, így ez jelenleg a hangkártya vonalbemeneti jelét adja. A `-o null` opció közli az *Ecasound*-dal, hogy nem vagyunk kíváncsiak a kimenetre.
Az `-ev` kapcsolóval nyomon követhetjük az amplitúdó változásokat, a `-c` kapcsoló pedig interaktív módban indítja az *Ecasound*-ot.
Kis szerencsével csupán pár hasznos információt kapunk hibaüzenet és figyelmeztetés nélkül.

Ütős hangszer esetén tüske jelentkezik a felvételben. Jelszint ellenőrzés közben tehát minden olyan technikát érdemes kipróbálni, amit később használni szeretnénk. Később így már nem érhetnek meglepetések. A kilépéshez használjuk a *Ctrl-C*-t. Valami ilyesmi kimenetet kapunk:

```
....
(audiofx) Peak amplitude,
➥period: pos=0.30495
➥neg=0.26996.
(audiofx) Peak amplitude, all :
➥pos=0.30495 neg=0.26996.
(audiofx) Clipped samples,
➥period: pos=0 neg=0.
(audiofx) Clipped samples, all :
➥pos=0 neg=0.
(audiofx) Max gain without
➥clipping, all: 3.27926.
(audiofx) -- End of statistics
--------------------------------
```

Először is ellenőrizzük, hogy nem volt-e túlvezérlés (pozitív vagy negatív). Ezután nézzük meg a maximális erősítést. Ez százalékosan megadja (elméletben), hogy mikor következne be a túlvezérlés.
Az eszközünktől függően változó, hogy mikortól kezd torzulni a hang, így jobban járunk, ha megismerjük az eszközt és csak utána kezdjük feszegetni a határokat.
Mixer állítás esetén ismételjük meg az előbbi lépéseket.
Ha megfelelőnek találjuk a beállításokat, állítsuk be a fejhallgató hangerejét is.

Felvétel: egy sáv vagy koncertfelvétel rögzítése

Az *Ecasound* egy parancssoros program, mely a több sávos hangfelvételen túl másra is képes. Az *Ecasound* alapkoncepciója a lánc. Számunkra a lánc, olyan mint a rendezőpanelen elhelyezett összekötő kábel. Az egyik komponens kimenetét összeköti a másik komponens bemenetével. Az ilyen összekötő kábel egy bemenetet és egy kimenetet köt össze. Ugyanez igaz itt is.
Az *Ecasound* be- és kimenetei leggyakrabban hangfájlok vagy hangeszközök. Teljesen normális, hogy összetett láncokkal dolgozunk. Az első hangfelvételünknél *JACK* lesz a be

1. táblázat *Láncok az első hangfelvételhez*

Lánc	Bemeneti forrás	Kimenet
1	JACK	JACK
2	JACK	track1.wav

és kimenet, de természetesen le is mentjük a bemenetet ***PCM*** formátumban. Az *1. Táblázatban* látható láncok szükségesek ehhez.
Ezt az alábbi paranccsal érjük el.

```
ecasound -c -b:64 \
-a:1,2 -i jack_auto \
-a:1 -o jack_auto \
-a:2 -o track1.wav
```

Az ***Ecasound*** indítás után parancsra vár. ***T***-vel indíthatjuk a felvételt és a lejátszást, ***S***-el megállíthatjuk. Ha elrontottuk, állítsuk meg, a ***setpos 0***-val előre tekerhetjük, majd a ***T***-vel újrakezdhetjük. A ***Q***-val léphetünk ki. Mentenünk nem kell.

2. táblázat *Második sáv rögzítése az első sáv lejátszásával*

Lánc	Bemeneti forrás	Kimenet
1	track1.wav	JACK
2	JACK	JACK
3	JACK	track2.wav

A fenti parancssor magyarázata:

- `-c`: nem az automatikus feldolgozás indul el, hanem az interaktív mód
- `-b:64`: a lehető legkisebbre állítjuk a puffer méretét, így csökkentve a késleltetést
- `-a:1,2 -i jack_auto`: létrehozunk két láncot (1 és 2), mindkettő bemenete a ***JACK*** lesz
- `-a:1 -o jack_auto`: az 1-es lánc kimenete a ***JACK*** lesz
- `-a:2 -o track1.wav`: a 2-es lánc kimenete egy ***wav*** állomány lesz

3. táblázat *Láncrendszer több sáv esetén*

Lánc	Bemeneti forrás	Kimenet
1	track1.wav	JACK
2	track2.wav	JACK
3	JACK	JACK
4	JACK	track3.wav

Végeredményben a 2-es lánc rögzíti a ***JACK*** (és valószínűleg a hangkártya)bemenetére érkező anyagot a `track1.wav` állományba. Az 1-es lánc segítségével pedig a rögzítendő anyagot élőben hallgathatjuk.

Többsávos felvétel

Hacsak nem élő hanganyagot rögzít az Olvasó, úgy bizonyára szeretne többsávos felvételt. Az ***Ecasound*** képes arra, hogy új sávot rögzítsen, miközben már egy korábbi felvett anyagot lejátszik.
Hogy a második sáv felvételekor halljuk a korábbi sávot, három ***Ecasound*** láncra lesz szükségünk, amint azt a *2. Táblázat* is mutatja.

4. táblázat *Több sáv egy sávvá keverése*

Lánc	Bemeneti forrás	Kimenet
1	track1.wav	all_tracks.wav
2	track2.wav	all_tracks.wav
3	track3.wav	all_tracks.wav

Ezzel az összeállítással lejátsszuk a `track1.wav`-ot, élőben hallhatjuk, amit éppen játszunk és ugyanezt le is mentjük `track2.wav` néven. A `track2.wav` csak az új sávot tartalmazza, később kell majd a két sávot eggyé alakítanunk.
A fenti láncrendszer ***Ecasound*** parancsként így néz ki:

```
ecasound -c -b:64 \
-a:1 -i track1.wav \
-a:2,3 -i jack_auto \
-a:1,2 -o jack_auto \
-a:3 -o track2.wav
```

A további sávok rögzítése hasonló eljárás. Minden, már rögzített sávnak készítünk egy láncot, melynek bemenete az adott fájl, kimenete pedig a ***JACK***. A *3. Táblázat* erre ad példát.
Ehhez az alábbi parancssort használjuk:

```
ecasound -c -b:64 \
-a:1 -i track1.wav \
-a:2 -i track2.wav \
-a:3,4 -i jack_auto \
-a:1,2,3 -o jack_auto \
-a:3 -o track3.wav
```

Felvétel szoftveres forrásból

Nem tudok dobolni, de néha szükségem van rá. Noha a ***Hydrogen*** az legjobb dob gép ***Linux*** alá, amit láttam, még nem érte el a bűvös 1.0-ás verziót és még nem tökéletes. Az egyik hibája az általam használt verziónak a *wav* fájlba történő mentés. Szerencsére a ***Hydrogen*** képes együttműködni a ***JACK***-el.
A ***JACK***-et és az ***Ecasound***-ot az alábbiak szerint konfiguráljuk, amennyiben ***Hydrogen***-hez hasonló alkalmazás hangját szeretnénk rögzíteni:

1. A ***Hydrogen***-ben a kimenetet állítsuk be a ***JACK***-re.
2. Állítsuk a ***Hydrogen***-t zene módba a kotta mód helyett.
   Futtassuk az alábbi parancsot:

```
ecasound -a:1 -i jack_auto -o
➥drum_track.wav -G
➥jack,ecasound,recv
```

4. Kattintsunk a ***Hydrogen*** lejátszás gombjára.

A parancs létrehoz egy ***Ecasound*** láncot, mely a ***JACK***-et tekinti bemenetnek és a `drum_track.wav`-ot kimenetnek. A `-G jack,ecasound,recv` utasítja az ***Ecasound***-ot, hogy figyelje a ***JACK*** start parancsát, mely akkor következik be, ha a ***Hydrogen***-ben lenyomtuk a lejátszás gombot.
Körülbelül 1 másodpercet vesz igénybe, mire az ***Ecasound*** elindul a start parancs észlelése után, így a fájl elején lesz egy kis csend.
Ha az Olvasó nem rendelkezik csalhatatlan ritmusérzékkel, úgy érdemes előbb a szoftveres forrásokat rögzíteni. Sokkal nehezebb a szoftveres forrásokat szinkronizálni – például a dobgépet – egy már meglévő felvétellel, mint fordítva. Ez az esetleges ***MIDI*** sávokra is vonatkozik.

Sávok keverése egy sávvá

Mostanra már jó pár *wav* állományunk van, amelyek az egyes rögzített sávokat tartalmazzák. ***Ecasound***-ot, ***SoX***-ot vagy akár ***Audacity***-t is használhatunk effektek hozzáadásához vagy apróbb korrekciókhoz, módosításokhoz.
Ha az összes sávot megfelelőnek találjuk, egy hangállománnyá keverhetjük. A különálló sávok eggyé keverése magától értetődő. Minden *wav* állományhoz létrehozunk egy láncot.
Az ***Ecasound*** ebben az esetben egyszerűbb megoldást biztosít. Az `all` láncot használva az összes bemenet egy kimenetbe irányítható. Jelen esetben `all_tracks.wav` fájlba:

```
ecasound -a:1 -i track1.wav \
-a:2 -i track2.wav \
-a:3 -i track3.wav \
-a:all -o all_tracks.wav
```

A végeredmény meghallgatása:

```
ecasound -a:1 -i all_tracks.wav
➥-o jack_auto
```

Természetesen fájlba írás előtt meg is hallgathatjuk:

```
ecasound -a:1 -i track1.wav \
-a:2 -i track2.wav \
-a:3 -i track3.wav \
-a:all -o jack_auto
```

Számos effekt végrehajtható a láncokon a fájlba írás előtt, mint például a visszhang, dinamikakompresszió és erősítés. Késleltetést vagy úsztatást is adhatunk egy-egy sávhoz, akár zajszűrést is. Mindez persze meghaladná ezen cikk terjedelmét.

Összefoglalás

Bemutattuk, hogyan készíthető egyszerű többsávos felvétel linuxos hangfeldolgozó programokkal. Megismertük a ***jackd***-t, majd az ***Ecasound***-ot, megmondhattuk honnan vegye a bemenetet, hová irányítsa a kimenetet, legyen az akár egyszerű, akár többsávos felvétel.
Minden sávot külön *wav* állományként rögzítettünk. Így bármely hangfeldolgozó programmal módosíthatunk rajta a keverés előtt, de keverés után is módosítható. Közismert hangfeldolgozó alkalmazás például az ***Ecasound***, a ***SoX*** és az ***Audacity***.
Csak érintőlegesen tárgyaltuk a témát. Jó esetben azonban ez stabil alapokat ad az Olvasóban rejtőző kreatív zseni kibontakozásához.

Linux Journal 2006., 151. szám

Matthew Geddes hobbija zene és a Linux. Szerencséjére, a hivatása is. Ha nem saját számait játssza, akkor bármit meghallgat Bach-tól és Son House-tól kezdve egészen Rachel Singleton-ig és A norexia Nervosa-ig. Ezen az email címen érhetjük el: lj@musicalcarrion.com vagy a http://www.musicalcarrion.com/ oldalon.

KAPCSOLÓDÓ CÍMEK

A cikk forrása:
➲ http://www.linxjournal.com/article/9269

Költözzünk Blog-Citybe!

A Blog-City.com egy Linux-alapú, funkciógazdag webnaplózó rendszer, mely hatékony és nagymértékben testre szabható blog szolgáltatást nyújt.

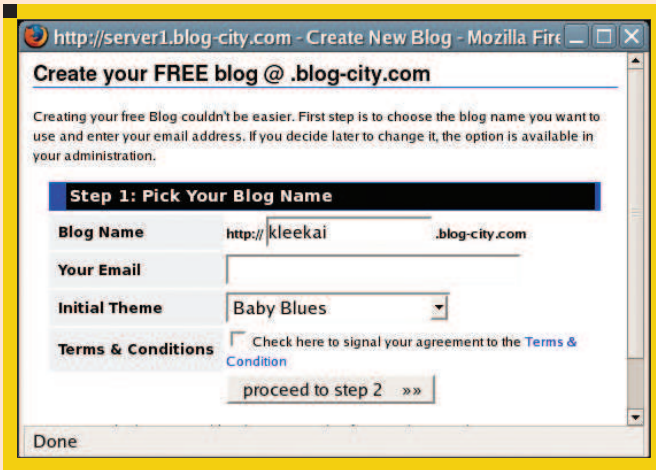
A *Blog-City.com* az interneten elérhető, egyszerre ingyenes és fizetős blogoldal. Az oldalt *Red Hat Linux 9*-et futtató hat darab kétprocesszoros *blade Dell PowerEdge Blade* kiszolgáló szolgálja ki. A *Blog-City.com BlueDragon* alatt a *Java JDK 1.4* valamelyik változatát futtatja (➲ *www.newatlanta.com/products/bluedragon*). A *BlueDragon* ugyanaz a *CFML (ColdFusion Markup Language)* alkalmazás, amely a világ ötödik legnagyobb webhelyét, a *MySpace.com*-ot is működteti.

A *Blog-Citynek* nincs statikus tartalma. A lekért weboldalakat dinamikusan építi fel egy *MySQL 4.1.x* adatbázis adataiból, amely biztonsági és adminisztrációs okokból replikálva van. Az elérési idő csökkentésére gyorsítótár szerverként *MySQL 5.0*-t használ, az összes elérési és hivatkozási statisztikához pedig *MySQL 4.1.x*-et. A terhelés megoszlik e külön adatbázis kiszolgálók között, így a különböző elérések nem blokkolhatják egymást.

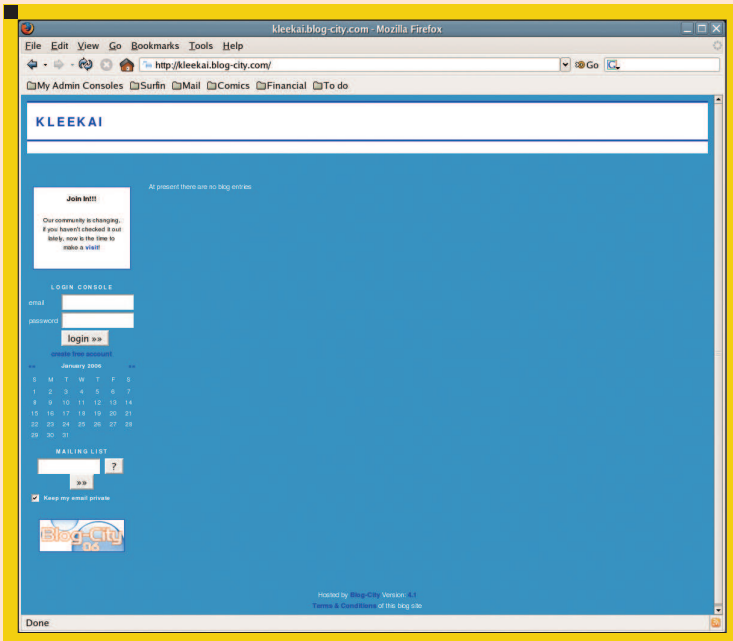
A webszolgáltatást az *Apache 1.3.x* nyújtja a *mod_backhand*-en keresztül kommunikálva a *BlueDragonnal* (➲ *www.backhand.org/mod_bachhand*). Ezt egy a cég által fejlesztett és közreadott *Java* könyvtárral teszi, amely az alkalmazáskiszolgáló terheléselosztását biztosítja. A *mod_backhand* modul alapértelmezetten csak a webszerver terheléskiegyenlítését végzi.

A *Blog-City* további minifarmok igénybevételével biztosít *RSS* szolgáltatást és kezeli a keresőmotorok lekéréseit. A levélszemét szűrését és a túlterheléses támadások kivédését, valamint a hibakövető-rendszert is saját fejlesztésű programok biztosítják.

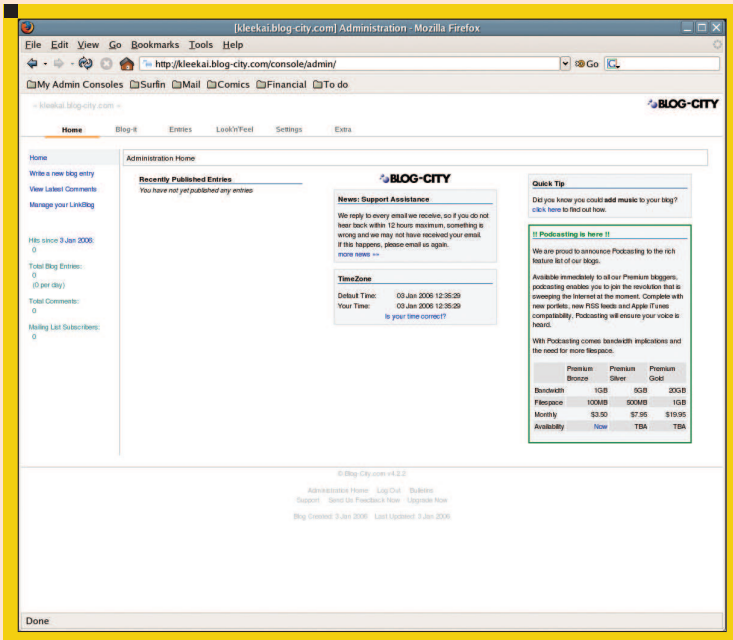
1. ábra A Blog-City főoldala



2. ábra Új Blog-City webnapló létrehozása



3. ábra A Kleekai blog kezdőoldala



4. ábra A Blog-City fő adminisztrációs oldala

Mindent egybevetve a ***Blog-City*** egy időzónákon és kontinenseken átnyúló rendszer.

Webes napló létrehozása

Mielőtt megpróbálnánk létrehozni saját naplónkat ***Blog-Cityben***, próbáljunk ki néhány ***akármilyenszó.blog-city.com*** formátumú ***URL***-t, hogy lássuk, melyek foglaltak (például ***kleekai.blog-city.com***). Ha találtunk egyet, ami még szabad, nyissuk meg a ➲ ***www.blog-city.com***-ot (*1. ábra*) és keressük meg a ***Free Signup*** (ingyenes regisztráció) dobozt. Írjuk be oda a leendő webnaplónk nevét. Az előző példa alapján ez a ***„kleekai"*** lenne. Kattintsunk a nyílra, ekkor előugrik az új blog létrehozása ablak (*2. ábra*). Csináljuk végig a szükséges lépéseket.

Kattintsunk a blogunkra mutató hivatkozásra, amely megtalálható a regisztrációs oldalon és az ellenőrző e-mailben is. Ha ez megtörtént, blogunk kezdőlapját kell látnunk (*3. ábra*).
Adjuk meg e-mail címünket és jelszavunkat a bejelentkezőkonzolon, és kattintsunk a ***sign-in*** gombra.
Ha eljutottunk a fő adminisztrációs oldalra (***Administration Home***, *4. ábra*), mentsük el könyvjelzőként. Itt változtathatunk a naplónkon, és itt nézhetjük meg a statisztikát is. A ***UNIX***-hoz hasonlóan ezen a felületen is több módon elérhetők ugyanazok az opciók. A cikkben mindegyiket csak egyféleképpen mutatjuk be.
A lap tetején található hivatkozások a következő helyekre mutatnak:

- ***Home***: a fő adminisztrációs oldal.
- ***Blog-It***: új blogbejegyzés létrehozása.
- ***Entries***: itt nézhetők meg és törölhetők a bejegyzések, valamint azt is tudhatjuk meg, egy-egy elemet hányszor olvastak és hány hozzászólást fűztek hozzá. A ***Blog-City*** különféle bejegyzéstípusokat támogat, többek között rendes blogbejegyzést, elemzést (***Review***), képgalériát (***Photo Albums***), eseményeket (***Events***) és podcastokat.
- ***Look'n'Feel*** (kinézet): ez a lap a webnaplónk elrendezésének, a témáknak, fejléceknek, menüknek és hasonlóknak beállítására szolgál.
- ***Settings***: itt módosíthatók a személyes és a blogbeállítások.
- ***Extra***: hozzáférés a statisztikákhoz, a felhasználói fiók adataihoz, stb.

Annak függvényében láthatók baloldalon az alábbi menük, hogy fizetős vagy ingyenes felhasználói fiókot hoztunk-e létre. A bal felső hivatkozások ide mutatnak:

- ***Home***: a fő adminisztrációs oldal.
- ***Write a new blog entry***: új blogbejegyzés létrehozása:
- ***View Latest Comments***: itt olvashatók a legfrissebb hozzászólások, a bejegyzések, amelyekhez tartoznak, a hozzászólások megjelenésének ideje, és itt törölhetjük őket, válaszolhatunk rájuk, vagy megjelölhetjük őket levélszemétként.

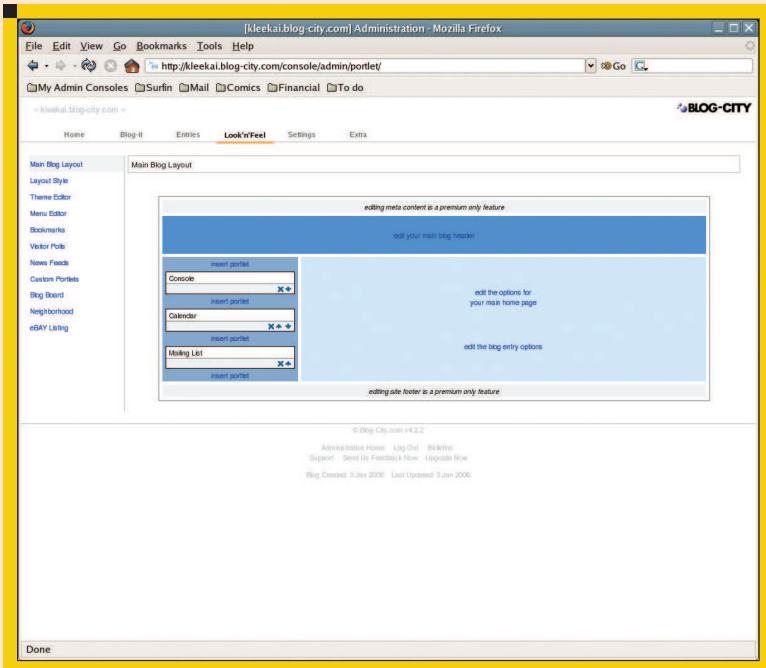
- *View Latest Trackbacks*: (csak a fizetős változatban) trackback-eket is létrehozhatunk ahelyett, hogy a hozzászólásokra válaszolnánk. Ezek tulajdonképpen hozzászólások a saját blogban. Ha rákattintunk erre a hivatkozásra, megváltoztathatjuk a trackback beállításokat (*Change Your Trackback Options*), és kérhetjük, hogy a *Blog-City* e-mailben jelezze, ha új trackback jön létre.
- *View Latest Statistics*: (csak a fizetős változatban) a saját statisztikák (*Statistics*) oldalra mutat. Itt megnézhetjük, hogy melyik blogbejegyzésünket olvasták a legtöbben, melyikhez szóltak hozzá a legtöbben, az egyes hónapokban hányszor nézték meg az oldalt és milyen böngészővel, milyen oldalak hivatkoztak blogunkra, valamint, hogy melyek a legkeresettebb podcastjaink.
- *View Podcast Statistics*: (csak a fizetős változatban): legnépszerűbb podcastjaink.
- *Manage Your LinkBlog*: a *LinkBlog* hivatkozások gyűjteménye, melyben mindegyik hivatkozáshoz tartozik egy rövid leírás. Magát a gyűjteményt itt lehet létrehozni és tartalmát kezelni. *URL* és *RSS* csatorna is rendelkezésünkre áll a tartalom eléréséhez.

A baloldali menük alatt található még:

- *Hits since date*: találatok az adott időponttól. Ez lehet a blog létrehozásának napja, vagy az az időpont, amikor a találatszámlálót lenullázták. A visszaállításhoz kattintsunk a dátumra, majd az *OK* gombra. A visszaállítás más statisztikára nem vonatkozik.
- *Total Blog Entries*: bejegyzéseink száma, és a napi átlagos bejegyzésmennyiség.
- *Total Comments*: a hozzászólások összmennyisége.
- *Mailing-List Subscribers*: hány személy kap e-mailben értesítést a legfrissebb bejegyzéseinkről. A hivatkozásra kattintva láthatjuk a nem anonim előfizetőket.

A fő adminisztrációs oldal további elemei:

- *Published Entries*: a legújabb öt közzétett bejegyzés.



5. ábra A Blog-City fő blog-elrendezés oldala

- *Draft Entries*: a még közzé nem tett, piszkozatként elmentett bejegyzések.
- *Jegyzetek*. (Az *Extra -> Your Notes* menüpontban hozhatók létre.)
- A *Blog-City* személyzet által küldött legfrissebb levelek.
- A helyi időzóna ellenőrzése (és beállítása).
- *Gyorstipp*: mert mindig lehet valami újat tanulni.
- *További információk*: új szolgáltatások (például a podcasting) és a tarifák.

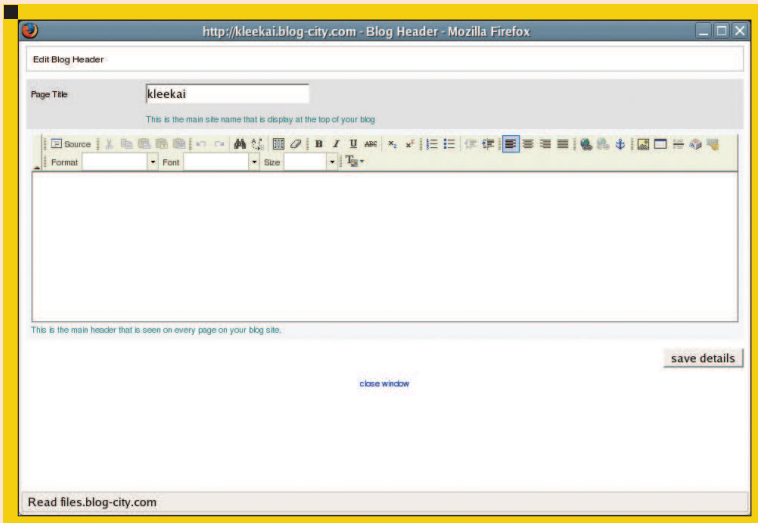
A *Settings* fülre kattintva további lehetőségeket találunk, többek között az *RSS* beállításokkal kapcsolatban. Legfelül, a bal sarokban levő hivatkozás saját naplónkra mutat, innen az bármikor elérhető. Ha próbaképp innen nyitogatjuk meg az oldalt, az nem befolyásolja a látogatói statisztikákat. Érdemes tehát ezt használni, nehogy túlságosan elbízzuk magunkat.

A megjelenés fontos

A legtöbben még a tartalom létrehozása előtt szeretnénk látni, hogy fog a blogunk kinézni. A *Look'n'Feel* fülön keresztül juthatunk a fő blog-elrendezés oldalra (*5. ábra*).

Itt címkével ellátott szemléltetőképeket találunk. Azok a címkék, amelyekre rákattinthatunk, a módosítható elemeket jelölik, a többi a fizetős felhasználói fiókok privilégiuma. Az oldal bal szélén található hivatkozások:

- *Main Blog Layout*: a megjelenési kép, amelyet éppen most nézünk.
- *Custom Home Page*: (csak a fizetős változatban) egy teljesen új kinézetet tervezhetünk.
- *Layout Style*: sablonok alapján kiválasztható az oldal elrendezése. A fizetős változat jóval több lehetőséget kínál.
- *Theme Editor*: itt változtatható az oldal témája.
- *Menu Editor*: (csak a fizetős változatban) navigálási célokra használható menük létrehozása oldalunkon.
- *Bookmarks*: könyvjelzővel megjelölt hivatkozások létrehozása és rendezése.
- *Visitor Polls*: (csak a fizetős változatban) szavazások hozhatók létre portletekként.
- *News Feeds*: (csak a fizetős változatban) az előfizetendő RSS csatornák beállítása, majd az *RSS* megjelenítése egy aloldalon (mint például a kleekai.blog-city.com/newsfeed) vagy portleten.



6. ábra A Blog-City blog fejléc oldala

- ***Custom Portlets***: (csak a fizetős változatban) a ***HTML*** szerkesztő ablakban létrehozhatjuk és kezelhetjük saját portletjeinket.
- ***Blog Board***: hozzáférhetünk ***Blog Boardunk*** tartalmához, kiüríthetjük, vagy késleltetést állíthatunk be a hozzászólásokhoz. (A blog board egy portlet, ahol a felhasználók üzeneteket hagyhatnak vagy cseveghetnek.)
- ***Neighborhood***: (csak a fizetős változatban) létrehozható olyan aloldal (mint amilyen a ***kleekai.blog-city.com/neighborhood*** is), amely az általunk követett blogokat listázza ki. Főoldalunkon elhelyezhetünk egy erre mutató hivatkozást.
- ***eBAY Listing***: (csak a fizetős változatban) az oldalon megjeleníthetők egy adott kategória vagy adott felhasználók legújabb termékei.

A kinézet-beállítás hivatkozásai:

- ***Edit the blog's META data*** (A blog metaadatainak szerkesztése, csak a fizetős változatban): megnyit egy ablakot, ahol blogunk metacímkéit szerkeszthetjük.
- ***Edit your main blog header*** (A blog fő fejlécének szerkesztése): megnyit egy ablakot (*6. ábra*), amelyben blogunk fejlécét szerkeszthetjük. Kezdjük az oldal címének megváltoztatásával – valószínűleg szándékunkban áll kifejezőbbé tenni, nem csak a blog nevével jelölni az oldalt. A lenyíló listában válasszuk ki, hogyan viselkedjen a fejléc, majd a formázó gombok alatt adjuk meg, mi jelenjen meg minden oldal tetején. Ez a doboz és a ***WYSIWYG*** formázási eszközei mindig ugyanúgy fognak kinézni, függetlenül attól, hogy éppen hol módosítjuk az oldal tartalmát. Ha nyers ***HTML***-lel kívánunk dolgozni, a bal fölső sarokban kattintsunk a ***Source*** gombra. Ha a változtatásokat szeretnénk látni, kattintsunk a ***Save Detailsre***, majd a főoldalon kattintsunk a blog nevére, hogy a korábban tárgyalt módon nyissuk meg blogunkat.
- ***Edit the options for your main home page*** (A fő kezdőlap opcióinak szerkesztése): megnyitja a kezdőlap opciói párbeszédpanelt, mellyel megadott viselkedéstulajdonságokat állíthatunk be a bloghoz.
- ***Edit the blog entry options*** (A blogbejegyzések opcióinak szerkesztése): megnyitja a megfelelő párbeszédpanelt, mellyel beállíthatjuk blogbejegyzéseink viselkedéseit.
- ***Edit your main blog footer*** (A blog láblécének szerkesztése, csak a fizetős változatban): megnyitja a megfelelő ablakot, ahol beállítható, mi jelenjen meg a blogoldalak alján.
- ***Insert Portlet*** (Portlet beszúrása): a portletek a blog szélénél megjelenő mókás szerkentyűk. Ha rákattintunk erre a hivatkozásra, megjelenik a portlet beszúrása párbeszédablak, amellyel új portleltet szúrhatunk be egy konkrét, kiválasztott helyre. Ha szeretnénk a portletet fölfele vagy lefele elmozgatni, kattintsunk a fölfele vagy lefele gombokra. A portlet eltávolításához kattintsunk saját dobozának X gombjára. A fizetős felhasználók jóval több portletből válogathatnak, mint az ingyenes változatot használók.

Mivel sokan szeretnék a blog-témákat testreszabni, bemutatjuk, hogy ez hogyan történik a ***Blog-Cityben***.
A ***Look'n'Feel*** oldalon keressük meg a lehetséges témákat. Itt előnézeti képet találunk a mindegyik ***Blog-City*** tag rendelkezésére álló témákról. Kattintsunk a ***Community Themes*** részre, válasszunk ki egy témát a listáról, vagy a nyilak segítségével nézzük végig mindegyiket. Ha egy téma csak egy fehér dobozként jelenik meg, akkor az még elérhető, csak nincsen róla előnézeti kép. Ha megtaláltuk a legszimpatikusabbat, kattintsunk a ***Use This Theme***-re. Majd, a ***Save Theme*** részben jelöljük be a ***Make this theme my live/public theme*** (Legyen ez a valódi/nyilvános témám) jelölőnégyzetet, és mentsük a változásokat (***Save Current Theme***). Töltsük be blogunkat, látni fogjuk, hogy az új témával tűnik föl.
Kiválaszthatunk egy témát pusztán alapnak, és tartalmát magunk megszerkeszthetjük. Kövessük az előbbi instrukciókat, de utána a ***Your Themes*** részben válasszuk ki a témát a lenyíló listából, és kattintsunk a ***Load Theme*** gombra. Ha ez megtörtént, a témához felhasznált összes stílus megjelenik a témaszerkesztő bal oldalán. A ***Preview Theme***-re kattintva a téma előnézetét láthatjuk. Ha az egérmutatót az előnézet fölé visszük, előugró ablakok mutatják melyik stílus címke melyik részhez tartozik.
A stílusbeállítások szerkesztéséhez bal egérgombbal kattintsunk rá, és előbukkannak az adott stílushoz tartozó ***CSS*** beállítások (*7. ábra*), melyeken változtathatunk. A bal oldalon közvetlenül begépelhetünk dolgokat, de

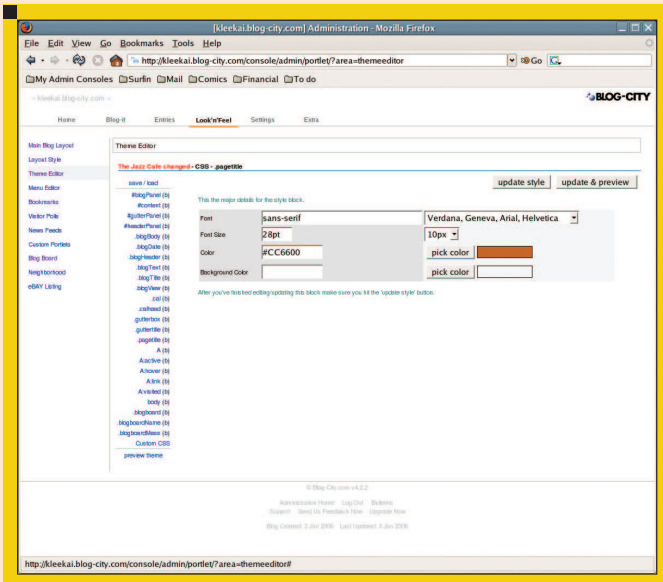
a jobboldali listáról is választhatunk a lehetőségek közül. Ha készen vagyunk a variálással, kattintsunk az ***Update & Preview*** gombra, így láthatunk egy előnézeti képet a változtatások utáni panelről. A ***Return*** gombra kattintva visszajutunk a stílus szerkesztéséhez, így további módosításokat eszközölhetünk.
Ha készen vagyunk a téma szerkesztésével, kattintsunk a ***save/load*** gombra a stíluslista tetején. Így visszajutunk a ***Theme Management*** főoldalára. Változtassuk meg a téma nevét a ***Current Theme Name***-ben, kattintsunk a dobozra, így a téma a valódi/nyilvános témánkká válik, majd kattintsunk a ***Save Current Theme*** gombra. Blogunk frissítésekor az új beállítások lesznek érvényben.

Naplóbejegyzések létrehozása

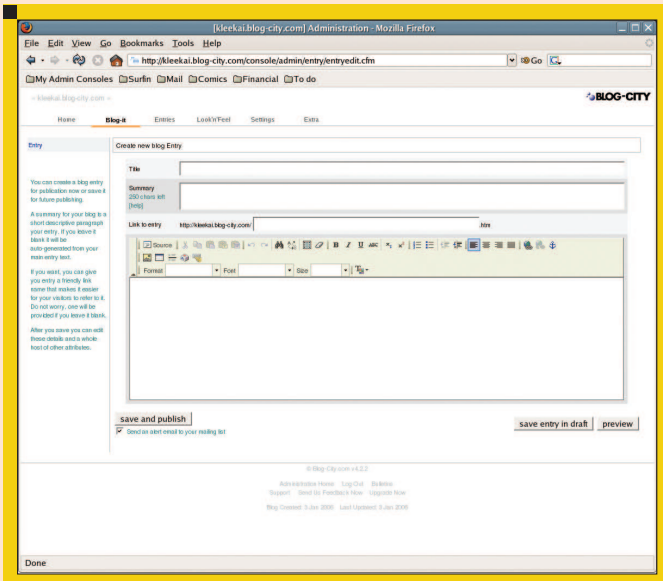
Most már blogunk szépen be van állítva, írni is kéne bele valamit. Kattintsunk az oldal tetején található Blog-it fülre, így arra az oldalra jutunk, ahol blogbejegyzéseket tudunk létrehozni (*8. ábra*). Töltsük ki a mezőket:

- ***Title***: a bejegyzés címe.
- ***Summary***: közvetlenül kitölthető, de a program automatikusan beírja ide bejegyzésünk első 250 karakterét. Nyilván előfordul ilyenkor, hogy levágja a mondat végét.
- ***Link to entry***: lehetőség van arra, hogy a ***Blog-City*** egy megadott ***URL***-t használjon a címben szereplő ***URL*** helyett - kapóra jön, ha később valakinek le kell diktálnunk a hivatkozást.
- A szövegdoboz: ide írhatjuk magát a bejegyzést.

Ha befejeztük, kattintsunk a ***Save and publish*** gombra, amennyiben a bejegyzés készen áll arra, hogy közszemlére tegyük. Ha még nem végleges, a ***Save entry in draft*** gombra kattintva piszkozatként menthetjük, illetve a ***Preview*** gombbal megnézhetjük, hogyan is fog mutatni. Ha valaki már feliratkozott levelezőlistánkra (a létszámot az ***Extra -> Mailing List*** menüpontban ellenőrizhetjük), a bejegyzés elküldése előtt érdemes bejelölnünk az „email értesítés küldése a levelezőlistának” jelölőnégyzetet. Ha a bejegyzést piszkozatként mentjük, elküldhetjük később a ***Home*** fülre kattintva, itt a piszkozatok között (***Draft Entries***) megtaláljuk, és a bejegyzés címére, utána a ***Publish Entry***-re kattintva közzétehetjük. Az ***Entries*** fülre kattintva más típusú bejegyzéseket is megadhatunk. Az ***entry***, azaz a bejegyzés általában blogbejegyzésre utal, de szemléket, fényképeket és eseményeket is publikálhatunk. Az adminisztráció rész mindig frissül, ha a ***Blog-City*** személyzete új funkciókkal jelentkezik. Időről időre fedezzük fel ezeket az új lehetőségeket, nehogy lemaradjunk valami jóról.
Kellemes blogolást!



7. ábra A Blog-City témaszerkesztő stílus oldala a Jazz Cafe Theme's .pagetitle CSS beállításokhoz



8. ábra A Blog-City Create New Blog Entry (Új bejegyzés létrehozása) oldala

Linux Journal 2006., 144. szám

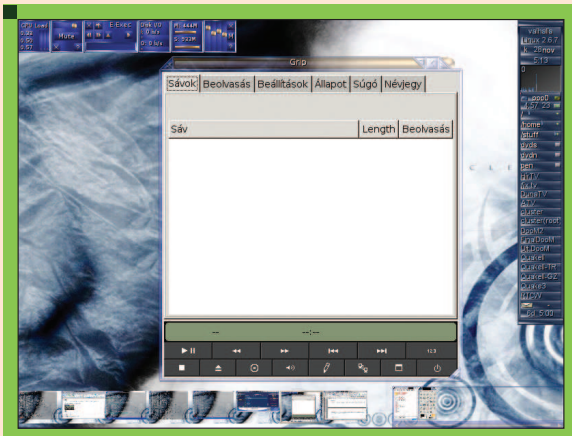
Dee-Ann LeBlanc (dee-ann.blog-city.com) Linuxra és minihuskykra szakosodott kitüntetett műszaki író és újságíró. Örömmel fogad hozzászólásokat a dee@renaissoft.com-on.

Grip – A hasfelmetsző újabb áldozata

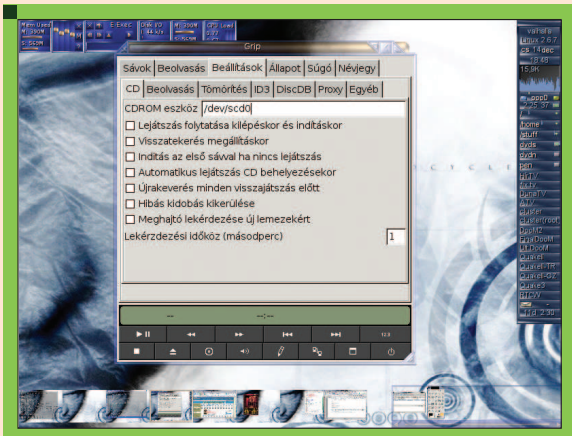
Biztosan sokan voltunk már úgy, hogy a CD-k folyamatos cserélgetését nyűgnek éreztük. Néha csak egy-egy dal miatt rakunk be egy lemezt, majd kivesszük, hogy egy másikat hallgathassunk. Ilyenkor jön a korszakalkotó megvilágosodás: alakítsuk át CD-nk tartalmát zenefájlokká és tároljuk tartalmát a merevlemezünkön! Hogy ez miként lehetséges, abban segítségünkre lehet a Grip, illetve ez a cikk.

Egy kis történelemóra...

Mai világunkban a nagy helyet foglaló, illetve nehézkesen kezelhető tárgyak kora leáldozóban van. Efelől nincs kétségem, bár ez közel sem jelenti azt, hogy az „elavult" dolgok rosszak lennének. Mindenki mindenből egyre kisebbet akar, de persze azért ez a kisebb tárgy legalább kétszer annyit tudjon, mint az előző nagyobb változat. Hogy milyen tárgyakról is van szó? Nyilván triviális lehet a cikk bevezetőjéből adódóan. De egy kicsit kanyarodjunk el a ***CD-ktől***, menjünk vissza a 60-70-es évekbe. Lehet, hogy nem mindenki tudja, de akkoriban még a számítógépek szobanagyságúak voltak, ráadásul a töredékét sem tudták annak, amit például egy - manapság inkább már csak szeméttelepeken megtalálható – ***386-os***, vagy ***486-os PC*** tudott évekkel ezelőtt. Ahogy telt-múlt az idő, a gépek egyre kisebbek lettek, viszont a tudásuk egyre nőtt. Van, ami sosem változik, de a számítástechnika egy nagyon képlékeny ágazata korunknak. Természetesen nem azt mondom, hogy éveken belül majd valóság lesz a ***Mátrix*** vagy a ***Terminátor***, de ki tudja. Rengeteg olyan fejlesztés történik, amiről nekünk, „földi halandónak" fogalmunk sincs.
Persze nem csak a számítástechnikában, hanem például a zeneiparban is bődületes változások álltak elő. A fejlődést ott sem lehet megállítani. Igazából nem a zenei stílusokra gondolok (amikből már újat kitalálni szerintem lehetetlen, legalábbis, ha valami újat kitalálnak, az gyorsan elsikkad és visszatérnek a klasszikus stílusok), hanem zenei adathordozókra, illetve zenék lejátszására alkalmas eszközökre. Gondoljuk csak el, hogy dédanyáink még lepukkant és mai szemmel nézve giccses, ósdi gramofonokon hallgatták a kamarazenét, a jelenkor embere viszont már egy kisujjnyi nyakba akasztható kütyüvel hallgathat szinte bármit, amit csak akar és ami a legfontosabb, akárhol.
Tehát a gramofonok után jöttek a szintén bakelit lemezt lejátszó lemezjátszók, amikhez már külön hangfalakat lehetett csatlakoztatni. A 70-es és 80-as években új őrület következett, mégpedig a kazettás magnó. A magnókazetta már jóval kisebb volt, mint a bakelit, ráadásul több adat (zene) fért rá. Praktikus megoldásnak tűnt akkoriban. Jómagam is magnókazettán nevelkedtem, még a 90-es évek közepén is a legfelkapottabb hanghordozónak számított.
Amíg a kazetta élte virágkorát, addig nyugatról sunyin bekúszott hozzánk egy újabb és jóval drágább dolog. Ennek a neve ***Compact Disc***, azaz rövidítve ***CD*** volt. Bizony, akkoriban ez még nagyon újnak számított és csak a kiváltságosok engedhettek meg maguknak H***i-Fi tornyot***, melyben ***CD-lejátszó*** is volt. A ***CD*** jóval flexibilisebb (hogy azt ne mondjam: kompaktabb) volt mind a bakelitnél, mind a kazettánál. Könnyű volt számokat váltani, nem kellett a tűt „saccperkábéra" tenni valahova, vagy az idők végezetéig tekerni. Ráadásul egy ***CD*** örök életű volt a lézertechnikának köszönhetően.
A bakelit elkopott/eltört, a kazetta szalagja meg szintén elkopott vagy betekerődött. Ugyanígy, a diát és a 8mm-es szalagot felváltotta a videokazetta.
Hogy egy kicsit visszakanyarodjak a számítástechnikához, bizony ott is jelentős változások voltak az adathordozókban. Kezdődött minden a lyukkártyával, majd jöttek a ***DAT-kazetták*** (szerintem a mai napig a legbiztonságosabb adathordozó, innen kapta a nevét a ***.tar***, azaz ***tape archive*** nevű kiterjesztés). Később itt is megjelentek a magnókazetták, mivel a magnószalagra nem csak zenét, hanem bináris adatot is lehet rögzíteni. Jött a nagy és a kisfloppy, majd a számítástechnika is elért a ***CD-k*** korszakához. Manapság még mindig a ***CD*** korát éljük, ez a legnépszerűbb adathordozó. Viszont megjelent az ***MD***, azaz ***MiniDisc***, illetve a ***DVD*** (***Digital Versitale Disc***), ami lassan, de biztosan át fogja venni a ***CD-k*** helyét a ranglistán, mivel azonos nagyságúak, viszont cirka 6x annyi adat fér rá. Jó pár éve a piacon vannak a ***CD-*** és ***DVD-írók***, melyekkel üres lemezeket tölthetünk meg adattal. A számítástechnika fejlődése lehetővé tette, hogy feltalálják a zenefájlokat. Természetesen már a ***Commodore-os*** időkben is voltak zenefájlok (az úgynevezett ***SID*** fájlok),



■ *1. ábra* Grip indítás után



■ *2. ábra* CD-beállítások

de ezek kezdetleges ***beep-ekből*** összeállított számítógépes zenék voltak (ettől függetlenül mindannyian szerettük/szeretjük őket).
Pár éve elérkeztünk abba a korba, amikor már teljesen mindegy, hogy milyen zenei vagy egyéb analóg vagy digitális adathordozóink vannak, mivel ezeket nem szükséges használnunk a rajtuk lévő zenék/filmek lejátszásához. A számítógép úgymond a zeneiparban is átvette a hatalmat, így egyeduralomra tört, tehát mondhatni egy univerzális médialejátszóvá vált.
Rengeteg fajta úgynevezett digitalizáló módszer létezik, például külön programok videokazetták digitalizálására (a rajtuk lévő adat számítógépre mentésére). Mi most egy olyan ***linuxos*** programot fogunk megismerni, ami lehetővé teszi azt, hogy zenei ***CD-ink*** tartalmát a számítógépünk merevlemezére helyezzük valamilyen zenefájl-konverzió segítségével.

CD-ripperek

A címben szereplő hasfelmetsző (ripper) most kivételesen nem hasakat, hanem ***CD-ket*** „szabdal". Persze ezt csak képletesen kell érteni, semmi bajuk sem lesz. A ***CD-ripperek*** a lemezen található dalokat alakítják át ***WAV-formátumba***, majd ezekből egy kódoló segítségével egy kisebb méretű zenefájlt kapunk. A rippelés általában a ***cdda2wav*** vagy a ***cdparanoia*** programok segítségével történik (persze ez ***Linux*** alatt értendő).
Jómagam még az ősrégi ***windowsos*** időkben találkoztam először a ***CD-ripper*** fogalmával, a nevezett program pedig az ***Alto MP3 Maker*** névre hallgatott. Ez az alkalmazás még csak ***MP3-akat*** tudott készíteni a ***CDDA*** fájlokból. Azóta változtak az idők és az igények, megjelentek újabb formátumok, tehát a ***CD-ripper*** programoknak is változniuk kellett. ***Windows-ra*** számtalan ilyen alkalmazást találhatunk, némelyikük ***shareware*** (fizetős), némelyik ***freeware*** (ingyenes). ***Linux*** alatt már nincs ekkora felhozatal.
Itt minden ingyenes, ráadásul minden konzolparancsokon alapszik. Van néhány jól bevált konzolos program, melyekre különféle – az egyes grafikus környezetekhez illő – skineket (felületeket) húznak. Ezek egyike a ***Grip***.

Grip

A ***Grip*** egy ***linuxos CD-lejátszó***, illetve ***CD-ripper*** program a ***Gnome desktop*** részére. Ebből rögtön kiderül, hogy a felülete ***GTK-s***, ami egy homogén, egyszerű, ámde nagyszerű kinézetet biztosít számunkra. Természetesen, mint azt már megszokhattuk, ez nem jelenti azt, hogy más grafikus felület alatt ne használhatnánk.
A jelenlegi legfrissebb verzió a 3.3.1 (ez a fejlesztői változat), ami a ***Grip*** hivatalos oldaláról letölthető.
Ezt a ➲ ***http://nostatic.org/grip/*** webhelyen találjuk. Innen akár a stabil, 3.2.0-s verziót is magunkévá tehetjük. Érdekes módon a honlapról kizárólag ***RPM-ben***, illetve forráskódban tölthető le a program, de azért ne keseredjünk el, mivel minden nagyobb disztribúció tartalmazza.
Az alkalmazás érdekessége, hogy tartalmazza a ***cdparanoia*** nevű programot, ezen felül még néhány extra funkciót is tartalmaz, amelyekről majd később fog szó esni.

Telepítés

Mivel a ***Grip*** majdnem minden nagyobb terjesztés része, ezért gyerekjáték a telepítése. Például ***Debian GNU/Linux*** alatt az

```
apt-get install grip
```

paranccsal, illetve ***Mandriva*** alatt az

```
urpmi grip
```

paranccsal telepíthető. Persze ezt forrásból is elvégezhetjük, de ebben az esetben vigyázzunk a függőségekre, illetve tüzetesen olvassuk el a forráshoz csatolt `INSTALL` és `README` fájlokat.

Használat

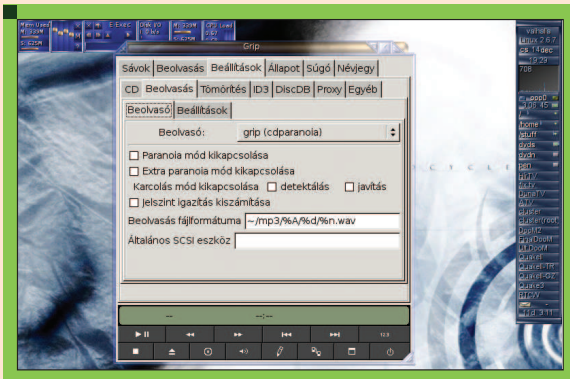
Telepítés után programunk – jó esetben – szépen bekerül a grafikus környezetünk menüjébe, tehát innen indíthatjuk. Ha ez mégsem történne meg, akkor indíthatjuk a ***Grip-et*** bármelyik xterminálból (***konsole***, ***eterm***, ***rxwt***, ***xterm***, ***aterm***, stb.) is a

```
grip
```

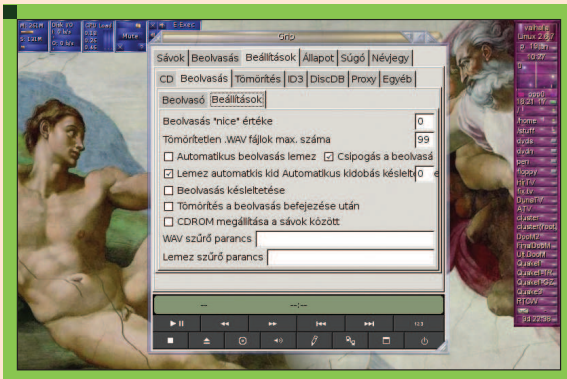
parancs begépelésével.

Kezelés és funkciók

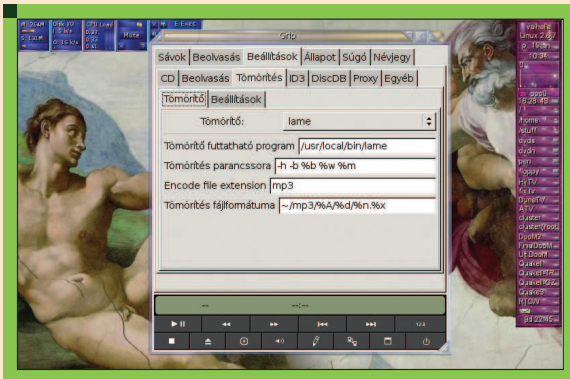
Lássuk akkor, hogy mit is tud a ***Grip***. Mint már említettem, ez egy ***CD-lejátszó***, illetve ***CD-ripper*** alkalmazás egyben, amit végtelenül egyszerű kezelni



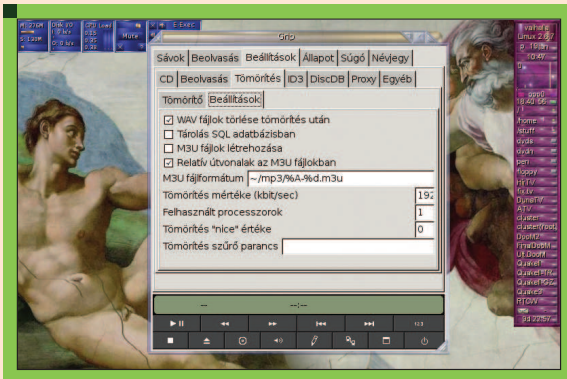
■ *3. ábra* Beolvasó beállítása



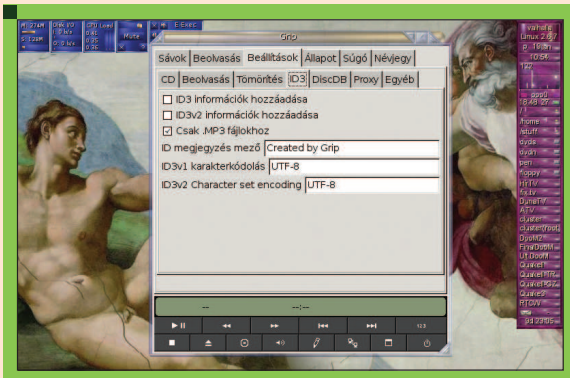
■ *4. ábra* Beolvasó konfigurálása



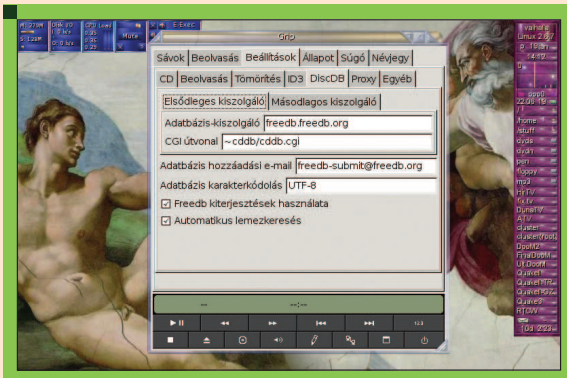
■ *5. ábra* Tömörítőprogram beállítása



■ *6. ábra* Tömörítők konfigurálása

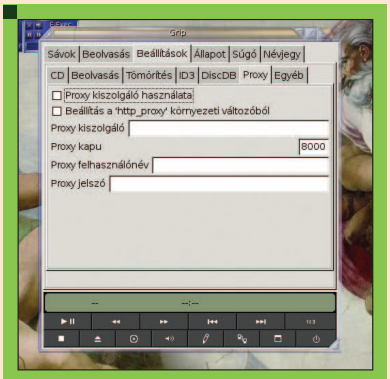


■ *7. ábra* ID3-Tag-ek finomhangolása

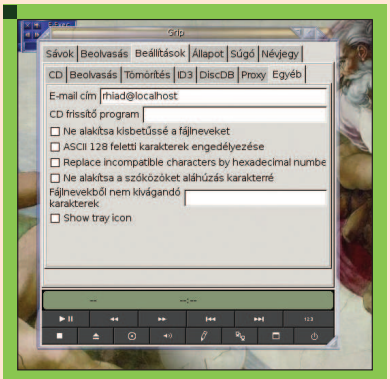


■ *8. ábra* DiscDB beállítások

a kezdőbb felhasználóknak is.
Az egész lényege a beállításokon múlik, melyeken szinte semmit sem kell változtatnunk a „gyári” alapértelmezetthez képest.
Nézzük a ***CD-lejátszóra*** vonatkozó beállításokat. Ezeket a ***Beállítások*** fülön találhatjuk.
Az első alfül, ami az ábrán látható, a ***CD-beállításainkat*** tartalmazza. Legfontosabb dolog, hogy helyesen adjuk meg a ***CD-ROM*** eszköz nevét, amelyen keresztül fogjuk lejátszani lemezeinket. Ez minden esetben */dev/eszköznév* alakú.
Többféle finomhangolásra is lehetőségünk nyílik itt, többek között: automatikus lejátszás, új *CD* behelyezése után automatikus meghajtólekérdezés, leállítás után a lemez elejére ugrás, stb.
Mind-mind kényelmi funkciók, melyeket nem taglalnék részletesen, mivel egyértelműek.
A második alfülre lépve a ***CD-beolvasó*** program beállításaival találjuk szembe magunkat.
Többféle beolvasó közül választhatunk, mint például *cdda2wav, cdparanoia, grip (cdparanoia)*.
Ez utóbbi automatikusan be van építve a ***Grip-be***, ajánlott ezt használni. Itt extra parancsokat is megadhatunk.
A beolvasó beállításait az „alfül alfülén” tehetjük meg.



■ ***9. ábra*** Proxy-beállítások



■ ***10. ábra*** Egyéb beállítások

■ ***11. ábra*** Zenei CD lejátszása

Az ábra magáért beszél. Beállíthatjuk, hogy az ***audiocd-nk*** beolvasása késlelletve legyen (így biztosan be legyen olvasva), illetve a kevés hellyel rendelkezőknek nagy segítség lehet a tömörítetlen ***WAV-fájlok*** számának a megadása.

A tömörítővel kapcsolatos beállítások a harmadik alfülön találhatók.

A ***gnormalize-ról*** szóló cikkemben már mindent leírtam az ***audiofájlok*** tömörítéséről. A ***Grip*** is ezen programokat használja, tehát például ***WAV-ból MP3-at*** a ***LAME*** nevű program csinál, és így tovább.

Ezen a fülön ezen programok elérési útját és egyéb parancsait adhatjuk meg. Megjegyezném, hogy a ***Grip*** egy remekül előrekonfigurált program, ami azt jelenti, hogy jó esetben nekünk semmit sem kell állítanunk, minden megy magától is.

A tömörítőknek is van konfigurációs paneljük.

Ez talán a legfontosabb rész a beállítások közül, ugyanis itt adható meg a meg a tömörítés minősége, azaz a bitráta. ***MP3-nál*** az úgymond „jó” minőség 128 kbit/sec-től kezdődik. Az „igen jó” minőség a 192-es bitrátánál kezdőik, a „tökéletes” hangminőség pedig a 320 kbit/sec.

A ***Grip***, mint minden valamirevaló hasonszőrű alkalmazás, képes az ***ID3-Tag-ek*** kezelésére.

Ezen beállításokban sincs semmiféle ördöngösség, bárki ki tudja tapasztalni egymaga is.

Soron következő beállítófülünk nem más, mint a ***CDDB*** (***Grip-beli*** nevén ***DiscDB***) beállításai.

Ez a kis program egy netes adatbázisból kérdezi le a lemezen található dalok címeit. Érdemes az alapértelmezett beállításokat meghagyni (➲ ***freedb.org***). Ha kedvünk szottyan, használhatunk ***proxy-t*** is, illetve egyéb célszerű és praktikus dolgokat is eszközölhetünk a legutolsó beállítófülön.

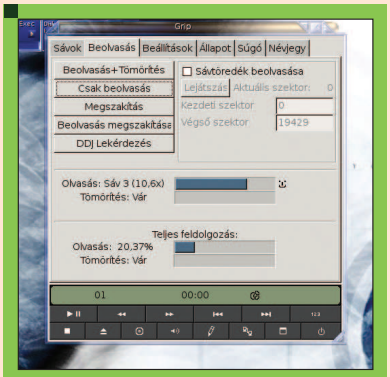
Ezek mind opcionális és triviális beállítások. Mindenki cselekedjen önmaga kénye-kedve szerint.

„No ezze' is megvó'nánk” – mondta a róka, mikor az utolsó csirkecsontot is lenyelte. Végre elkezdhetjük használni a programunkat.

Az érdemi részben vizsgáljuk először a lejátszást. A beállítások előtt volt szó a ***Grip*** indításáról (*2. ábra*). Ez a felület egyben a lejátszónk is. Amennyiben a konfiguráció során helyesen adtuk meg a ***CD-eszköz*** nevét, akkor egy zenei lemez behelyezése után egyből megjelenik a lejátszólista.

Alul találhatóak a navigációhoz szükséges gombok, gyerekjáték használni.

A ***ripper*** funkció már kissé bonyolultabb, de ez sem „egy nagy vasziszt-dasz”. Ehhez szintén szükség van az előbb említett ***zenei CD-re***, illetve a lejátszólistára. Minden egyes dalcím előtt található egy kis négyzet, mely kipipálható. Ezek arra szolgálnak, hogy kijelölhetjük a „rippelendő” dalokat. Miután ezt megtettük, nyissuk meg a ***Beolvasás*** nevezetű fület. Többfajta lehetőség közül választhatunk: beolvasás és tömörítés egyszerre, csak beolvasás, megszakítás, stb. Jelen esetben a beolvasás azt jelenti, hogy a lemezen található ***CDDA*** fájlokat a program (egy másik program segítségével) ***WAV-fájllá*** alakítja. A tömörítés pedig annyit tesz, hogy a már átalakított ***WAV-okat*** egy tömörítő (lásd ***gnormalize*** cikkem) segítségével egy újabb zenefájllá alakítja.



■ ***12. ábra*** A „rippelés”

Ha gyenge processzorral rendelkezik a gépünk, akkor mindenképp külön végezzük el a beolvasást és a tömörítést, azonban egy cirka 1 GHz-es ***Celeronnal*** felszerelt masina már nagyjából bírja a gyűrődést.

Végezetül megemlíteném az ***Állapot*** nevű fület, mely a ***Grip*** által használt konzolparancsok kimenetét mutatja folyamatosan. Ezt tanulmányozva talán jobban megismerhetjük ezt a nagyszerű kis összetett programot.

Ne felejtsük el, hogy ezt az alkalmazást nem kalózkodásra találták ki. Csak arra használjuk, amire feltétlenül szükséges, és vigyázzunk, hogy ezzel másoknak kárt ne okozzunk.

Apagyi György, (killall)
(killall@root.hu)

25 éves, jelenleg az ELTE programozó matematikus szakán másodéves hallgató. Hobbija a zene (gitározás), az olvasás (Stephen King) és a számítástechnika (Linux, Unix, VMS).

A Blizzard játékai Linux alatt

A címben szereplő csapatot talán nem kell különösebben bemutatnom a játékrovat Olvasóinak: elég megemlítenem a Warcraft sorozatot, a Diablo két részét vagy a Starcraft világát.

A friss játékrovat olvasói kérésre készült: ezen a téren nem tartottam szükségesnek átlépni a *Win32 wrapperek* világába, de az utóbbi időben elég sok levelet kaptam ebben az ügyben – így nekiültem kivesézni a témát. A kérésekhez igazodva megpróbálok aprólékos leírással szolgálni a technikai részleteket illetően. Elöljáróban annyit, hogy a használandó kompatibilitási réteget a szabadon elérhető *WINE wrapperben* kerestem, a játékprogramok közül pedig hármat választottam ki: név szerint a *Starcraft*-ot, a *Diablo2*-t, és a *Warcraft3*-at. A három műremek minden vérbeli játékos számára közismert (és nagyra becsült) szoftver, ezért a bemutatásukat a legrövidebb terjedelemre fogom szorítani – sokkal inkább az üzembe helyezés és a hibalehetőségek ismertetése érdemel több figyelmet.

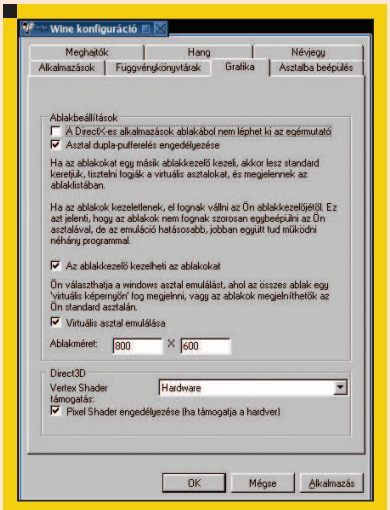
Miért a Blizzard, miért a WINE?

A cég története számtalan helyen fellelhető az éterben, ezzel nem untatnék senkit. Sokkal érdekesebb információ az, miszerint a *Blizzard* fejlesztői maguk is „játékőrültek". Nem csoda hát, hogy az összes projektjük bombasztikus sikerű. A kiszemelt három játékszoftver a *PC*-s játékipar megfelelő kategóriájának éllovasa: természetesen akad konkurens program, mely több téren is aktuálisabb náluk (törvényszerűen, hiszen például a *Starcraft* lassan már a tizedik évét tapossa), ennek ellenére a *Blizzard* remekei összességében utolérhetetlenek. Sajnos azonban a csapat mereven elzárkózik a *Linux* rendszerek támogatásától. A linuxos futtatás ezért „kerülőúton", a *Win32* környezet imitálásával valósulhat meg. Erre a célra szolgál a bevezetőben említett, nagy múltú *wrapper.* Valószínűleg kevesekneknek idegen a téma, hiszen párszor már e magazin is cikkezett a szabadon elérhető *WINE* környezetről. Dióhéjban arról van szó, hogy a projekt egy utánzott *Windows* felületet, legfőképpen ennek *DirectX* rétegét próbálja meg a kiszemelt (játék)szoftver alá „erőltetni", több-kevesebb sikerrel. Az érdeklődők a *Linuxvilág 2006/09* számában (*Apagyi György: Egy korty Windows*) ismét naprakész információkhoz juthattak, így az üzembe állítás ismertetését mellőzném. Ha valaki lemaradt volna erről a lapszámról, azért ne essen kétségbe: a fő *Linux* terjesztések magukban hordozzák a projektet, ahol pedig mégsem lenne így, ott a rendszer csomagkezelőjével pillanatok alatt segíteni lehet a hiányosságon. Végső esetben a forráskód lefordítása is megoldás lehet, de mint mondtam, a naprakész és magára valamit is adó terjesztések általában megkímélnek a hosszadalmas pluszmunkától.

WINE plusz

A favorit *wrapper* gyorsan fejlődik: nagyjából kéthetente jelenik meg frissebb kiadása, mely kifinomultabb (utánzott) környezetben, újabb képességek implementálásában, valamint hibajavításokban tér el az előző verziótól. A folyamatos változásoknál azonban sokkal szembetűnőbb az az újítás, miszerint a program futását befolyásoló bejegyzések nemrégen átkerültek a *fake* regisztrációs adatbázisba az e célra fenntartott speciális állományból. Emiatt a régebbi *WINE* változatokat máshogyan kell beállítani, mint az újabb jelölésűeket (régebbi kiadásokkal rendszerint előző generációs *Linux* terjesztésekben találkozhat a felhasználó – illetve akkor, ha ezen a téren szándékosan az öregebb megoldásokat keresi). Fontos dolgok ezek: a folyamatos „piszkálásra" állandóan szükség van, hiszen minden windowsos program más beállításokat kíván – mint ahogy a három *Blizzard* játék is különböző *WINE* környezetet igényel: ez a jelenség a játékmotor és a megjelenés időpontjában aktuális *Windows* verziók eltérése miatt törvényszerű.
A *WINE* verziószámát konzolon kiadott

```
wine --version
```



1. ábra A WINE „hangoló eszköze"

2. ábra A Starcraft telepítése, Linuxon

3. ábra A Starcraft KDE asztalon fut

paranccsal kérdezheti le a felhasználó. Az egyszerű `wine` parancs építi fel a *wrapper* virtuális környezetét: először összeállítja a használható fontkészletét, majd generálja a regisztrációs adatbázisát és az imitált mappaszerkezetét a */home/$/.wine* úton. Az újabb kiadásokhoz tartozó `winecfg` parancs egy grafikus interfészt varázsol a képernyőre, ahol a lehetőségekhez mérten sok minden beállítható – kezdve az utánozandó környezettől egészen a grafikus kártyánk árnyaló egységeinek kezeléséig.

A beállítás eredményét a *fake Windows* regisztrációs adatbázisában rögzíti, sok más (e tekintetben jelentéktelen) információ mellett. A régebbi verziók szintén rendelkeznek grafikus felületű hangolóval, ellenben a beállítások ott egy tagolt (kategóriánként deklarált), és némi rálátással könnyen kezelhető fájlban tárolódnak. Ez a fájl a személyes *.wine* könyvtár gyökerében található *config* állomány, melynek kézi szerkesztése jóval hatékonyabb (volt) az előbb említett „kényelmes” megoldásnál. Mivel azok a felhasználók, akik egy adott fejlettségű projektnél kitartanak, azok valószínűleg ismerik annak csínját-bínját, így most az újabb keletű *WINE* változatok célirányos használatát fogom előtérbe helyezni (a kezdő felhasználók igényei szerint).

Starcraft-tól kezdve

A *Starcraft* egy valós idejű stratégiai játékszoftver. Izometrikus nézetű *2D* világa elkalauzol bennünket a jövő egyik háborújába, ahol bármely fél (*Emberek*, *Zergek*, *Protossok*) oldalán beszállhatunk a küzdelmekbe. Aprólékos kidolgozottság, remek zenei kíséret, korszakalkotó ötletek jellemzik a kódot, mely mindhárom fajhoz különböző játékstílust kapcsol. Megjelenésekor szerte a világon napok alatt milliós nagyságrendben értékesítettek a népszerű, színvonalas és örökzöld projektből. A következő technikai jellegű információk kiemelten fontosan számunkra: a játék *Linuxon* a *fake Windows98* környezetben érzi igazán jól magát, a futását segítő *WINE* ebben az esetben jellemzően ki nem állhatja a *DGA* kiterjesztést. Hangszolgáltatása is kényes, leginkább *OSS* kapcson keresztül szeretne „megszólalni”. A játék sok gépen szaggatottan fut teljes képernyőn, továbbá allergiás a telepítő *CD* hiányára – már a lemez címkéjének eltérésére megtagadja a futást (*supermount* mechanizmussal befűzött korongok esetén nem ritka jelenség). Nos, ezeket a sajátosságokat figyelembe véve hívjuk életre a „sötét jövőt” rendszerünkön! Fűzzük be egy csatolási pontra a *Starcraft* lemezét, majd konzolon adjuk ki a `winecfg` parancsot! Előbb az *„Alkalmazások”* fülön válasszuk ki a *Windows98* imitálását, majd a *„Grafika”* szekcióban engedélyezzük az asztal dupla pufferelését. A felbontást fixen *640x480* ablakos értékre kell rögzíteni úgy, hogy az egérmutató lehúzható legyen a windowsos alkalmazásról (a *Shaderek* kezelése ebben az esetben lényegtelen, mivel a kiszemelt szoftver *2D* leképezést használ). A *„Hang”* lehetőségei közül válasszuk ki az *OSS* kaput (ehhez a *Linuxnak* rendelkeznie kell *OSS* képességgel, de legalább egy működő *ALSA/OSS* átjáróval), majd a gyorsítást állítsuk emulált módra. Végezetül a *„Meghajtók”*

fülön adjuk meg az optikai meghajtót: a *„Hozzáad”* gombra kattintva feltűnik a következő használható betűjel, aminek az elérési útja szerkeszthető. A *„Haladó”* lehetőséget kell választanunk, hogy tudassuk a *WINE* környezettel a játék lemezének címkéjét. Most már csak annyi munkánk maradt, hogy a hangolót az *„Alkalmazás, Ok”* gombbal bezárjuk. Lépjünk a befűzött *CD* gyökerébe, és adjuk ki a `wine setup.exe` parancsot! A telepítés befejeztével a játék az elérési útjára állva indítható, a szokásos szintaktikával: `wine starcraft.exe`. Az eredmény nem marad el, hiszen a *Starcraft Linuxon* nem különböztethető meg meg a Windows rendszereken mutatottaktól.

4. ábra A Diablo2 is használható

5. ábra 800x600 felbontású virtuális világ

A Diablo2-n át

A *Diablo2* nem kevésbé sikeres, mint az előző játék. Egy szintén izometrikus felépítésű programról van szó, mely egyedi módon ötvözi az akció és szerepjáték vonalat, minimális stratégiai beütéssel. Az első rész történeti szálát tovább „húzva” öt (küldetéslemezzel hét) kaszt bőrében küzdhetünk az ellenséggel, négy (illetve öt) virtuális világban. A fejlődési rendszer, a kiváló ötletek, a komoly kihívás és a hálózati opció sikere minden idők egyik legnagyobb kasszasikerévé emelte ezt a műremeket (a *Diablo* sorozat hozzám is különösen közel áll, hiszen a két epizód előtt ülve sok ezer órát töltöttem el egyedül és csapatban egyaránt). Linuxos futtatásáról érdemes azt tudni, hogy a *fake Windows2000* környezetben működik a legstabilabban. Képi világát szereti fix felbontású *WINE* ablakban felépíteni (ami felbontás függ a játék verziójától is). A kiegészítő lemez installálása után az első négy világ a merevlemezre került adatokból generálódik, azonban az utolsó, *Harrogath*-al kezdődő pályasor már a *CD*-meghajtóról töltődik (és rögtön megmutatkozik az optikai tároló winchesterekhez viszonyított szerény elérési ideje). A telepítés igazi kihívás: az alapverzió három lemezből áll, erre jöhet a kiegészítő projekt diszkje (ami által a játék új távlatokat nyit a kalandorok előtt), majd erre a különböző *patch*-ek, esetleg a magyarított feliratok és a hazai szinkron. A nehézséget a használandó médiumok leválasztása és csatolása jelenti, miközben a *WINE* futtatja az azokon lévő *Win32* binárisokat (vagy egyéb állományokat használ). Ha valaki szeretné a körülményes telepítést *Windows* nélkül kivitelezni, annak elektronikus levélben szívesen segítek – most azonban zárjuk le gyorsan ezt a pontot: telepítsük fel a játékot egy *Win32* hajtotta gépre, majd teljes egészében húzzuk át a */home/$/.wine/* útra. Ezek után adjuk ki a `winecfg` parancsot! A *wrapper* működését állítsuk *Windows2000* imitálására. Ezután engedélyezzük az asztal dupla pufferelését, és állítsuk a virtuális asztalt fixen 800x600 felbontásúra. A hangszolgáltatást lehetőleg *OSS* meghajtóra irányítsuk, a optikai tárak kezelésénél pedig hasonlóan járjunk el, mint a *Starcraft* esetén (előtte azonban fűzzük be a játék kívánt lemezét: *Diablo2* esetén a *„Play* „diszket, *Diablo2LOD* esetén az *„Expansion”* lemezt). Alkalmazzuk a beállításokat, majd a *Linuxra* költöztetett szoftver mappájába állva adjuk ki a `wine game.exe` parancsot: nos, így indítható a kaland...

6. ábra A Warcraft3-at sem kell nélkülözni Linuxon

A kiegészítő projekt ügyén említettem, hogy a *Harrogath* világ *CD/DVD* drájvról fut, aminek érezhetően szerény a hozzáférési ideje. Segítsünk a helyzetünkön a következők szerint: hozzunk létre egy mappát valamely írható területen, majd másoljuk ide az *„Expansion”* média tartalmát. A *WINE* hangoló ablakában az optikai tár elérési útjaként adjuk meg ezt a mappát, majd a *„Haladó”* lehetőséget használva gépeljük be a lemez címkéjét. Így a *Diablo2LOD* már nem kér semmilyen *CD*-t futás idejére, mivel a keresett adatok a háttértárról lesznek felhasználva (ezt a trükköt azonban a *wrapper* nem minden verziója támogatja, mint ahogyan a játék is kényes lehet erre – a használt *patchek* verziószámától függően).

A Warcraft3-ig

A *Warcraft3* valós idejű, *3D* stratégiai program. „Mesebeli” világával, kiforrott ötleteivel, több játszható kasztjával érte el, hogy szintén milliók hódolnak neki: az előző két részben feldolgozott *Orkok* és *Emberek* harcát kiegészítették néhány idegen fajjal. Kasztjainak eltérő harcmodorával, látványos grafikájával és varázslataival sajátos atmoszférát teremtett maga köré, ráadásul ennek a hangulatnak a gyökerei egészen az előző két rész hírnevéig is visszanyúlnak. *Linuxon* a játék leginkább imitált *WindowsXP* környezetben szeret futni: grafikája és hangszolgáltatása egyaránt összetett – nem egyszerű feladat *WINE* által megfelelő alapot biztosítani számára.

A grafikai leképezés itt már *3D*-s, a hangleképezés *Surround/3D* képességű (*EAX*). A használandó *fake* konfiguráció a következő: a *wrapper* működése az említett operációs rendszerre állítandó. A képi világot illetően az imitált környezet *Shader* támogatását itt ugyan nem érdemes piszkálni, ennek ellenére állítsuk *Emulated* állásba. A programot semmiképpen ne engedjük teljes képernyőre, tehát egy fix felbontású asztalra feszítsük rá, például *800x600* méretben. A hang leképezését pedig (ha lehetséges) az *OSS*-nél többre képes *ALSA* megoldásra kell rábíznunk, hardveres gyorsítást kérve.

A telepítő diszket befűzve (és az optikai egység elérését az előző példák szerint beállítva), gyökerébe állva indítsuk el az installálást (`wine setup.exe`). Miután minden állomány a helyére került, a játék elérési útján adjuk ki a `wine jwar3.exe` parancsot: az eredmény a mellékelt képeken látható... Magától értetődik, hogy a *Warcraft3* futtatásához működő *GLX* illetve *DRI* kapoccsal ellátott *3D* grafikus hardver szükséges!

Konklúziók és kiegészítések

A három játék hardverigénye mára szerénynek mondható, hiszen a *Starcraft* kompromisszumoktól mentes futtatásához egy *166 MHz* órajelű *x86* processzor is megteszi, de az erősebb gépet kívánó *Warcraft3* is beéri *1 GHz*-es *CPU*-val, (valamint egy harmadik generációs, mai szemmel egyszerű *3D* grafikus kártyával). *WINE* környezetben használva a központi egységre vonatkozó igényeik nem nőnek számottevően, a szükséges memória mennyisége viszont jellemzően a másfélszerese, kétszerese (sőt, néhányszor akár triplája) is lehet a *Windows* környezetben mért, megfelelő értékeknek.

Így a *Starcraft*-hoz *128 MByte*, a *Diablo2*-höz *256 MByte*, a *Warcraft3*-hoz pedig már *384-512 MByte* központi tár szükségeltetik. Természetesen ezek alatt is használható mindegyik játékszoftver, de soha ne feledjük: a program „mellett” fut egy *wrapper* alkalmazás, ez alatt pedig egy komplett *Linux* rendszer, működő *X szerverrel* – aminek a memóriaigénye extrém esetben már önmagában is elérheti a *0,5 Gbyte* értéket! A leírt információk minimális mértékben rendszerenként eltérhetnek, ebből adódóan előfordulhat (bár nem valószínű), hogy nem hoz 100%-os eredményt a *WINE* beállítása valamely játéknál.

Remélem azért a kezdő felhasználók hasznos támpontként fogják átlapozni e néhány oldalt, mivel ezeknek a játékoknak a környezetét eddig nagyjából húsz-huszonöt idegen gépen, különböző *Linuxokon* is ilyen „elvek” alapján állítottam be. Végezetül: talán feltűnik néhány Olvasónak, mennyire kerülöm a teljes képernyős futtatást. Meggyőződésből használom ezeket a szoftvereket mindig ablakban, mivel az ilyenkor le is húzhatom róluk az egérmutatót, ráadásul a játék esetleges összeomlása ilyen feltétel esetén nem rántja magával az *X* kiszolgálót. Tartalmas kikapcsolódást!

Kovács Zsolt (kovi@linuxforum.hu)

Quake fanatikus. Négy éve a debreceni linuxosok egyike. Töretlenül hisz a Slackware terjesztésben.