Beköszöntő

Linuxvilág

Bátran állíthatom, hogy az októberi lapszám a ***Linuxvilág*** életében egyfajta mérföldkő.
Mint azt olvasóink közül nyilván már sokan tudják, néhány hónapja azzal a kéréssel fordultunk a magyar linuxos közösséghez (➲ ***www.linuxvilag.hu/szerzoknek***), hogy segítsenek nekünk megújítani a lap arculatát azzal, hogy cikkek formájában átadják megszerzett tudásuk egy-egy darabkáját a többieknek.
A változás tulajdonképpen már a szeptemberi számon is világosan látszott, hiszen számos új, a leghatározottabban tehetséges szerző bukkant fel benne. Az a lapszám, amit az olvasó most a kezében tart attól olyan különleges, hogy ezt – a kernelhírektől eltekintve – már ***kizárólag*** magyarok írták.

Amikor nyár elején szervezni kezdtem ezt az akciót, a lelkem mélyén tulajdonképpen éreztem, hogy ennek a dolognak működnie kell. Biztos persze nem lehettem benne, már csak azért sem, mert a nyár valahogy nem a „kirobbanó aktivitás" időszaka. Ha júniusban valaki azt mondja nekem, hogy októberre nemhogy egy teljes lapszámot „összehoznak" a magyar szerzők, hanem számos cikket várakoztatni kell, mert egyszerűen nem fér bele a 82 oldalba, valószínűleg jót derültem volna rajta.
Az egész stáb nevében köszönöm nektek!
Folytassátok!

Büki András

Most pedig lássuk, mit olvashatunk októberben.
Az aktuális kernelhírek után megtudhatunk néhány pikáns részletet kiválasztott pingvinünk, Tóbiás szerelmi életéről. (Csak 18 éven felülieknek!) Aztán hogy ***Szy György*** főszerkesztőnk kérésének – miszerint minél nagyobb számban jelenjenek meg lányok a lapban – is eleget tegyünk, a ***Magyar Linux Dokumentációs Projekt*** kapcsán a nők és a ***Linux*** kapcsolatáról lesz szó.
A ***„Fejlesztői sarok"*** rovatban két régi szerzőnk, ***Fülöp Balázs*** és ***Komáromi Zoltán*** folytatja a még az elmúlt korszakban megkezdett sorozatát (***Java*** és ***PEAR***).
Aztán egy körkép következik a legnépszerűbb ***Linux*** terjesztések közül néhányról. A tervek szerint ezt a témát a következő lapszámokban még újra elővesszük.

A jelenlegi menü: ***System Rescue CD***, ***Ubuntu***, ***Slackware***, ***Frugalware***, ***blackPanther OS***.
Medve Zoltán a szerverszobák hőmérsékletének mérésére mutat be egy kissé talán rendhagyó, de mindenképpen olcsó megoldást, ***Sütő János*** pedig a spamszűrők működésének alapjait tekinti át.
A ***„Grafika, multimédia"*** rovatban folytatódik a ***Blender*** (***Szalai András***), a ***PoVRay*** (***Auth Gábor***) és az ***SVG*** (***Novák Áron***) sorozat, illetve megtudhatjuk, hogyan készíthetünk „óriáspókos-sikongatós" horrorfilmet ***Blenderrel*** (***Szabó Péter***).
Végezetül egy vállalati levelezőrendszer kialakításába kaphatunk betekintést ***Pallos B. Richárd*** segítségével.
Kellemes olvasást kívánunk!

Hírek

Átkeresztelt Mambo

A népszerű, korábban ***Mambo*** név alatt futó tartalomkezelő rendszer a továbbiakban ***Joomla!*** név alatt lesz elérhető – jelentették be a tervezet vezetői. A fejlesztők már életre hívták a tervezet új webhelyét is, a ***Joomla! 1.0*** kiadását pedig szeptemberre ígérik. Az első kiadás természetesen nem csak annyiban fog eltérni az utolsó ***Mambotól***, hogy átírják a nevét, hanem számos hibajavítást is tartalmaz majd, illetve a webhelyen található úti terv szerint a jövőben is lendületesen folytatódik az újabb szolgáltatások megvalósítása.

➲ *www.joomla.org*

Hamarosan összecsavarhatjuk az újságot

A ***Philips*** egyik leányvállalata, a ***Polymer Vision NV*** a németországi ***IFA*** kiállításon összetekerhető elektronikus kijelző bemutatását tervezi. A ***Readius*** nevű készüléket hordozható ***e-readernek***, elektronikus újságfélének szánják, létének célja egyelőre a technológia működésének demonstrálása, tényleges termékként nem fog megjelenni. A ***Readius*** a ***PV-QML5*** típusjelzésű megjelenítőre épül. A 100 mikron vastagságú eszköz ***320x240*** képpontos felbontást tud, átmérője 5", kontrasztaránya 10:1. Ha használója elolvasta az őt érdeklő tartalmat, akkor a kijelzőt egy 100 x 60 x 20 mm méretű tokba tudja összecsavarni.

Netrádió

A felső kategóriás hangszórókat gyártó brit ***Acoustic Energy*** ***„Wi-Fi internet radio"*** névvel internetes rádiókészüléket mutatott be. A rádiókészülék rádásul vezeték nélküli is – noha ez a hagyományos rádióknál magától értetődő, itt arról van szó, hogy az internetkapcsolatot is vezeték nélküli csatolóval hozza létre. A ***Wi-Fi*** rádió a ***Real Audio***, a ***WMV*** és az ***MP3*** adatfolyamokat támogatja, így elvileg az interneten elérhető rádióadások 99 százalékát képes fogni. Az adók adatait a ***Reciva*** cég kiszolgálójáról tölti le, mely több ezer adót ismer a világ minden tájáról, de a tulajdonos saját számítógépéről származó tartalmak otthoni sugárzására is alkalmas. A rádióvevő ***Linux*** alapú, ennek köszönhető viszonylagos rugalmassága, amely a formátumok és a protokollok kezelése és a hálózati kapcsolattal összefüggő biztonsági szolgáltatások terén egyaránt megmutatkozik. A várhatóan novemberben megjelenő rádió ára kissé borsos, körülbelül 70000 forint lesz.

Technológiák, egyesüljetek

DECT rendszerű, ***VoIP*** hívások kezelésére is képes telefonkészüléket mutat be hamarosan a francia ***Thomson***. Az ***EV5203-C*** jelzésű telefon előnye, hogy használatával számítógép közreműködése és ***SkyPE*** vagy egyéb alkalmazás futtatása nélkül is kényelmesen lehet ***VoIP*** hívásokat bonyolítani. A színes kijelzővel ellátott készülék egyszerre két hívást is képes kezelni, két ***Ethernet*** aljzatával pedig közvetlenül ***DSL***- vagy kábelmodemhez, illetve forgalomirányítóhoz lehet csatlakoztatni. A ***Thomson*** a nagyközönség számára egyelőre nem kívánja elérhetővé tenni az új típust, megvásárlása kizárólag a szolgáltatók számára lesz lehetséges – ők viszont számos lehetőséget biztosíthatnak majd általa, ilyen például a telefonkönyv tartalmának szinkronizálása a hálózattal, az értesítések küldése a levelekről, a webböngészés, az ***SMS*** és ***MMS*** küldés, valamint a készülék belső programjának szolgáltató által végzett, távoli frissítése. A készülék operációs rendszere ***Linux*** alapú, és a szolgáltatók számára különféle testreszabási lehetőségeket is biztosít – éppen ezért talán nem is fogjuk ***Thomson*** névvel látni. Fontos megjegyezni, hogy a ***Thomson*** egységét hagyományos telefonhálózatra is lehet csatlakoztatni, aminek elsősorban a segélykérő hívások lebonyolításakor lehet jelentősége, hiszen a ***VoIP*** szolgáltatók ilyen számokat nem feltétlenül üzemeltetnek.

mobil iTunes

A ***Motorola***, a ***Cingular Wireless*** mobilszolgáltató és az ***Apple Computer*** együttműködésének eredményeként elindult az első olyan szolgáltatás, amely mobiltelefonról teszi lehetővé zenei tartalmak vásárlását az ***iTunes*** boltból. A ***Motorola*** a ***MotoROKR*** nevű telefonkészülékkel szállt be az üzletbe, mely jelenleg az egyetlen ***iTunes***-képes mobilkészülék, értelemszerűen a ***Cingular*** adja a telefonhálózatot, az ***Apple*** pedig magát az online boltot – természetesen az új szolgáltatástól mindhárom szereplő forgalomnövekedést vár.

ThinkFree Office

A ***ThinkFree*** bemutatta ***Office*** nevű programcsomagjának legújabb, 3-as számot viselő változatát. A ***Microsoft Office*** formátumaival teljes kompatibilitást ígérő csomag három alkalmazást tartalmaz, a ***Write*** nevű szövegszerkesztőt, a ***Calc*** nevű táblázatkezelőt és a ***Show*** nevű bemutatókészítőt. A ***ThinkFree Office Java*** alapú, ennek köszönhetően nem okozott gondot, hogy azonnal háromféle operációs rendszerre is elkészítsék: ***Linuxra***, ***Windowsra*** és ***Mac OS X*** alá. A csomag sokféle eszközzel segíti az operációs rendszerek közötti átjárást, ilyen például az ***XML*** és a ***PDF*** formátum támogatása és az, hogy beépítve tartalmazza a leggyakrabban használt betűtípusokat, az ***Arialt*** és a ***Times New Romant***. A ***Java*** nyelv révén a csomag vállalati környezetben nem csak az egyes munkaállomásokra telepíthető fel, de központi kiszolgálóról is elérhetővé tehető, így csökkenthetők a felügyeleti terhek, illetve kisebb gondot okoz egy-egy munkaállomás meghibásodása. A cég online változatot is kínál belőle, ez a dobozos változat minden elemét tartalmazza, továbbá minden felhasználónak 30 MB tárhelyet biztosít a dokumentumainak tárolására. Az 50 dolláros csomag jelenleg 11 nyelven beszél, a közeljövőben várhatóan további nyelvekre is honosítani fogják.
➲ ***www.thinkfree.com***

Infráék erősítenek

Az ***Infrared Data Association (IrDA)*** új, a korábbinál jóval nagyobb sebességet nyújtó infravörös adatátviteli protokollt fogadott el. Az ***IrSimple*** nevű protokoll idővel akár 100 Mbit/s sebességet is képes lesz biztosítani, ami 25-szörös növekedést jelentene a jelenlegi 4 Mbit/s-hoz képest. Az új infravörös szabványt elődeihez hasonlóan főként hordozható készülékekhez, digitális fényképezőgépekhez fejlesztik, ugyanakkor életképessége finoman szólva kétségbe vonható, hiszen a rádiós megoldások, mint a ***Bluetooth*** és a ***Wi-Fi***, az utóbbi időben erősen háttérbe szorították az infravörös megoldásokat. Az ***IrSimple*** terjedését elősegítheti, hogy a megfelelő szoftverfrissítés végrehajtása után a meglévő adó-vevőket is képes lesz 16 Mbit/s sebességűre gyorsítani, ugyanakkor ellene szól, hogy használata a rádiós összeköttetésekhez viszonyítva továbbra is kényelmetlen marad.
➲ ***www.irda.org***

Okos ellenségek

Az izraeli ***Alseek*** az ***Ageia*** a fizikai modellezést segítő ***PhysX*** lapkájának mintájára újabb kiegészítőt tervez a számítógépes játékok számára: mesterséges intelligenciát növelő célprocesszort.
A számítógépes játékok régi problémája, hogy a gépi játékosok túlságosan ostobák, sokszor céltalanul, rossz irányba, ide-oda rohangálnak, a különféle akadályok és tereptárgyak között nem találják a célhelyhez vezető útvonalat, esetleg éppen egymásban akadnak el. A játékfejlesztő cégek komoly erőfeszítéseket tettek a háttérben futó algoritmusok javítására, ám az eredmény mind a mai napig megkérdőjelezhető. Ennek oka részben a számítógépek számítási kapacitásának korlátossága, illetve az, hogy a mesterséges intelligencia biztosítása rengeteg igen/nem jellegű döntést igényel, míg a számítógépek processzorai inkább lineáris jellegű feladatok elvégzésére alkalmasak. Az ***Alseek AIS-1*** lapkája, melyből már el is készült az első prototípus, a tervek szerint ezen fog segíteni. A feladat természetesen sokrétű, hiszen az építkezős-háborúzós játékok egészen másfajta mesterséges intelligenciát igényelnek, mint a lövöldözősek vagy éppen a sakkjátékok. A lapka négy fő területen lesz erős: az útvonalkeresésben, a terepelemzésben, a gépi harcosok vagy kisebb csoportok mozgatásában és az érzékelőszervek szimulálásában; utóbbi révén lehetővé válik, hogy a gépi szereplők élethű módon reagáljanak a kívülről érkező hatásokra. Az ***AIS-1***-et a játékosok számára ***PCI*** foglalatba illeszkedő kiegészítő kártyán kívánják elérhetővé tenni, illetve a gyártók szándékaitól függően akár a játékkonzolokba beépítve is viszontláthatjuk majd – egyelőre még bizonytalan, hogy mikor.

Medgyesi Zoltán
(mz@rettesoft.hu)
A Linuxvilág hírszerkesztője. Szabadidejét legszívesebben a barátnőjével tölti, szeret autózni és bográcsban főzni.

Mi újság a rendszermag fejlesztése körül

A *git* jelenség folytatódik. A saját fejlesztését kiszolgáló, kezdetektől villámgyors *git*, és a *Cogito* nevű közkedvelt csomagoló (wrapper) szkriptjei folyamatosan javulnak, és mára már hatalmas lökést adtak a kernel fejlesztésének. Különböző projektek állnak át *BitKeeper*-ről *git*-re. A *Net driver* és a *libata* fejlesztése máris átállt. Ugyanígy tett a *JFS* és *NTFS* projekt is. A stabil *w.x.z.y* kernelfa, melyet *Greg Kroah-Hartman* és *Chris Wright* tartanak karban, szintén nem régen cserélte le a *BitKeeper*-t, *git*-re. Néhány kernelhacker számára a *git* előrelépést jelent a *BitKeeperhez* képest, és sokkal több munkát tudnak vele elvégezni, mint azelőtt. Még maga *Linus Torvalds* is újragondolja a foltok kezelésének módját abból a célból, hogy azt a *git*-hez hozzáillessze.

A *BitKeeper* dokumentációt eltávolították a kernel forrásából és a levelezési listák, mint amilyen a *bk-commits-head*, amelyek eredetileg azt a célt szolgálták, hogy a *BitKeeper* változásait tegyék rajtuk közzé, ma a *git* kernelfolt megjegyzéseket fogadják.

Ugyanakkor a *git* nem való mindenkinek. Amikor *Andrew Mortont* megkérdeztük, azt válaszolta, hogy nem áll szándékában a *git*-et saját *-mm* kernel fájának menedzselésére használni, mert saját patch szkriptjei teljesen kielégítik igényeit. *Matt Mackwall* szintén saját villámsebességű verzió követő rendszerével, a *Mercuriallal* dolgozik. A *Mercurial* szintén egy kiváló eszköz, és sok szempontból egyenértékű a *git*-tel, kiváltképpen a sebesség területén. Valójában, ahogyan *Linus* rámutatott, mindkettő az alapvető viselkedésük tekintetében azonos. Valóban, mindkettő a verzió követés egy teljesen új aspektusát képviseli, és a projektek ahogyan az *arch* melyek korábban vezetők voltak ezen a téren, most hirtelen küzdenek azért, hogy felzárkózzanak.

Markus Klotzbuecherhas készített egy érdekes új virtuális fájl rendszert, amit *mini_fo*-nak (*fanout overlay*) neveznek. Mely engedélyezi a felhasználóknak, hogy csak olvasható fájlrendszeren található fájlokba írjanak, azáltal, hogy létrehoz egy írható területet valahol máshol és a felhasználó változásait egy, a csak olvasható terület fölötti rétegben helyezi el. A felhasználó számára mindezt átlátszó módon. Ami eddig csak olvasható adat volt, most írhatónak is tűnik, annak ellenére, hogy a csak olvasható adat sohasem változik. A *mini_fo* eszközt arra a célja, hogy elérhetővé váljon a szoftver frissítés beágyazott rendszerekben, de egyéb felhasználási területet is találtak a szoftvernek, és még kétségtelenül lesz néhány.

Alan Cox és *Bartlomiej Zolnierkiewicz*, a két remek *IDE* fejlesztő, bajba kerültek amikor egységesíteni próbálták igyekezeteiket. Mialatt *Bartlomiej* a jelenlegi *IDE* karbantartó, *Alan* sok előkészületet tett azért, hogy a régi *IDE* kódot karbantartható állapotba hozza a kimondhatatlan (hulla-szór) rémálomból, amilyen már évek óta volt. Habár *Alan* az utóbbi időben némiképp a kívülről figyelte a *Linux* fejlődését, visszatért, hogy ellenőrizze az *IDE* haladását, és nem tetszését fejezte ki azokkal a változásokkal szemben, amit *Bartlomiej* tett. Úgy tűnik, mintha harag lenne kettejük között, ahogyan ez szokássá vált bármely két fejlesztő között, akik egyszerre, vagy egymás után dolgoztak az *IDE* fejlesztésén. *Bartlomiej* felajánlotta *Alan*-nek, hogy ágaztassák el a kódot és készítsék külön a fejlesztést, amennyiben ez őt boldoggá teszi.

Azt mondják az *IDE* fejlesztése még mindig ellenségeskedést szülhet a fejlesztők között. Ezért nekünk az *IDE* lemez iparágat magát kell vádolni, amelyik teljesen egyoldalú és megsemmisített minden lehetséges szabványt, kivételt halmozva kivételre, mindezt elegyítettek kereskedelmi titkokkal, és szabadalmaztatott dokumentumokkal, amitől bárki megőrülhetne, még akkor is, ha megkísérli karbantartani az *IDE* kernel kódot. Ez a család, mint *Bartlomiej* és *Alan*, és azok az emberek, akik előttük jöttek, mint *Mark Lord* és *Andre Hedrick*, éppen ezt csinálják hálával tartozunk ezért nagylelkű természetükért. Az *IDE* kód nélkül, legtöbbünk közel sem találná a *Linuxot* ilyen használhatónak.

Benjamin LaHaiser nemrégiben megpróbálta egyszerűsíteni és jobban karbantarthatóvá tenni a szemaforra alapozott kölcsönös kizárást a teljes kernel architektúrában. A jelenlegi kód komplex, nehezen olvasható, és sok architektúra specifikus részletet tartalmaz. Ezek a kis árnyalatnyi különbségek a támogatott architektúrák számával arányosan nőnek, és érthető az erőfeszítés egy generikus szemafor rendszerre, amely lefordítható, és működtethető egységesen az összes architektúrán. Azonban a szemaforok alacsony szintű műveletek, és a nagy sebességű kód szükségessége vitathatatlan. Pontosan ezért kisebb lassulás, elképzelhető, hogy jelentős mértékű kernel lassulást eredményezne, ezért feltehetően bármelyik kísérlet a kód egységesítésben erős ellenállással találkozna a különböző architektúrát karbantartók részéről. Ez volt az, ahogyan valójában *Benjamin* munkáját fogadták. Habár néhány javítást természetesen készítettek, nem tűnik valószínűnek tehát, hogy a szemafor kód valamikor is igazán általános és egyszerű lesz. A sebesség még mindig elég erős ösztönzőerő.

Zack Brown

Linux Journal 2005. 137. szám

Novell Security Manager – Védelem egy egész hadsereg erejével

A Novell Security Manager egy kényelmesen felügyelhető, integrált csomag a vállalatok és intézmények biztonságos működésének megteremtéséhez.

Informatikai rendszerünk stabil működésének elengedhetetlen feltétele a biztonság megteremtése. A külső fenyegetésekkel szembeni védelem megvalósításához azonban sokféle biztonsági eszközre és alkalmazásra van szükség egy vállalatnál: tűzfalra, virtuális magánhálózatra, behatolásvédelemre, vírus- és spamszűrésre, valamint ***URL***-szűrésre. Mindezen védelmi módszerek különféle gyártóktól való beszerzése és telepítése költségigényes, és a rendszer felügyeletét is jelentősen megnehezíti. A ***Novell Security Manager*** mindezeket egyben, egy kényelmesen felügyelhető, integrált csomagban kínálja.

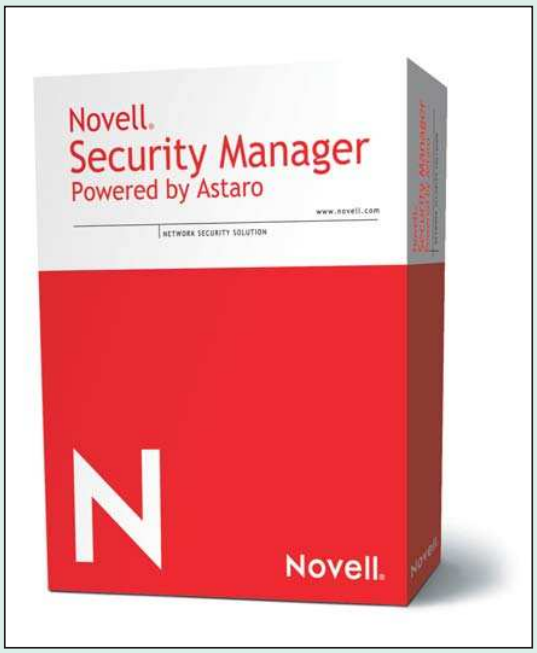
A ***Novell*** a ***Linux*** és a személyazonosság alapú biztonság területén piacvezető technológiáját és az ***Astaro*** hálózati biztonsági szoftverek és berendezések terén szerzett tapasztalatát ötvözve fejlesztette ki a ***Novell Security Manager*** legújabb verzióját. A ***Linux*** alapokra épülő biztonsági megoldás kihasználja a nyílt forráskódú közösség együttműködésének eredményeit; egyetlen kereskedelmi szoftver sem kínálja ugyanezeket a funkciókat.

A ***Novell Security Manager*** nagy teljesítménye miatt hatékony védelmet nyújt az első vonalban felmerülő biztonsági fenyegetésekkel szemben, de használható egy meglévő tűzfal mögötti kiegészítő szintként is.

Áttekintés

A folyamatos, zökkenőmentes kommunikáció az ügyfelek, az üzleti partnerek és az alkalmazottak között létfontosságú egy vállalat működéséhez. Az internetet használó szervezetek fokozottan ki vannak téve a különféle rosszindulatú támadásoknak: vírusok, férgek, alkalmazások kihasználása, ***szolgáltatásmegtagadási támadások (denial-of-service; DoS)***, spam, információlopás és még sorolhatnánk.

Az egyre több irányból fenyegető veszélyek ellen többféle védekezési módszert is kell használni – a rendelkezésre álló költségkereten belül. A ***Novell Security Manager*** minden más megoldásnál alacsonyabb költségszinten képes mindezt megvalósítani. Mivel ***Linuxra*** épül, nem csak költséghatékonysága, de páratlan biztonsága miatt is népszerű.



Könnyen telepíthető és felügyelhető, méretezhető és mégis biztonságos. Használatával a cég teljes kimenő és bejövő kommunikációs forgalma gyorsan, teljesen és hatékonyan biztosítható.

A Novell teljes hálózati biztonsági megoldása összesen hat határbiztonsági alkalmazást és integrált felügyeleti platformot biztosít. A ***SUSE LINUX*** alapú alkalmazás átfogó biztonsági infrastruktúrája a betörők, vírusok, férgek, levélszemét és betörések jelentette biztonsági fenyegetések ellen is védelmezi a vállalatokat.

Tűzfal

A ***Novell Security Manager*** tűzfala az összes hálózatról érkező kommunikációs csomag fejlécét átvizsgálja, és a kommunikációs folyamatok megsértésének felderítése végett nyomon követi az eseményeket.

Blokkolja a kommunikációs forgalmat, ha az nem felel meg a portokra, protokollokra, valamint a várt forrásokra és célhelyekre beállított szabályoknak (állapotfigyelő csomagvizsgálat és alkalmazásszintű szűrés). Képes megvédeni a forgalmat a vezeték nélküli eszközöktől is.

A számos hálózati kapcsolat felügyeletének egyszerűsítésére és a teljesítmény növelésére az ***Astaro*** biztonsági proxykat alkalmazza a legfontosabb protokollokhoz (például a ***HTTP***, ***DNS***, ***SOCKS***, ***POP3***, ***Ident*** és ***SMTP*** esetében). A fontos tűzfalfunkciók közé tartozik még a hálózati címfordítás,

a maszkolás, valamint a szolgáltatmegtagadási támadásokkal szembeni védelem.

Virtuális magánhálózat

A távoli felhasználók számára, akik az internetet használják kommunikációra, a *Novell Security Managerben* található ***virtuális magánhálózat (VPN)*** rendkívüli mértékben képes csökkenteni a kommunikációs költségeket azzal, hogy megszünteti a drága bérelt vonalak szükségességét. Mivel a *VPN*-átjáró rendkívül rugalmas – sokféle architektúrát támogat –, *VPN* kapcsolat teremthető a távoli irodákkal, az otthoni munkahelyekkel és egyéb nyilvános helyekkel (például szállodák konferenciatermeivel). A legfejlettebb titkosítási algoritmusok és a hitelesítési módszerek, valamint a *VPN*-kliensek széles köre áll rendelkezésre.
A *Novell Security Manager* saját tanúsítványhatósággal rendelkezik a digitális aláírások egyszerű és biztonságos kezelésére, és teljes mértékben támogatja a nyilvános kulcsú titkosítás használatát.

Védelem a behatolásokkal szemben

A *Novell Security Manager* behatolásvédelmi összetevője egy több mint 2000 mintát és szabályt tartalmazó adatbázis alapján (amelynek karbantartását a vezető nyílt forráskódú ***Intrusion Detection Snort***™ behatolásvédelmi projekt végzi) felderíti és blokkolja az alkalmazás- és protokollspecifikus támadásokat. Beállítható, hogy gyanús tevékenység észlelése esetén e-mailben értesítse a rendszergazdát, vagy azonnal blokkolja a gyanús forgalmat a tűzfalon keresztül. Ennek felügyeletét az egyedi szabályok vagy a teljes kategóriák szintjén is végezheti. A szabályokat a *Novell Up2Date* szolgáltatás folyamatosan frissíti; ezekhez új szabályok adhatóak, vagy a meglévők testre szabhatóak. Leállítható, vagy korlátozható a legújabb kommunikációs formákkal kapcsolatos tevékenység: például az azonnali üzenetküldés, a csevegés vagy az ***egyenrangú (peer-to-peer) hálózatok***. Ez kritikus fontosságú kiegészítője lehet a biztonságnak, különösen, mivel egyelőre kevés védekezési mód létezik az ilyen típusú visszaélésekkel szemben.

Vírusvédelem

A *Novell Security Manager* szűrő keretrendszere átvizsgálja az e-mail üzeneteket, fájlokat és a webes forgalmat a vírusok, férgek, trójai programok és egyéb rosszindulatú szoftverek után kutatva. A *Novell Security Manager* kétféle típusú vírusvédelmet kínál: a hagyományos e-mailekhez és fájlokhoz, valamint a webböngészőben letöltött e-mailekhez és fájlokhoz is.
A *Novell Security Manager* sokféle vírusellenőrző módszert használ annak érdekében, hogy a lehető legtöbb vírust megfogja: elemzi az e-maileket és a csatolmányokat az ismert, vírusokhoz társítható kódok megkeresésére, heurisztikus módszerekkel keres az ismert vírusmintázatokhoz hasonló kódokat és ezeket hagyja végrehajtódni egy védett környezetben, ahol a problémák a megfertőződés veszélye nélkül felismerhetők.
A gyanús kódokat összeveti a *Kaspersky Lab* adatbázisával, amely a világ egyik legnagyobb, 100 ezer vírusjellemzőt tartalmazó listája. Az átjáróban történő vírusellenőrzés lehetővé teszi az új vírusokra való gyors reagálást, mielőtt azok elérnék a belső rendszereket. Ez a szolgáltatás az asztali vírusellenőrzők kritikus fontosságú kiegészítője, mert azokat gyakran bonyolult a teljes szervezetben frissíteni. Megadható, hogy el kívánja-e dobni a gyanús leveleket és csatolmányokat, vagy visszautasítja azokat a küldőnek szóló üzenettel, illetve figyelmeztetéssel átengedi a felhasználó felé vagy karanténba zárja őket, hogy az adminisztrátor megvizsgálhassa azokat és megtehesse a szükséges intézkedéseket.

Spam elleni védekezés

A *Novell Security Manager* számos spamfelismerő módszert használ a kéretlen levelek azonosítására és blokkolására. Ellenőrzi a levelek forrását az ismert spamküldők listájával összehasonlítva, saját fekete- és fehérlistákat hoz létre, szabályokat és mintázatokat használ a levelek szövegének elemzésére és hozzájuk rendel egy „spam pontszámot". A kívánt küszöbérték beállításával a gyanús levelek eldobhatók, visszautasíthatók egy küldőnek szóló üzenettel, figyelmeztetéssel átengedhetők a felhasználó felé, illetve karanténba zárhatók, ahol a rendszergazda megvizsgálhatja őket és megteheti a megfelelő intézkedéseket. Ez a rugalmasság lehetővé teszi a finom egyensúly megteremtését a spamszűrés és jó levelek véletlen blokkolásának elkerülése között. A *Novell Security Manager* jelentést készít a spamüzenetek számáról és méretéről, így felismerhetőkké válnak a mintázatok és a trendek is.

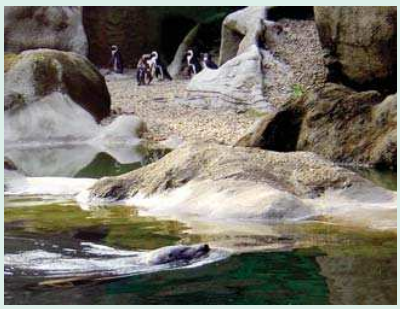
Barangolásvédelem (URL-szűrés)

Az internet rendkívül fontos eszköz a cégek számára, de ha a munkatársak túl sok időt töltenek a weben barangolva, a termelékenység csökkenhet, illetve ha nem helyénvaló vagy jogvédett anyagokat töltenek le, akkor jogi problémák is felvetődhetnek. A barangolásvédelem lehetővé teszi a webes tevékenységek védelmét a webhasználati irányelvek kidolgozásával.
A *Novell Security Manager* segítségével a cég vezetése 58 különféle kategória használatával határozhatja meg a webhasználati irányelveket, ilyenek például a szex, a szerencsejátékok, a törvénytelen tevékenységek, illetve az amúgy törvényes, de a munkához nem szükséges tevékenységek mint például a vásárlás, az árverések látogatása, a szórakozás vagy a munkakeresés. Az üzemeltetést végző osztály egyszerűen mérheti a webes tevékenységeket és jelentéseket készíthet ezekről a problémák azonosítására és a prioritások megadására, vagy blokkolhatja bizonyos ***URL*** kategóriák elérését, így azok hozzáférhetetlenek lesznek a felhasználók számára.
A *Novell Security Manager* egy (a *Cobiontól* származó) 20 millió kategorizált webcímet tartalmazó adatbázist használ, ami jelenleg a kereskedelmi forgalomban kapható legnagyobb lista. Ez a lista természetesen további saját fekete- és fehérlistákkal bővíthető. A felhasználók különféle csoportjai számára pedig külön fekete- és fehérlisták is létrehozhatók. ■

Tóbiás a befutó

Előző lapszámunkban felhívást tettünk közzé; olvasóink segítségét kértük pingvin-örökbefogadási vállalkozásunkhoz. Ezennel közhírré tétetik a Linuxvilág totyogó befutója: nem más ő, mint Tóbiás (szül. 1998, Rostock).

Amiképpen e teltkarcsú palimadár belopta magát a szavazók szívébe, aképpen lopakodtam be az ***Fővárosi Állatkert Kis-sziklájának*** pingvin-lakta tövéhez, hogy közelebbről szemrevételezhessem a legtetszőbb ábrázatú példányt. Az egyik gondozó hölgyet faggattam, merre lelem ***Tóbiást***. A kedves hölgy feleletéből megtudhattam, hogy a keresett egyed éppen tanúkat mellőzve s ezért láthatatlan elvonult kötelességét végezni – tojásain ül…
Némiképp gyanakodván, hogy tán áldozata lettem az elhamarkodottságnak, miszerint feltételeztem, hogy ***Tóbiás*** ivarát tekintve hím, rákérdeztem egyenest a valóra. Abban a megtiszteltetésben lehetett részem, hogy e madár szinte érkezésemmel egy időben váltott nemet. Tudniillik pingvinéknél csemetekorban nem lehet csalhatatlan biztonsággal megmondani leányságot, avagy legénységet (leszámítva fölös és kockázatos vizsgálatok igénybevételét). Így hát marad a tapasztalatokon és megfigyeléseken nyugvó szemek általi vizsgálat, mint például testnagyság-vizslatás, szexuális pozitúrák kifürkészése és rácsodálkozás ülep alatti tojásokra. Míg ***Tóbiás*** ez utóbbiakon foglalt helyet szemérmesen, én magánéletéről faggatóztam. Nyilvánvalóan korábbi időkre tehető eladósorba jutása, s ***Gilbert*** ama madártárs, ki nem hagyta őt pártában hervadni el.
Ők ketten számos dologban hasonlítanak, talán legfőképpen a megtévesztő látszat (és tojások) keltésében. Ugyanis ***Gilbertről*** is kiderült, hogy nem éppen férfi a talpán...
Letépve az előítéletek láncait, akár pingvin a heringes vödörben halászgattam tovább a szerelmi hajlandóság zavaraiban. Úgy látszik, ha testi szomjúságot kell oltani, e szárnyasok sem szőrszálhasogatóak párválasztásuk folyamán. Ellenben igen monogám és hűséges természetűek, s noha az erény mindenképpen tiszteletet követel, mégiscsak örömteli lenne idővel szülőkből nagyszülőkké is válnunk. Reményekre adhat okot, hogy a minap friss szállítmány érkezett az Állatkertbe tucatnál is több pingvinfiú személyében.



Bízva a rajtaütés sikerében, hogy a sajtótájékoztató alkalmából személyesen is kokettálhatunk a jómadárral, s megtudhatjuk, hogy látszat- és tojáskeltésen kívül mikkel múlatja szabadidejét, illetve mily pontos paraméterekkel bír is ő derékbőségétől kezdve gyomra heringbefogadói képességéig.

(Folyt. Köv.)
Halusz Léna

A nők és a Linux

Interjú Konkoly Katalinnal, az MLDP idei díjazottjával.

Bár egyesek véleménye szerint a Linux HOGYAN-ok és más dokumentumok magyarra fordítása puszta időpocsékolás – mondván a számítástechnika „hivatalos nyelve" úgyis az angol – tény, hogy különösen a kezdők számára az anyanyelven rendelkezésre álló irodalom óriási segítség. Éppen ezért jött létre Daczi László vezetésével a Magyar Linux Dokumentációs Projekt, amely minden évben díjazza legjobb fordítóit, lektorait. Lapunk az egyik díjazottat, Konkoly Katalint kérdezte arról, hogyan is keveredett a linuxosok közé...

Elöljáróban meg kell mondanom, hogy nem érzem magamat méltónak a megtiszteltetésre (úgy értem, vannak nálam sokkal több fordítást végzett önkéntesek), mert csak nemrég kezdtem (körülbelül egy éve) ***Linuxszal*** foglalkozni, a fordítói-lektori munkát is csak pár hónapja végzem. Az is meglepett – bár nagyon jólesett –, hogy ***dacas (Daczi László)*** díjazásra érdemesnek tartotta az eddigi munkámat.
A jászberényi ***Lehel Vezér Gimnáziumban*** tanítok, német-francia szakos tanárként, tehát papírforma szerint semmi közöm a számítástechnikához. 2004. júniusában került az első ***Linux*** a gépemre, azonnal elfoglalva az addigi ***Windows*** helyét. Ez épp az ***UHU 1.1*** volt, és mivel meglehetősen szokatlan volt az új rendszer, feliratkoztam az ***UHU***-listákra, ahol nagyon sokat tanultam, és sok jó barátra tettem szert. Többek között ott ismertem meg ***Emersont (Tersánszky Csaba)***, aki az ***UHU Live CD***-ket készíti. Vele aztán egy ***CD*** erejéig együtt is dolgoztunk, ahol nekem – mivel programozáshoz nem értek – az apró helyesírási és gépelési hibák kigyomlálása, valamint a ***CD*** tesztelése volt a feladatom. Mindez a nyári szünetben zajlott, és oly mértékben fellelkesültem a szabad szoftverek, a ***Linux***, a közösségi szellem világa láttán, hogy a tanévnyitó értekezlet előtti napon ***Szász Endre Istvánnal (Szaszka)*** tartottunk egy ***Linux***-bemutatót a tantestület számára, az ***UHU Live*** segítségével. Sajnos a kollégák között nem sikerült elérnem, hogy nagyobb számban használjanak ***Linuxot*** (gyakorlatilag egyikük sem tért át...), de azt fontos eredménynek érzem, hogy legalább ***hallottak*** a szabad szoftverekről.
Ősszel már úgy éreztem, én is tartozom a linuxos közösségnek azzal, hogy amihez értek, azt valami módon közkinccsé tegyem. Sok fórumon azt tapasztaltam, hogy az újonnan áttért felhasználóknak gyakran gondjuk van a dokumentációkkal, hiányos angol nyelvismeretük miatt. Ekkor kezdtem el keresgélni a neten valamilyen szabad szoftverekkel kapcsolatos fordítási lehetőség után, s így bukkantam a ***tldp.fsf.hu*** oldalára, ahol örömmel láttam, hogy önkéntes fordítókra mindig szükség van :). Jelentkeztem is ***dacasnál*** (***Daczi László***), és a fordításra váró ***HOGYAN***-ok listájából azonnal kiszemeltem a ***HOWTO Encourage Women in Linux (Hogyan bátorítsuk a nőket a Linux használatára)*** című. Hamarosan el is készült, népszerű is lett (a ***HUP*** letöltési statisztikája szerint). Azóta egy nagyobb frissítésen dolgoztam: a ***System Administrators Guide-on (Linux rendszeradminisztrátorok kézikönyve)***, és két kisebb lektorálási munkám volt (***A katedrális és a bazár*** jegyzetei, illetve a ***Webkamera-HOGYAN***).
Ez év tavaszán meghívást kaptam a ***Debrecenben*** megrendezett V. ***LOK*** konferenciára előadóként. A téma, amiről beszéltem: A szabad szoftverek felhasználásának különböző lehetőségei (***Windows*** alatt, ***live Linuxszal***, illetve telepített ***Linux*** rendszeren). Az előadás sikerén felbuzdulva a tantestületben is megemlítettem, hogy szívesen vállalok a diákok számára is előadást, akár osztályfőnöki, akár számítástechnika óra keretében. A kollégák mindez ideig nem éltek az ajánlatommal.
Ez persze nem szegte kedvemet, mert úgy gondoltam: ha kissé illegálisan is, de a saját tanóráimon is beszélhetek a gyerekeknek a szabad szoftverekről. Így aztán az iskolai laptop és projektor segítségével a német és francia óráimon tartottam néhány érdeklődő csoportomban linuxos bemutatót – természetesen szigorúan csak a tanév utolsó napjaiban, miután a tantervben előírtakat teljesítettük. Örömmel láttam, hogy néhányukat megérintette a ***Linux*** világa, segítséget, tanácsot, telepítőcédéket kértek tőlem. Természetesen nem célom minden diákom „megtérítése", de fontosnak tartom, hogy ezen a módon is tudomást szerezzenek a szabad szoftverekről, a ***Linuxról,*** s működés közben is lássák.
Sok önálló itthoni ***Linux***-tanulás és gyakorlás után úgy éreztem, egyedül már nem tudok továbblépni e téren, ezért tavasszal beiratkoztam egy rendszerinformatikus tanfolyamra, ahol nagyon jó társaságba kerültem, és sok új ismeretre tettem szert. Az iskolám anyagilag nem támogatja ezt a képzésemet.

Kávéfőzés lépésről lépésre (5. rész)

A programozó pedig ablakokat és nyomógombokat hozott létre. Így a felhasználó kedve szerint kattinthatott, az ő legnagyobb megelégedésére...

Ha a kedves Olvasó idáig figyelemmel kísérte a cikksorozatot, bátran állíthatja, hogy azokkal az objektumközpontúsággal kapcsolatos fogalmakkal, melyek sokaknak mindössze valamilyen titkos szekta misztikus varázsigéit jelentik, bizony már szoros baráti viszonyban áll. Az osztály, az objektum, az interfész, az öröklődés és a kivételkezelés hallatán szívéhez közel álló eszközök és módszerek jutnak eszébe, melyek segítségével egy alkalmazás tervezése, megvalósítása és karbantartása is egyszerűbbé és gyorsabbá válik.

Joggal jelenthetjük ki tehát, hogy mindent megtettünk saját magunk, programozók örömére. Azonban ne felejtsük el, hogy nincs programozó felhasználó nélkül, hiszen az sem jó színész, akit a rendezőn kívül más nem látott szerepelni. Ezért most, hogy már elhangzottak a kenetteljes szentbeszédek a programozókat kiszolgáló objektumközpontú technikák mellett, eljött az ideje, hogy a felhasználóknak is kedvezzünk. A felhasználó csak annak tud örülni, amit lát, adjunk tehát most a külcsínre.

Mivel mindig hangsúlyozom, hogy az informatikában is a célnak kell meghatároznia az eszközt, és nem fordítva, először gondoljuk meg, milyen programozási környezet lenne a legalkalmasabb grafikus felhasználói felület (*GUI, Graphical User Interface*) létrehozásához. A cikksorozat eddigi részei a hűséges Olvasóban talán már gyanút ébreszthetnek, ám mielőtt kimondanánk a boldogító igent a Java oldalán, mérlegeljük a mellette és az ellene szóló érveket.

A *Java* objektumközpontú, ami a jelenleg ismert leghatékonyabb módszer ablakozó rendszer programozására. Emellett a futtatókörnyezettől bájtkód szinten független, ezért könnyen hordozható. Ugyanakkor lehetőséget ad akár az operációs rendszer által nyújtott eszközkészlet, akár egy saját, minden rendszeren azonosan megjelenő eszközkészlet használatára. A képernyőn megjelenő vezérlőkhöz köthető eseménykezelők pedig teljes és jól átlátható rendszert adnak.

Ezzel szemben tudni kell, hogy egy *Java* alkalmazás virtuális gépet igényel (*JVM, Java Virtual Machine*). Ez minden szélesebb körben használt operációs rendszerre elérhető, de nem szükségszerű, hogy fel is telepítették. Erőforrásigénye sem nevezhető kimondottan alacsonynak. Így eleshetünk egyes, matuzsálemi korú berendezésekkel dolgozó felhasználóktól. Azt se felejtsük el, hogy bár létezik megoldás 3 dimenziós ábrák és animációk használatára, kevés ilyen elemeket tartalmazó Java alkalmazást láthatunk.

Ezek után a készítendő program dönti el, hogy a mérleg nyelve merre dől. Én személy szerint azért teszem erre a nyelvre a voksomat, mert így könnyen elérhetem, hogy programjaimat, melyeket a Microsoft által készített operációs rendszereken kell bemutatnom, szeretett ***Linuxom*** alatt írhatom meg. Mindenki maga döntsön, majd a helyes döntést követően olvassa tovább a cikket.

Ismerjük el, egy kissé kezd unalmas lenni, hogy mindenki állandóan a világot köszöntgeti. A sorozat első részében már mi is megtettük, első *Java* nyelven írt grafikus alkalmazásunk legyen ennél valamivel érdekesebb. Lássuk a kódot *(1. kód)*!

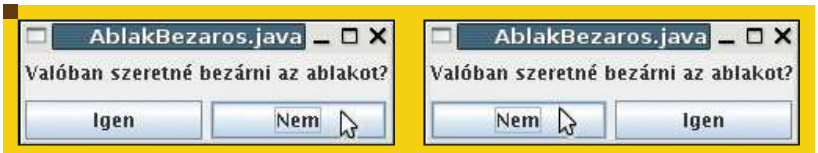
Minden szerénységem megtartása mellett meg kell jegyezzem, hogy tisztában vagyok azzal, mennyire lebilincselőek az írásaim. Ennek ellenére most arra kérném a kedves Olvasót, hogy mielőtt továbbolvasná a cikket, gépelje be az itt látható program kódját, fordítsa és futtassa azt.

Két nyomós indokom van arra, hogy arra kérjem, hogy olyan sorokat gépeljen, melyeknek jelentése nem azonnal adódik, s így ez esetleg fárasztó terhet jelenthet.

Az első, hogy az alkalmazás egyfajta feladvány, ugyan nem a komoly fejtörést okozók fajtájából, de mindenképpen tanulságos. A kód azonnali elem-

1. kód

```
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JButton;
import javax.swing.JPanel;
import java.awt.GridLayout;
import java.awt.Container;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

/**
* Az AblakBezaros jatek megvalositasa.
*/
public class AblakBezaros extends MouseAdapter
➥implements ActionListener {

    /**
    * A teljes ablak
    */
    private JFrame ablak;

    /**
    * A kerdest tartalmazo cimke
    */
    private JLabel szovegCimke;

    /**
    * Bal- es jobboldali nyomogombok
    */
    private JButton balGomb;
    private JButton jobbGomb;

    /**
    * Igen es Nem allandok
    */
    private final String igenSzoveg = "Igen";
    private final String nemSzoveg = "Nem";

    /**
    * A konstruktor letrehozza az ablakot,
    * felteszi ra a vezerloket, es beallitja
    * az esemenykezeloket.
    */
    public AblakBezaros() {
  // ablak es vezerlok letrehozasa
  ablak = new JFrame("AblakBezaros.java");
  szovegCimke = new JLabel("Valóban szeretné
  bezárni az ablakot?",
      JLabel.CENTER);
  balGomb = new JButton(igenSzoveg);
  jobbGomb = new JButton(nemSzoveg);

  // gombok onallo panelt kapnak
  // az elrendezeskezelo miatt
  JPanel gombokPanelje = new JPanel(new
  GridLayout(1, 2, 5, 5));
  gombokPanelje.add(balGomb);
  gombokPanelje.add(jobbGomb);

  // az ablak tartalmanak feltoltese
  Container tartalom = ablak.getContentPane();
  tartalom.setLayout(new GridLayout(2, 1, 5, 5));
  tartalom.add(szovegCimke);
  tartalom.add(gombokPanelje);

  // esemenykezelok mind ebben az objektumban
  balGomb.addMouseListener(this);
  jobbGomb.addMouseListener(this);
  balGomb.addActionListener(this);
  jobbGomb.addActionListener(this);

  // vegso "simitasok"
  ablak.setDefaultCloseOperation
  (JFrame.DO_NOTHING_ON_CLOSE);
  ablak.pack();
  jobbGomb.requestFocusInWindow();
  ablak.setVisible(true);
    }

    /**
    * Akkor fut le, ha valamelyik gomb fole kerul
    * az egerkurzor. Ekkor felcsereli a ket gomb
    * szoveget es a fokuszt is athelyezi.
    */
    public void mouseEntered(MouseEvent esemeny) {
  Object forras = esemeny.getSource();
  if (((JButton) forras).getText().equals
  ➥(igenSzoveg)) {
      if (((JButton) forras).equals(balGomb)) {
     balGomb.setText(nemSzoveg);
     jobbGomb.setText(igenSzoveg);
     balGomb.requestFocusInWindow();
      } else {
     balGomb.setText(igenSzoveg);
     jobbGomb.setText(nemSzoveg);
     jobbGomb.requestFocusInWindow();
      }
  }
    }

    /**
    * Akkor fut le, ha valamelyik gomb benyomodik.
    * Ha a benyomott gomb felirata "Igen", az
    * alkalmazas bezarul.
    */
    public void actionPerformed
    ➥(ActionEvent esemeny) {
  Object forras = esemeny.getSource();
  if (((JButton)
forras).getText().equals(igenSzoveg)) {
      System.exit(0);
  }
    }

    /**
    * Belepesi pont, egy peldanyt hoz letre az
    * osztalybol.
    */
    public static void main(String[] args) {
  new AblakBezaros();
    }

}
```



1. ábra Ön mit tenne az adott szituációban? (2 pont)

zésével a megoldás magáért beszélne. A másik, hogy nehéz úgy egy grafikus alkalmazást megérteni, hogy az eredményét pusztán a képzelőerőnkre bízzuk. Különösen igaz ez akkor, ha még nem sok tapasztalattal rendelkezünk a grafikus felhasználói felületek készítése területén. Ezért mindenképpen javaslom a játék azonnali kipróbálását. A felejthetetlen játékélményt követően nézzük meg, hogyan épül fel az alkalmazás. Az előző részben is használtuk az `import` kulcsszót, most viszont már-már ijesztő méretekben fordul elő. Az `import` a megadott osztályt hozza be az alkalmazás névterébe. Ennek révén rövidebben hivatkozhatunk ugyanarra az osztályra, vagyis a behozatal után nem kell minden egyes helyen kiírni, hogy `javax.swing.JButton`, elegendő a `JButton` használata. Ennek természetesen megvan az a hátulütője, hogy nem hozhatunk létre saját `JButton` nevű osztályt az alkalmazásban, de erre legtöbbször nincs is szükség.

Miután csak három csomagból használunk osztályokat, nevezetesen a `javax.swing`, a `java.awt` és a `java.awt.event` csomagokból, az első tíz sort lerövidíthettük volna így is:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
```

Így a csomagok összes osztálya a névtérbe került volna. Ez viszont azért nem javasolt, mert ha a későbbiekben akár nekünk, akár másnak elemeznie kell a kódot, nem egyértelmű, hogy melyik csomag miért kerül behozatalra, más szóval milyen osztályokból építkezik a program. Nagy valószínűséggel hibát nem követhetünk el, mert az adott helyen létrehozott osztályok felülbírálják a behozott neveket, mégsem jó programozói gyakorlat olyannal szennyezni a névteret, ami nem oda való.

Behoztunk tehát tíz olyan osztályt, amivel a sorozatban eddig nem találkoztunk. Az aggodalom azonban csak addig tarthat míg fény nem derül a ***Swing*** és az ***AWT*** mibenlétére. A kezdetekben grafikus felhasználói felületek létrehozásához ***Javaban*** az ***AWT (Abstract Windowing Toolkit)*** állt csak rendelkezésre. Ez egy olyan osztályokból és interfészekből álló halmazt biztosított a programozónak, mellyel ablakokat hozhatott létre, rajta vezérlőket, és kezelhette a különféle eseményeket.

Az egyik legnagyobb hátránya ennek a megoldásnak pont az absztraktságában rejlett. Miután az ***AWT*** használatával a programozó teljes mértékben a futtatókörnyezetre bízta a megjelenítést, csak olyan elemek voltak elérhetőek számára, amelyek minden rendszeren megtalálhatók. Így az ***AWT*** a különféle ablakozó rendszerek beépített grafikus eszközkészletének legszűkebb keresztmetszetét kínálta csupán. Ráadásként az sem volt minden esetben elfogadható, hogy nehezen megjósolható a program kinézete is, hiszen mindent a futtatókörnyezet dönt el.

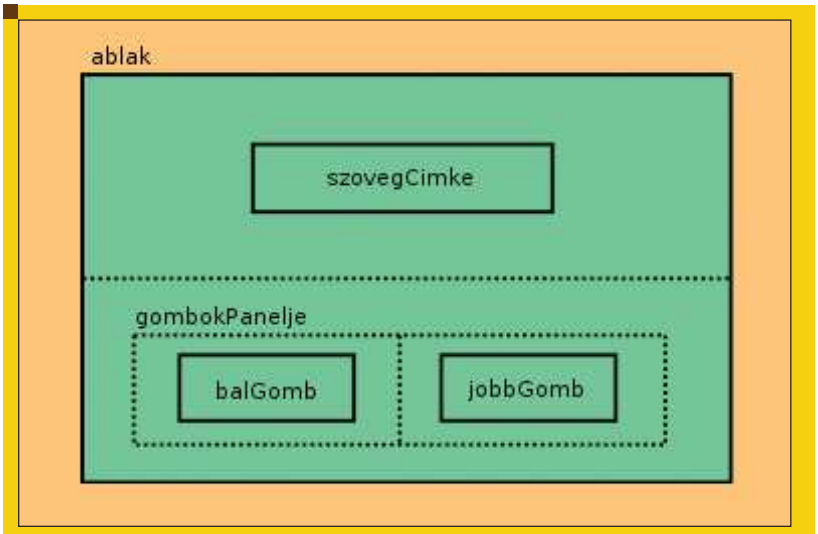
A ***Java 1.3*** megjelenésével megérkezett a ***Swing***, amely már egy önálló eszközkészlet a nyelvben. Teljesen *Java*-ban íródott, ami jópár előnnyel jár. A grafikus alkalmazás így nem veszít hordozhatóságán, ugyanakkor több és nagyobb tudású vezérlőket használhat, mivel az ablak nem függ a futtatókörnyezettől. A kinézet egységes minden rendszeren, sőt, témákat is alkalmazhatunk a még egyedibb megjelenítés érdekében.

A ***Swing*** hátterében az ***AWT*** áll, így nem lehet azt állítani egy alkalmazásról, hogy „tisztán swinges". A vezérlőknél még igen szembetűnően minden osztály neve elé oda kell ragasztani egy `J` betűt (`Button` helyett `JButton`), az eseménykezelő osztályok és interfészek viszont a jó öreg ***AWT*** csomagból valók. Ez nem baj, mert eddig is kifogástalanul működtek, és nem lett volna értelme újraírni őket. A két csomag remekül kiegészíti egymást, és jól használhatók, ha nem hanyagoljuk el az ***API*** tanulmányozását.

Ugorjuk át most az osztály fejlécét és nézzük meg, mi van a törzsében. Először a tagváltozók felsorolását látjuk. Egy ablakban egy szövegcímkét és két nyomógombot jelenítünk meg, ez négy önálló objektumot jelent. Itt találhatunk továbbá két állandót, melyek a gombok feliratait adó szövegfüzérek. A `final` kulcsszóval biztosítjuk, hogy a változók értéke a későbbiekben ne legyen módosítható. Az ilyen mezők életükben egyszer kaphatnak értéket. Ez a művelet kerülhetett volna a konstruktorba is, ekkor azonban feltétlenül ügyelni kellett volna arra, hogy még az első állandóra való hivatkozás előtt ellássuk azt a megfelelő kezdőértékkel.

Az ezután következő konstruktor hozza létre az ablakot. Első sorában a `JFrame` osztályt példányosítjuk, és a kapott referenciát az ablak tagváltozóban tároljuk. A `JFrame` egy ***Swing*** összetevő, amely egy ablakot képvisel. Több konstruktora is van, mi most azt a változatot használtuk, mellyel rögtön meg is lehet adni az ablak címét. Fontos tudni, hogy ettől a sortól az ablak még nem jelenik meg. Alapértelmezésként a láthatósága hamis, ami azért jó, mert így a háttérben felépíthetjük az ablakot, amit csak akkor jelenítünk meg, ha már teljesen készen van.

A következő sorban hozzuk létre a szövegcímkét. Konstruktorában megadjuk a szöveget, továbbá, hogy a rendelkezésre álló helyen középre igazítsa azt. Ez a címke még nem az ablak része. Semmilyen módon nem fejeztük ki, hogy a vezérlő, amit létrehoztunk, hol jelenjen meg. Ugyanez a helyzet az ezután álló sorokban létrehozott `Igen` és `Nem` feliratú gombokkal. Az első sorokban tehát még csak különálló objektumokat hoztunk létre. Ahhoz, hogy ezek az ablakban megjelenjenek, ki kell választanunk egy elrendezéskezelőt. Az elrendezéskezelő az a láthatatlan kéz, amelyik kirakosgatja az ablakhoz hozzáadott elemeket az ablak területére. Mi a `GridLayout` nevű elrendezéskezelőt használjuk, amely egy táblázatként értelmezi az ablakunkat. A `GridLayout` példányosításakor megadhatjuk, hogy



2. ábra Az ablak felépítése

hányszor hányas táblázatot szeretnénk használni. Ha például kétszer ketteset választunk, akkor az első hozzáadott elemet az ablak bal felső sarkába helyezi, a másodikat a jobb felsőbe, a harmadikat a bal alsóba, a negyediket pedig a jobb alsóba.

Most egy olyan képet szeretnénk elérni, ahol a címke az ablak felső részében van, a gombok pedig alul, egymás mellett. Ennél az elrendezéskezelőnél maradva ezt úgy tudnánk elképzelni, hogy a kétszer kettes tábla felső sorában álló cellákat összevonjuk. Ez azonban itt nem megoldható. Ehelyett úgy tudjuk ugyanazt a hatást kelteni, hogy az ablakot két részre osztjuk csupán. A felső rész tartalmazza majd a szövegcímkét, az alsó pedig egy új panelt. Az új panel egy bal és egy jobboldali cellából fog állni, ezekbe kerülnek a gombok.

A vezérlők objektumainak létrehozása után ezt a panelt készítjük el.

A `JPanel` kosntruktorának rögtön át is adunk egy új `GridLayout` objektumot, amely 1 sorból, 2 oszlopból álló táblázatban rendezi el az elemeket, és mind vízszintes, mind függőleges irányban 5 képpontot hagy ki a cellák között. Miután a panel rendelkezik egy elrendezéskezelővel, az elemek hozzáadása egy egyszerű add metódussal történik. Az ablakon belüli tényleges, x és y koordinátákkal történő pozicionálás már a `GridLayout` dolga.

Megjegyzem, hogy a `JPanel`-nek van olyan konstruktora, amely nem vár paramétert, tehát nem kell megadnunk az elrendezéskezelőt. Ilyenkor az alapértelmezett `FlowLayout` végzi a vezérlők helyének meghatározását. Ez balról jobbra folyamatosan tölti fel az ablakot a vezérlőkkel, és ha a következő már nem fér ki az adott sorba, akkor a következőben folytatja. Ez a viselkedés nagyon közel áll egy szövegszerkesztőben megtekintett bekezdésre. Az ablak mérete alapvetően befolyásolja itt a vezérlő helyét, mi több, az ablak átméretezésével az egész megjelenés megváltozhat, ezért csak kevés értelmes felhasználása van ennek az elrendezéskezelőnek.

Megváltoztatható az elrendezéskezelő egy `JPanel` esetében a `setLayout` metódussal. Viszont mivel a `JPanel` paraméter nélküli konstruktora létrehoz egy példányt a `FlowLayout` osztályból, a felülbírálással egy felesleges objektum létrehozását okozzuk. Ezért, amikor csak lehet, érdemes megadni még a konstruktorban a saját elrendezéskezelőt, hogy elkerüljük az értelem nélküli példányosítást.

Miután elkészültünk a panellel, felrakhatjuk az elemeket az ablakra. Ehhez közvetlenül nem adhatjuk hozzá a vezérlőket, viszont az ablaktartalom egyszerűen lekérdezhető a `getContentPane` metódussal. Ehhez először hozzárendelünk egy `GridLayout` elrendezéskezelőt, 2 sorral, 1 oszloppal, és 5 képpontos hézagokkal. `JFrame` esetében nem is lett volna lehetőségünk a konstruktorban ezt meghatározni, így ezt itt tesszük meg. Ezt követően az add metódussal felhelyezzük a címkét és a gombok paneljét az ablakra.

Mielőtt megjelenítenénk az ablakot, beállítjuk az eseménykezelőket. Elsőként az egérrel kapcsolatos eseményekkel foglalkozunk. Bármely osztály, amely megvalósítja a `MouseListener` interfészt, hozzárendelhető egy vezérlőhöz. Így, ha a vezérlő területére belép az egérkurzor, vagy kilép onnan, kattintás történik, esetleg a felhasználó lenyomja az egér valamelyik gombját, vagy felengedi azt, az interfészt megvalósító osztály megfelelő metódusához kerül a vezérlés.

Viszont ne felejtsük el, hogy egy interfész megvalósítása az összes előírt metódus megvalósítását jelenti. Jelen esetben nekünk erre nincs szükségünk, hiszen csak az egér mozgását szeretnénk követni. Ha maradnánk a `MouseListener` interfésznél, a többi előírt metódust üres törzzsel ugyan, de szerepeltetni kellene a programban. Ennek kiküszöbölésére használható a `MouseAdapter`, amely egy olyan osztály, ami megvalósítja a `MouseListener` interfészt, üres függvénytörzsekkel. Ha ebből származtatjuk saját osztályunkat, csak azokat a metódusokat kell felüldefiniálnunk, melyeket ténylegesen használni szeretnénk.

Ezzel úgyszólván átvágtuk a gordiuszi csomót. Egyetlen osztályunk, az `AblakBezaros` kiterjeszti a `MouseAdapter`-t, ezt jelezzük is az osztály fejlécében. Elkészítjük a `mouseEntered` metódust, melynek meghívását az az esemény válthatja ki, ha az egérkurzor belép a vezérlő területére. Az eseménykezelővel később foglalkozunk, itt, a konstruktorban csak annyit állítunk, hogy a bal- és jobboldali gomb egérrel kapcsolatos eseményeivel jelen objektum (`this`) törődik. Ezt fejezik ki az `addMouseListener` hívások.

Gombok esetében ennél többet is figyelhetünk. Egy nyomógombbal kapcsolatos legfontosabb esemény az, ha lenyomják. Ez történhet egérrel is, de a billentyűzet segítségével is odalépkedhet a felhasználó, és rátenyerelhet a szóköz billentyűre. Ezt az eseményt is szeretnénk

kezelni, amit egy olyan osztály példányával tehetünk meg, amely megvalósítja az `ActionListener` interfészt. Az előzőekkel szemben nincs `ActionAdapter` osztály, de ez két ok miatt sem gond. Egyrészt az `ActionListener` csak egyetlen metódus megvalósítását írja elő, pont azt, amire szükségünk van. Másrészt többszörös öröklődés híján újabb osztályt már ki sem terjeszthetnénk.

A szemfüles Olvasó felfigyelhet arra a tényre, hogy a gombra történő kattintás egyszerre két eseménykezelőt is érint. Egyrészt az `ActionListener` interfészt megvalósító osztály példányának `actionPerformed` metódusa, másrészt a `MouseAdapter`-t kiterjesztő, vagy a `MouseListener`-t megvalósító osztály példányának `mouseClicked` tagfüggvénye. Ezt jelen esetben nem valósítottuk meg, de használata nem ütközött volna semmilyen problémába. Ilyen helyzetben mindkét függvény meghívásra kerül.

A „végső simítások” megjegyzéssel illetett részben elsőként megadjuk, hogy mi történjen, ha a felhasználó az ablakkezelőn keresztül próbálja

meg bezárni az alkalmazást. Ez legtöbbször a jobb felső sarokban található `X`-re való kattintást jelenti.

A `setDefaultCloseOperation` metódusnak a `JFrame` osztály megfelelő konstansai közül kell átadnunk egyet. Az alapértelmezett művelet az elrejtés (`HIDE_ON_CLOSE`). Ezt bíráljuk felül azzal, hogy ne történjen semmi (`DO_NOTHING_ON_CLOSE`). A leggyakrabban használt művelet minden bizonnyal a kilépés (`EXIT_ON_CLOSE`), ám itt nem erre van szükségünk.

A `pack` metódus meghívásával életre keltjük a láthatatlan kezet, amely szépen elrendezi az ablaktartalmat. Majd mivel fontos, hogy a fókusz ne a bal oldali gombon legyen, a jobboldalinak meghívjuk a `requestFocusInWindow` metódusát. Ez még nem biztosítja, hogy a vezérlő valóban meg is kapta a fókuszt, és egy komoly alkalmazásnál érdemes is megvizsgálni a visszatérési értéket e tekintetben. Viszont a legtöbb helyzetben elegendő, ha csak meghívjuk a függvényt. Végül láthatóvá tesszük az ablakot.

Az alkalmazás elindításával az imént tárgyalt konstruktor fut le. A további két tagfüggvény csak meghatározott események hatására lépnek életbe. Ezen végletekig felhasználóbarát program ablakában a felhasználó a kilépésért küzd. Ha az „Igen” feliratú gomb fölé viszi a kurzort, a gombok látszólag helyet cserélnek. Valójában csak a feliratuk változnak meg, és a vezérlők a helyükön maradnak. Ezt a cselt valósítja meg a `mouseEntered` metódus. Minden, a `MouseListener` által előírt metódus paraméterként egy

`MouseEvent` típusú objektumot kap. Az egérkurzor belépését figyelő függvény törzsének első sorában ennek az eseményobjektumnak a forrását kérdezzük le a `getSource` metódus segítségével. Tudnunk kell ugyanis, hogy melyik gomb fölé került a kurzor, hiszen mindkettőt figyeljük. Ám a `getSource` `Object` típusú referenciát ad vissza, nekünk pedig `JButton`-ra van szükségünk. Ezért később, ahol a `forras` változóra hivatkozunk, egy egyszerű típuskényszerítést hajtunk végre. Ugyanezt megtehettük volna már a lekérdezésnél is, így:

```
JButton forras = (JButton)
➥esemeny.getSource();
```

Ekkor a további típuskényszerítések feleslegesek lennének. Viszont hangsúlyozni szerettem volna a visszatérési érték típusát. A `(JButton) forras` tehát arra a gombra vonatkozik, amelyik kiváltotta az eseményt. Előbb meg kell vizsgálnunk, hogy a gomb felirata *Igen*-e, ellenkező esetben ugyanis nincs semmi dolgunk. Ehhez előbb lekérdezzük a feliratot a nyomógomb `getText` metódusával, majd ennek a `String` objektumnak hívjuk meg az `equals` tagfüggvényét az `igenSzoveg` paraméterrel. Tekintsük az alábbi példát:

```
if ( ((JButton) forras).getText()
➥== igenSzoveg ) { ... }
```

Helyes-e ez ebben a formában? Másképp fogalmazva, ugyanazt fejezi-e ki ez a sor, mint a forráskódban látható? Bár agyunknak az a része, ami a világot egyszerűbbnek szeretné látni, azt mondatja, hogy persze, sajnos nem így van. Az `==` operátor egyenlőséget vizsgál ugyan, de nézzük meg, mi áll a bal- és a jobboldalon. Mindkettő `String` referencia, azaz `String` típusú objektum hivatkozása. Ha két hivatkozást hasonlítunk össze, nagy valószínűséggel annak ellenére kapunk hamis eredményt, hogy az objektumok tartalma azonos. Miután meggyőződtünk arról, hogy az eseményt kiváltó gomb felirata *„Igen”*, hasonló eljárással megnézzük, hogy a baloldali gombról van-e szó. Ha igen, akkor a baloldali feliratát megváltoztatjuk *„Nem”*-re, a jobboldaliét *„Igen”*-re, és a baloldali kapja meg a fókuszt. Hasonló gondolatmenettel a jobboldali nyomógombra vonatkozó ág is értelmezhető.

Végezetül az `actionPerformed` metódus adja a menekülési útvonalat a felhasználónak. Ha sikerül lenyomnia a *„Nem”* feliratú gombot, az alkalmazás bezárul. Ez a metódus a korábbiak alapján magáért beszél. Az egyetlen újdonság a `System.exit(0)` használata. Az `exit` a `System` osztály egy statikus metódusa, melynek paramétere az a visszatérési érték, melyet az alkalmazás a kilépéskor szolgáltat. A függvény meghívásával a program bezárul.

Ebben a rövid programban szegény felhasználó kergeti az egérrel azt a gombot, amelytől azt várja, hogy be tudja zárni az ablakot. Hosszabb-rövidebb idő után rá kell jönnie arra, hogy ebben a csilli-villi ablakozós világban sem lehet elfelejteni a billentyűzet jelentőségét.

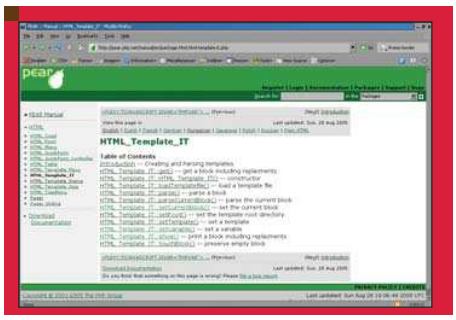
Bár ebben a hónapban még mindig nem teljes mértékben a felhasználó kényeztetése lebegett a szemünk előtt, következő alkalommal már hasznos és látványos animációkat fogunk készíteni, bepillantva a szálkezelés mesterségébe.

Fülöp Balázs
(bigwig42@gmail.com)
21 éves, imádja a Túró Rudit, a Debian Linuxot és a teheneket. Kedvenc írója Slawomir Mrožek. Leginkább a számítógépes hálózatok biztonsága érdekli. A BME VIK műszaki informatikus szak hallgatója.

Sablonkezelés PHP nyelven, PEAR módra

Tovább folytatjuk a PEAR modulokból történő alkalmazásépítést. Legutóbb a DB adatbázis elvonatkoztatási réteget tanulmányoztuk részletesen, ezúttal a nem kevésbé fontos sablonkezelési témát boncolgatjuk, a legnépszerűbb témába vágó PEAR modul, az IT ismertetésével egybekötve.

Bármilyen programozási nyelvről legyen is szó, a kód tisztasága, átláthatósága kulcsfontosságú. Nem csak a továbbfejlesztésről van szó, de az esetleges hibakeresés is rémálommá válhat abban az esetben, ha nincs szerkezete a forráskódunknak. A cél elérésének módja alapvetően az, hogy a fejlesztő rendesen dolgozik, odafigyel, hogy milyen munkát ad ki a kezéből. Nem tud viszont mit kezdeni azzal a ténnyel, hogy számtalan szkriptnyelv, többek között a *PHP* is alapértelmezetten a *HTML* kódba épül be, tehát a kinézet és a kód összekeveredik, ami nem csak logikailag és a működés szempontjából nézve számít szerencsétlen körülménynek. A fenti állapot spagettikóddal összemérhető igénytelenségű kódot eredményez, amelyet a legjobb szándékkal sem lehet tisztán tartani. Az egyetlen megoldás a kód és a kinézet szétválasztása, elkülönítése.



Ez minden esetben valamilyen sablonkezelő rendszert használatával érhető el, amely a kinézetet tartalmazó sablont összeolvasztja az adatokat elkészítő *PHP* kimenettel, és az egészet kiteszi a felhasználónak. Így nem pusztán kódot és kinézetet lehet szétválasztani, de megvalósítható a manapság oly divatos *MVC* architektúra is, ahol a modell a nézet és a vezérlő jól elkülönített hármasából áll össze az alkalmazás.

A sablonkezelést valahogy úgy kell elképzelni, mint az előre megírt papír formanyomtatványokat, ahol ezen a bizonyos „okmányon” szerepelnek azok a részek, amelyek minden esetben ugyanúgy néznek ki, s nekünk az a dolgunk, hogy a változó adatokat kézzel töltsük ki. A programozás területén a formanyomtatvány szerepét a program megjelenési felülete tölti be, ahol szintén az a dolgunk, hogy a fix részek közé a program futása során beírjuk a változó adatokat, magyarul „ki kell töltenünk” a sablont.

Ehhez a sablonban meghatározott alakú címkéket helyezünk el, amelyeket aztán a program segítségével kicserélünk, átalakítunk, vagy épp a sablonban található logika szerint értelmezünk. Minden program esetében igaz az, hogy a futás egy része kizárólag a kinézet előállítására fordítódik. A sablonkezelő rendszerek használata során sincs ez másként, azzal a jelentős különbséggel, hogy elhatárolódik egymástól a működési és a megjelenítési logika. A gyakorlatban a teljes program vagy oldal egyetlen *PHP* címkéből áll, amely sablonkezelő rendszert utasítva írja ki a megjelenítendő *HTML* kódot az alapértelmezett kimenetre.

A sablonok és a PEAR

Mielőtt ismét újra feltalálnánk a kereket, érdemes egy kicsit körülnézni. Azt már tudjuk, hogy a *PEAR* rendszer épp az ilyen általános megoldások kezelésére nyújt kész megoldásokat, kezdjük hát itt a nézelődést.

A sablonkezelőket a *HTML* kategóriában, a *Template* csoportban találjuk. (Megjegyzés: a kategóriában nincsenek valódi csoportok, a sablonkezelőket is csak a nevük kezdete teszi csoporttá.) Jelenleg öt sablonkezelő csomag áll a fejlesztők rendelkezésére, mindegyikük végleges (stable) állapotban. Az összes sablonkezelőt azonban nehéz lenne bemutatni. Általában a végleges állapotúakat szoktam ismertetni, jelen esetben ez azonban nem szűkíti a kört, úgyhogy a másik jellemző módszert választottam: népszerűségük alapján döntöttem, s a statisztikák szerinti legkedveltebb csomagot választottam (sok tízezer ember nem tévedhet). Ez az *Integrated Templates* nevű (teljes nevén *HTML_Templates_IT*) *PEAR* összetevő, amely egy igen egyszerű és jól használható, csere alapú sablonrendszer.

Az IT csomag

Az *IT* csomag egy egyszerű sablonkezelő alkalmazásfejlesztési felület (*API*), amely döntően a benne elhelyezett különleges formátumú címkék értékekre történő kicserélését végzi. Egy az *IT* számára értelmezhető sablon a hagyományos *HTML* címkékből, {címke} alakú ún. helyfenntartókból (placeholder), és az ezeket tagoló blokkokból áll. Amikor a sablont „kitöltjük”, a {címke} alakú helyfenntartókat cserélgetjük ki a programból az általunk meghatáro-

1. lista Az első sablon

A sablon

```
<html>
  <body>
    <!– BEGIN header –>
      Üdvözöllek! Ma {DATE} van.
      <hr>
    <!– END  header –>

    <!– BEGIN footer –>
      <hr>
      Látogató IP-je: {IP}
    <!– END  footer –>
  </body>
</html>
```

A PHP forrás

```
<?php
  require_once "HTML/Template/IT.php";
  $tpl = new HTML_Template_IT("./templates");

  //sablon betöltése
  $tpl->loadTemplatefile("main.tpl.html");

  //aktuális blokk kijelölése
  $tpl->setCurrentBlock("header");
  //címke-érték megfeleltetés
  $tpl->setVariable("DATE",date("Y.m.d"));
  //az aktuális blokk lefordítása
  $tpl->parseCurrentBlock();

  //az oldal megjelenítése
  $tpl->show();
?>
```

zott értékekre. A hatékony működés érdekében ezek a címkék blokkokba vannak szervezve. Ha nem adunk meg ilyet, akkor is az alapértelmezett `__global` blokkban szerepelnek, csak legfeljebb a mindennapos használat során ez nem tűnik fel. Amikor címkecserét hajtunk végre, először kijelöljük, hogy mely blokkban szeretnénk mindezt megtenni, majd a `címke -> érték` hozzárendelés után a sablonkezelő értelmezi és „lefordítja” a kívánságunkat igazi ***HTML*** kódra.
Csak ezután fog az adott blokk megjelenni a kimeneten, ha majd arra utasítást adunk. Egy-egy blokk kitöltése, értelmezése (fordítása) tetszőleges számban következhet egymás után, így ismétlődő részeket (például) táblázatok sorait hozhatunk létre egyedüli sorokból. Ez így elsőre talán nehezen érthető, úgyhogy forduljunk a már jól megszokott módszerhez: lássunk egy példát!

Építsünk bemutató holnapot az IT segítségével!

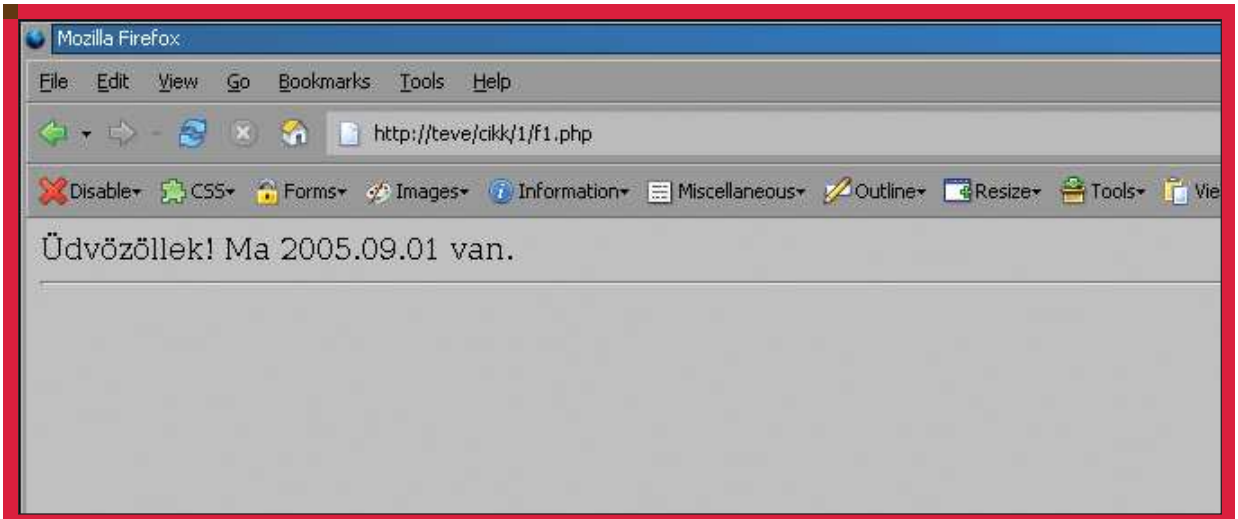
Esetünkben mindjárt példák egész sorával találkozhatunk. A feladat: apróbb lépésenként felépíteni egy nagyon-nagyon egyszerű, Forma-1-es világbajnokok adatainak megjelenítésére szolgáló webalkalmazást, amelynek építése során végigmenyünk az összes gyakori problémán. Előbb azonban telepítsük gépünkre a PEAR környezeten belül az ***IT***-t a

```
pear install HTML_Template_IT
```

parancs kiadásával.

A kezdeti tipegések

Az első lépés egy oldal létrehozása, amelynek van egy fejléce. Ez tartalmazza az aktuális dátumot. A sablonok az aktuális könyvtár ***templates*** alkönyvtárában találhatók.
A kód a sablonkezelő rendszer beemelésével kezdődik, rögtön ezután létre is hozunk egy példányt a sablonkezelőnkből, a továbbiakban ez a változó fogja megtestesíteni a rendszert, amely a paraméterként átadott könyvtárban fogja keresni a sablonokat.



1. ábra Az eredmény

2. lista A kiegészített sablon

A sablon

```
<html>
  <body>
    <!- BEGIN header ->
      Üdvözöllek! Ma {DATUM} van.
      <hr>
    <!- END  header ->
    <!- BEGIN footer ->
      <hr>
      Látogató IP-je: {IP}
    <!- END  footer ->
  </body>
</html>
```

A PHP forrás

```
<?php
  require_once "HTML/Template/IT.php";

  $tpl = new HTML_Template_IT("./templates");

  $tpl->loadTemplatefile("main.tpl.html");

  $tpl->setCurrentBlock("header");
  $tpl->setVariable("DATUM",date("Y.m.d"));
  $tpl->parseCurrentBlock();

  $tpl->setCurrentBlock("footer");
  $tpl->setVariable("IP",$_SERVER
  ➥['REMOTE_ADDR']);
  $tpl->parseCurrentBlock();

  $tpl->show();
?>
```

```
$tpl->loadTemplatefile("main.tpl.html");
```

Betölti a szükséges sablonfájlt, amely tartalmazza a blokkokat és címkéket, amiken később a műveleteket elvégezzük.

```
$tpl->setCurrentBlock("header");
```

Ha már betöltöttük a sablont, rendeljünk értékeket az elhelyezett címkékhez, ám mielőtt ezt megtennénk, ki kell jelölni, hogy melyik blokkban szeretnénk ezt megtenni. Egy változónév egy blokkban is szerepelhet többször, de azok azonos értékekre fognak kicserélődni. Ez a változás azonban nem érinti a más blokkokban elhelyezett azonos nevű változókat.

```
$tpl->setVariable("DATE",date("Y.m.d"));
```

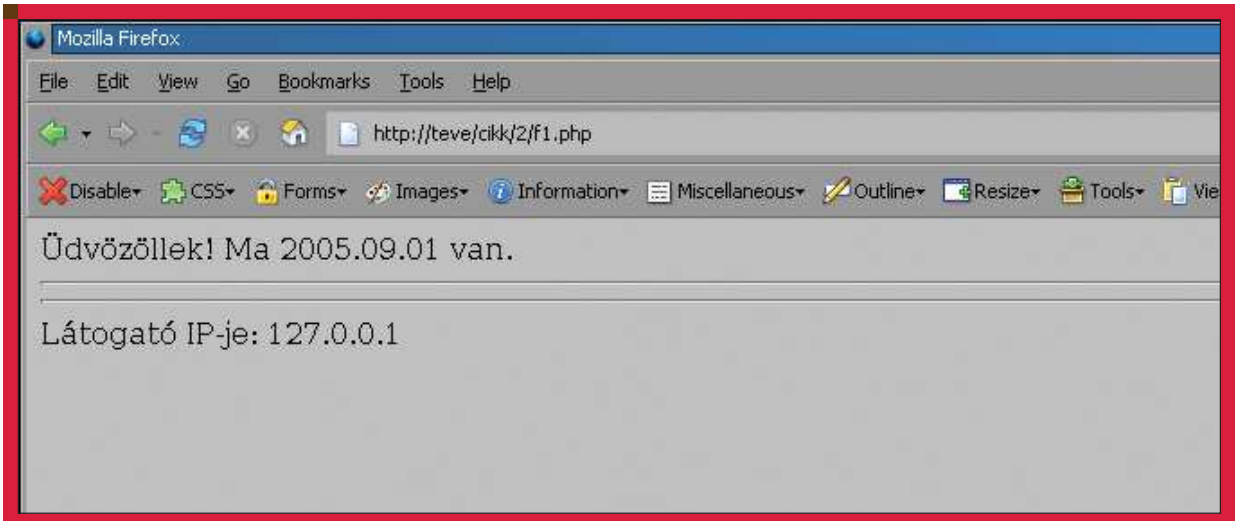
Ez a parancs végzi a konkrét megfeleltetést, és a `DATE` címkéhez a mai dátumot rendeli (természetesen csak a `header` blokkon belül).

```
$tpl->parseCurrentBlock();
```

A megfeleltetés után ezzel a paranccsal tudjuk kicseréltetni a címkéket értékekre, valamint ennek hatására az aktuális blokk bekerül a megjelenítendő tartalmak közé. Ellenkező esetben, ha egy blokkal nem csinálunk semmit, azaz nem rendelünk értékeket a címkékhez, és nem alkalmazzuk ezeket a változtatásokat, akkor az adott blokk tartalma nem fog megjelenni a kimeneten.

```
$tpl->show();
```

Ez a parancs végül kiteszi a már elkészült, kitöltött sablont az alapértelmezett kimenetre. Előfordulhat azonban, hogy nekünk nem erre van szükségünk, nem kiíratni szeretnénk az értéket, hanem a tartalmával műveletet végezni (levélben küldeni, stb.). Ilyen esetekben a `get()` metódust hasz-



2. ábra Az új eredmény

3. lista Ismétlődő elemek egy sablonban

A sablon

```
<html>
  <body>
    <!– BEGIN header –>
      Üdvözöllek! Ma {DATE}. van.
      <hr>
    <!– END  header –>

      <!– BEGIN list –>
        <h4>Versenyzők</h4>
        <a href="f1.php">Nyitólap</a>
        <ul>
        <!– BEGIN list_item –>
          <li><a
href="f1.php?number={NUMBER}">{NAME}</a>
        <!– END list_item –>
        </ul>
      <!– END list –>

    <!– BEGIN footer –>
      <hr>
      Látogató IP-je: {IP}
    <!– END  footer –>
  </body>
</html>
```

A PHP kód

```
<?php
       //az adatforrás
  $drivers[1] =
array('firstname'=>"Michael",'lastname'=>
➥"Schumacher",'number'=>1,'age'=>35,
➥'num_of_championships'=>7);
  $drivers[2] = array('firstname'=>"Mika",
➥'lastname'=>"Hakkinen",'number'=>
➥2,'age'=>33,'num_of_championships'=>2);
  $drivers[3] = array('firstname'=>"Jacques",
➥'lastname'=>"Villeneuve",'number'=>
➥3,'age'=>32,'num_of_championships'=>1);
  $drivers[4] = array('firstname'=>"Damon",
➥'lastname'=>"Hill",'number'=>4,
➥'age'=>37,'num_of_championships'=>1);

  require_once "HTML/Template/IT.php";
  $tpl = new HTML_Template_IT("./templates");
  $tpl->loadTemplatefile("main.tpl.html");

  $tpl->setCurrentBlock("header");
  $tpl->setVariable("DATE",date("Y.m.d"));
  $tpl->parseCurrentBlock();

  $tpl->setCurrentBlock("list_item");
  foreach ($drivers as $driver) {
    $tpl->setVariable("NUMBER",$driver
    ➥['number']);
    $tpl->setVariable("NAME",$driver
    ➥['firstname'][0].". ".$driver
    ➥['lastname']);
    $tpl->parseCurrentBlock();
  }
  $tpl->parse("list");

  $tpl->setCurrentBlock("footer");
  $tpl-
>setVariable("IP",$_SERVER['REMOTE_ADDR']);
  $tpl->parseCurrentBlock();

  $tpl->show();
?>
```

3. ábra Az eredmény

nálhatjuk, amely egy karaktersorozat formájában visszatérési értékéül adja a kitöltött sablont.

Még mindig tipegünk

A blokkok kitöltögetését természetesen egymás után végezhetjük, a dolgunk annyi, hogy meg kell mondani a megjelenítés előtt, hogy melyik blokk legyen a következő, amelyen műveleteket szeretnénk végezni. Egészítsük ki a fenti példát egy lábléccel, amely a látogató *IP*-jét mutatja!

Ismétlődő elemek, egymásba ágyazva

Ugorjunk most egy kicsit nagyobbat. Készítsünk egy adatforrásból egy listát a már meglévő lapon, amely a későbbiekben megjelenítendő világbajnokok neveit tartalmazza, amelyek mindegyi-

4. lista Blokkok

A sablon

```
<html>
  <body>
    <!- BEGIN header ->
      Üdvözöllek! Ma {DATE}. van.
      <hr>
    <!- END  header ->

    <table>
      <tr valign="top">
      <td>
      <!- BEGIN list ->
        <h4>Versenyzők</h4>
        <a href="f1.php">Nyitólap</a>
        <ul>
        <!- BEGIN list_item ->
          <li><a
href="f1.php?number={NUMBER}">{NAME}</a>
        <!- END list_item ->
        </ul>
      <!- END list ->
      </td>
      <td>

      <h4>Részletes adatok</h4>
      <!- BEGIN havetoselect ->
        Kérem, válasszon a felsorolt nevek közül.
      <!- END havetoselect ->

      <!- BEGIN detail ->
        Sorszám: {NUMBER}.<br>
        Vezetéknév: {LASTNAME}<br>
        Keresztnév: {FIRSTNAME}<br>
        Életkor: {AGE} év<br>
        Világbajnoki címek: {NUM_OF_CHAMPIONSHIPS}
        ➥db<br>
      <!- END detail ->
      </td>
      </tr>
    </table>

    <!- BEGIN footer ->
      <hr>
      Látogató IP-je: {IP}
    <!- END  footer ->
  </body>
</html>
```

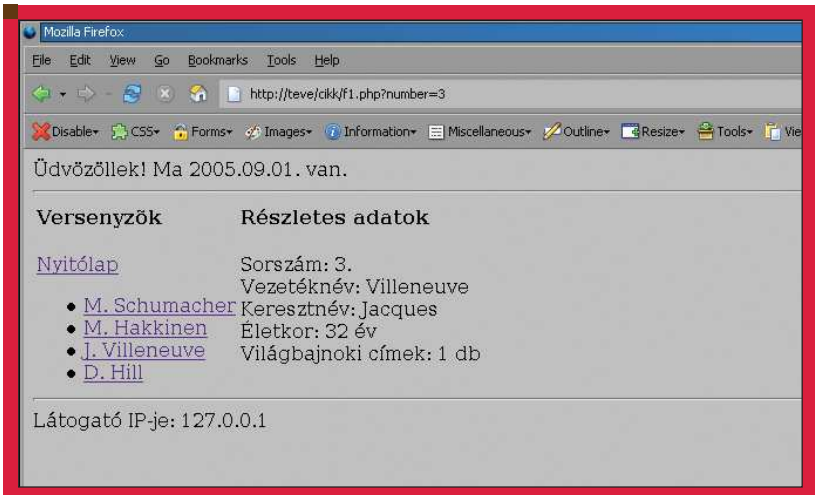
A PHP kód (az adatforrás nélkül)

```
<?php
        //az adatforrás: lást előző példa
 require_once "HTML/Template/IT.php";
  $tpl = new HTML_Template_IT("./templates");
  $tpl->loadTemplatefile("main.tpl.html");

  $tpl->setCurrentBlock("header");
  $tpl->setVariable("DATE",date("Y.m.d"));
  $tpl->parseCurrentBlock();

  $tpl->setCurrentBlock("list_item");
  foreach ($drivers as $driver) {
    $tpl->setVariable("NUMBER",$driver['number']);
    $tpl->setVariable("NAME",$driver['firstname']
    ➥[0].". ".$driver['lastname']);
    $tpl->parseCurrentBlock();
  }
  $tpl->parse("list");

  if (!isset($_GET[number])) {
     $tpl->touchBlock("havetoselect");
  } else {
    $tpl->setCurrentBlock("detail");
    $tpl->setVariable("FIRSTNAME",$drivers
    ➥[$_GET[number]]['firstname']);
    $tpl->setVariable("LASTNAME",$drivers[$_GET
    ➥[number]]['lastname']);
    $tpl->setVariable("NUMBER",$drivers
    ➥[$_GET[number]]['number']);
    $tpl->setVariable("AGE",$drivers
    ➥[$_GET[number]]['age']);
    $tpl->setVariable("NUM_OF_CHAMPIONSHIPS",
    ➥$drivers[$_GET[number]]
    ➥['num_of_championships']);

    $tpl->parseCurrentBlock();
  }

  $tpl->setCurrentBlock("footer");
  $tpl->setVariable("IP",$_SERVER['REMOTE_ADDR']);
  $tpl->parseCurrentBlock();

  $tpl->show();
?>
```

ke egy-egy hivatkozás, ami ***GET*** paraméterként tartalmazza a versenyzők sorszámát, valamint egy fix hivatkozást, amely az alapállapotba (nyitólapra) mutat.

A sablonban két egymásba ágyazott blokkot találunk. Általánosan igaz az, hogy mindig a legbelső blokk kitöltésével kell kezdenünk. Ez a fenti példában a listaelem (`list_item`) blokk. Már említettem, hogy egy-egy blokkot tetszőlegesen sokszor lefordíthatunk, és hozzárendelhetünk a kimenethez, így hozva létre ismétlődő ***HTML*** kódrészleteket. Jelen esetben egy ciklussal végigmentünk az adatforráson, miután kijelöltük a blokkot, majd a címke -> érték hozzárendelés után „hozzáfűztük” a lefordított sablonhoz, majd ezt ismételgettük ugyanezzel a blokkal addig, amíg el nem értük a kívánt számú listaelemet.

A listaelemek felszaporítása után „megparancsoltuk” a külső lista blokknak, hogy rendelje hozzá önmagát



4. ábra Így néz ki az elkészült példafeladat

a belső blokkal együtt a kitöltött kimenethez. Ha ez a külső blokk egyéb behelyettesítendő címkét is tartalmazott volna, akkor először meg kellett volna csinálni a hozzárendeléseket, majd utána utasítani a sablonkezelőt, hogy az immáron kész blokkot fűzze hozzá a végleges állapotúakhoz. Hasonlóan kell eljárni a tetszőleges számú egymásba ágyazott blokkok esetén is, folyamatosan belülről kifelé haladva.

Blokkok: akarom, nem akarom?

Egészítsük ki a példánkat a versenyzők részletes adatainak megjelenésével. Ha egy névre kattint a felhasználó, akkor a paraméterként kapott sorszám alapján a versenyzők listája mellet, jobbra jelenjen meg a pilóta összes adata. Ha még nincs kiválasztva egyetlen versenyző sem, akkor a részletes adatok helyett egy figyelmeztetés jelenjen meg, amelyben felszólítjuk a látogatót, hogy jelöljön meg egy nevet.
Már említettem, hogy alapértelmezetten azok a blokkok, amelyekkel nem végzünk műveleteket, nem is jelennek meg a kimenetben. Jelen esetben ezt a tulajdonságot használtuk ki, amikor a paraméter meglététől függően vagy az egyik, vagy a másik blokkot töltöttük ki. Illetve a dolog nem ilyen egyszerű, ugyanis a figyelmeztető blokk nem tartalmazott címkét, csak állandó szöveget, így azzal nem is tudunk műveleteket végezni. Ennek kezelésére való a `touchBlock()` metódus, amelynek hatására a megadott blokk változatlan formában bekerül a kész blokkok közé.
Mivel a fenti példában a sablon kezd elbonyolódni, célszerű lenne az egyes részeket külön fájlba tenni – már csak azért is, mivel a fejléc és a lábléc több oldalon is szerepelhet. Az *IT* az ilyen esetek kezelésében is jártas. Ha például a blokkok mentén vágnánk a fájlt külön darabokra, a fenti példában a kódot csak annyiban kellene módosítani, hogy a blokkok kijelölése előtt (`setCurrentBlock()`) be kellene tölteni még az adott nevű sablont (`loadTemplateFile`) is, majd ha a sablon összes címkéjét kicseréltük, a kimenetre írjuk azt, vagy a már emlegetett `get()` metódus segítségével egy változóban összefűzzük.

Végszó az IT és a sablonkezelés ügyében: egyszerűség!

Az előző bekezdés alapján már látszik, hogy a több fájlos megoldás (tortaszeletelő módszer) és ama bizonyos `get()` tagfüggvény segítségével, komoly, modulos rendszerű, komplett webes rendszerek készíthetők, úgy hogy mindvégig megőrizhetjük a módszer nemes egyszerűségét.
Ha jobban belegondolunk, ez a rendszer nem sok mindent tud: kicserél bizonyos címkéket bizonyos értékekre, amit már az `str_replace()` függvénnyel is megtehetünk. „Csupán” annyival fűszerezi a dolgot, hogy egy szemtelenül egyszerű utasításkészlettel burkolja a cserélgetést, illetve a blokkok alkalmazásával eléri, hogy kellően áttekinthető sablonokat készíthessünk. Ahhoz, hogy megértsük, mit csinál, elég elolvasni egy ehhez hasonló cikket. Ahhoz, hogy hatékonyan tudjunk vele dolgozni, nem kell további tetemes gyakorlati ismeretet szereznünk, egy saját magunk által kitalált kis példaalkalmazást megvalósítva könnyen ráérezhetünk az ízére.
Az elkészült kód egyszerű, áttekinthető, logikus és mások által is érthető – azt is mondhatnánk: közkincs. Az ilyen egyszerű megoldásokban rejlik a *PEAR* ereje!
Az *IT*-n kívül természetesen ott a többi sablonkezelő rendszer is, ám ennek a cikknek és a cikksorozatnak nem az a célja, hogy referenciát adjon a programfejlesztéshez. Sokkal inkább cél az, hogy bemutassa a *PEAR* rendszer erejét egy-egy jellegzetes témakörön keresztül. Aki a maradék *PEAR* sablonkezelőkről is szeretne részletesen tájékozódni, az alábbi oldalakon teheti ezt meg.

Komáromi Zoltán
(komi@kiskapu.hu)
25 éves, a BME hallgatója, mellette PHP-programozóként dolgozik. Kedvenc területe a multimédia.

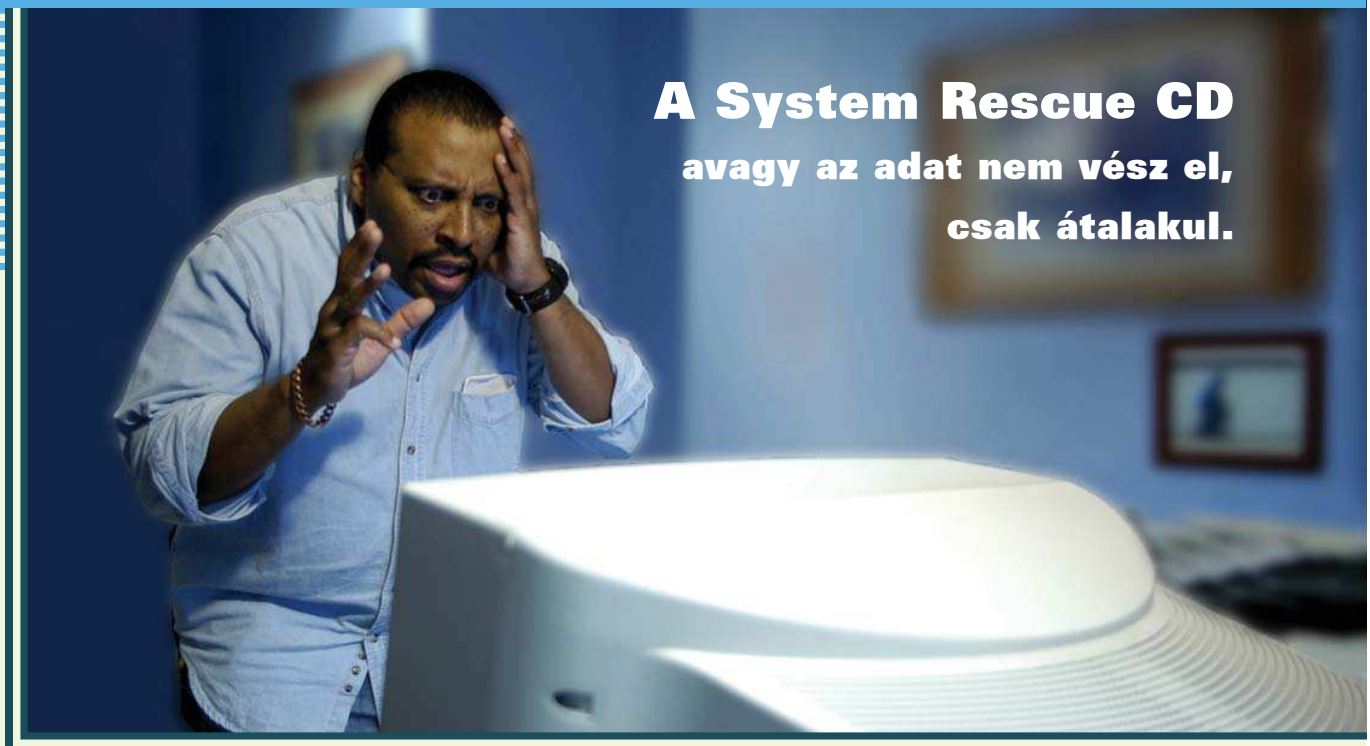
KAPCSOLÓDÓ CÍMEK

Flexy:
➲ http://pear.php.net/package/HTML_Template_Flexy

PHPLIB:
➲ http://pear.php.net/package/HTML_Template_PHPLIB

Sigma:
➲ http://pear.php.net/package/HTML_Template_Sigma

Xipe:
➲ http://pear.php.net/package/HTML_Template_Xipe



A System Rescue CD

avagy az adat nem vész el, csak átalakul.

Hétfő, reggel fél kilenc.
„Ó, hogy az a...” – kiáltott a tehetetlenségtől dühös orángutánná változott felhasználó.
„System Rescue CD, máris hozom.” – válaszolt nyugodtan a rendszergazda a szomszéd szobából.

Ha nem szabad szoftverről lenne szó, talán valamelyik kereskedelmi csatornán két mosópor reklámja között találkozhatnánk a fenti párbeszéddel... Ez a cikk annak az életérzésnek az elsajátításáról fog szólni, amikor csak berakom a lemezt és már mentek is.
A ***System Rescue CD***-vel a ***Linuxvilág*** 2004. szeptemberi számához mellékelt lemez képében ismerkedtem meg. Első pillantásra egy átlagos „digitális svájci bicskának” tűnt, ám amikor elolvastam a felhasználói kézikönyvet, és megláttam, milyen kiváló segédeszközök kaptak helyet ezen az életmentő korongon, be kellett látnom, hogy itt többről van szó. Remélem, hogy az alábbi, képernyőfotókkal tűzdelt írás a kedves Olvasót is meggyőzi erről, és a szoftver használatával értékes időt és energiát takaríthat meg.

Beszerzés

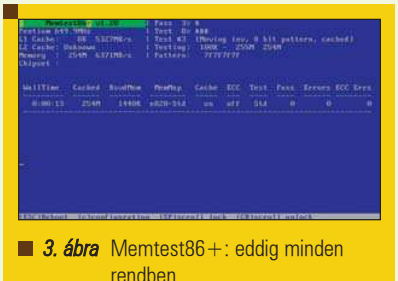
Ez a terjesztés is, mint megannyi más a ***GPL 2 (GNU General Public Licence)*** licensz hatálya alá tartozik, így ingyenesen elérhető bárki számára.
Mi több, magunk is bővíthetjük ízlésünknek megfelelően. A körülbelül 110 megabájt méretű képfájl a hivatalos honlapról indulva, a ➲ ***http://www.sysresccd.org/download.en.php*** oldalról érhető el. Jelenleg a 0.2.15-ös változat a legfrissebb stabil kiadás. Amint az a ***UNIX*** világában természetes, nem az 1.0 jelenti a használhatóságot, hanem a megbízható fejlesztők.
Érdemes egy kicsit elidőzni a projekt honlapján, ahol a ***Gyakran Ismételt Kérdések*** mellett letölthető egy ***PDF*** formátumú leírás is. Ez a közel 50 oldalas, angol nyelvű ismertető kellően részletes az elinduláshoz, de nem helyettesíti az egyes programok saját, részletes dokumentációját. Ellenben jól olvasható és keresni is könnyű benne. Nekem mindössze az volt furcsa, hogy a ***„connection”*** (kapcsolat) következetesen ***„connexion”*** alakban szerepel. Ezt először az információs szupersztrádán született szubkultúrák egyikének törzsi szokásaként tudtam be. Csak később vettem észre, hogy a ***CD*** mellett az útmutatót is francia anyanyelvű szakemberek állították össze.

Rendszerindítás

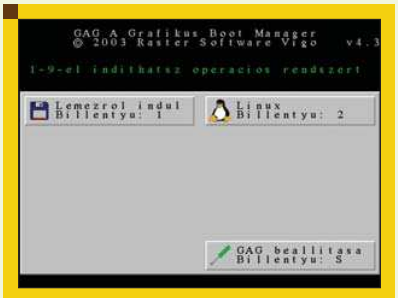
Miután ***CD***-re vésték a képfájlt, biztosítanunk kell, hogy a számítógép az optikai meghajtóról végezze el a betöltést. Ez alaplapunk leírását követve a ***BIOS***-ból könnyedén elvégezhető. Ha véletlenül elfelejtettük volna az ehhez szükséges jelszót, a megfelelő alaplapi kapcsoló, vagy jumper segítségével alapértelmezett értékekkel tölthetjük fel a ***BIOS***-t. Ha ezt nem találjuk, még mindig nem kell kétségbe esnünk, hiszen a ***CMOS*** memória tápellátását biztosító gombelem eltávolításával rövid idő után annak tartalma elvész.
Ha sikerrel vettük az első akadályt, egyszerűen helyezzük be a lemezt a meghajtóba és indítsuk újra a számítógépet. Ezt követően az *1. ábrán* látható alábbi kép fogad bennünket. Mint az szakértő szemeknek bizonyára azonnal feltűnik, egy ***LILO (Linux Loader)*** indítóképernyővel van dolgunk. Itt az ***Enter*** billentyűt használva azonnal megkezdődhet a betöltés, ám mielőtt továbbmennénk, nézzük meg, milyen lehetőségeink vannak ezen kívül. Az ***F2***, ***F3*** és ***F4*** billentyűkkel súgóoldalakat jeleníthetünk meg a betölthető rendszerekkel kapcsolatban. Nem tévedés, a ***System Rescue CD*** nem pusztán egy ***Gentoo Linux*** alapú ***LiveCD***, emellett még további meglepetéseket is tartalmaz.

■ *1. ábra* Az indítóképernyő

■ *2. ábra* A FreeDOS betöltése



■ *3. ábra* Memtest86+: eddig minden rendben

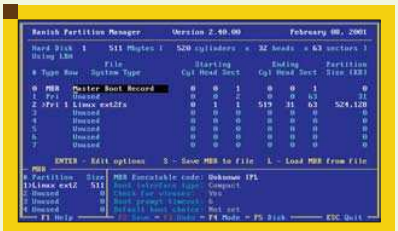


■ *4. ábra* A magyarul beszélő GAG

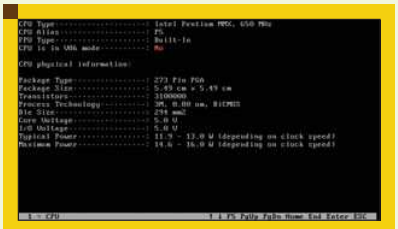
Ha egy *CD*-ről futó *Linuxra* van szükségünk, itt kiválaszthatjuk a képernyő felbontását. Az `fb640` *640x480*-as, az *fb800 800x600*-as, az `fb1024` rendszermag kép pedig *1024x768*-as *FrameBuffer*-es képernyőt biztosít.
A *FrameBuffer* röviden arra szolgál, hogy grafikus kijelzőn dolgozhassunk *X* kiszolgáló használata nélkül. Ezért, ha szeretnénk futtatni a grafikus alkalmazások egyikét, például a *QtParted* nevű partícionáló segédprogramot, feltétlenül ezen három kép valamelyikét kell választanunk úgy, hogy a nevét beírjuk a betöltő parancssorába.
Az alapértelmezett az `fb640`, vagyis az *Enter* gomb megnyomásával ennek a betöltése kezdődik meg. A legtöbb mai számítógép videovezérlője és monitora alkalmas az *1024x768*-as módra, ezért érdemes némi gépelés árán nagyobb felbontásban dolgozni, különösen, ha a grafikus böngészőprogramot is szeretnénk használni. Megjegyzem, a PC megjelenése körüli időkből származó számítógépek használóira is gondoltak. A `nofb` hatására *FrameBuffer* nélkül indul a *Linux*.
A számunkra megfelelő rendszermag kiválasztásával még közel sem zárultak be lehetőségeink. Ezt követően nyílik mód a rendszermag paramétereinek meghatározására. A `nokeymap` hatására a rendszer betöltés közben nem kérdez rá a billentyűzetkiosztásra, hanem az alapértelmezett amerikai angolt használja. A `cdcache` segítségével a *CD* teljes tartalma a memóriába töltődik, így a lemez lecsatolásával betehetünk egy másikat a helyére.
A `root` kapcsolóval megadhatunk egy meglévő *Linux* rendszerpartíciót, így például sérült *MBR (Master Boot Record)* esetén használhatóvá tehetjük rendszerünket. Végül a `nonet` az önműködő hálózati csatoló felismerést kapcsolja ki.
Még számos kapcsoló van, melyekről itt nem szóltam, mivel megpróbáltam a legfontosabbakat összegyűjteni. További segítséget az *F* billentyűkkel elérhető súgólapokon talál az Olvasó. Ezeket a kapcsolókat szóközzel elválasztva a rendszermag kép neve után lehet megadni. Fontos, hogy ha az alapértelmezett magot használjuk, de szeretnénk megadni egy kapcsolót, akkor nem hagyhatjuk el az fb640 beírását. Én az alábbi sorral indítottam a rendszert:

```
boot: fb1024 nonet
```

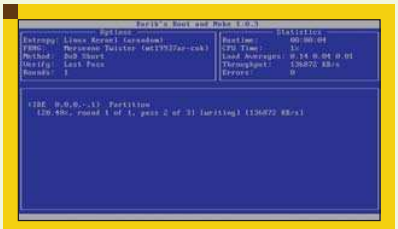
Mielőtt viszont rátérnénk az ennek hatására elinduló Linux tárgyalására, vessünk egy pillantást azokra a meglepetésekre, amelyeket korábban említettem. Ha a `freedos` szót adjuk meg a parancssorban, az alábbi képernyőhöz jutunk.
Itt egy *Entert* ütve a *FreeDOS* nevű operációs rendszerhez jutunk, ami teljesen *MS-DOS* megfelelő, és emellett *GPL* licenszű. Fejlesztése folyamatos, kezeli a *CD*-olvasót és az egeret is. Ha *DOS* programot szeretnénk futtatni, jobbat keresve sem találhatnánk. Egy dir /p parancsot kiadva a segédprogramok között régi ismerősök köszönnek vissza, mint a `CHKDSK`, az `FDISK`, vagy a `FORMAT`. Az „újdonságok" között pedig ott van az 1998-as `FIPS`, amellyel a merevlemez használatban lévő *DOS* partíciói méretezhetők át adatvesztés nélkül.
Ha a *CD* betöltőjének a `memtest` szót adjuk meg, a *Memtest86+ 1.20*-as változata indul el. Ez egy valóban megbízható eredményt adó memória tesztelő program. Ugyanis legyen szó bármilyen operációs rendszerről, az ezek alatt futtatott tesztprogramok a memória egyes részeihez nem férnek hozzá, például mert ott található a rendszermag. További gond a korszerű rendszerek nagyfokú hibatűrése, ami elrejti a hibákat a programok elől. Ezért ha valós képet szeretnénk kapni, használjuk a következő képen is látott *Memtest86+*-t, mivel ez nem igényel futtatókörnyezetet. Készüljünk fel viszont arra is, hogy ha sok a fizikai memória a gépünkben, a tesztsorozat igen-igen sokáig eltarthat.
A `gag` szót megadva a *LILO* parancssorában a *GAG* nevű grafikus rendszerbetöltő 4.3-as változatát érhetjük el. Ezt telepíthetjük floppyra, vagy a merevlemez *MBR* területére is. Legfeljebb 9 rendszer betöltését teszi lehetővé, képes logikai partícióról tölteni, tud magyarul is és kifejezetten egyszerű a beállítása. A képernyőfotón látható *GAG* menüje már tartalmaz egy linuxos partíciót.
Amennyiben indításkor a `ranish` szót adjuk meg, a *Ranish Partition Manager 2.40.00*-ás változata tárul elénk. Ez egy karakteres menüvezérelt felülettel bíró teljes értékű partícionáló program, melyhez nem kell végigvárnunk egy *Linux* betöltést. Elsőre talán nehezen átlátható, de a beépített súgó és a partícionálást segítő varázsló jelentősen megkönnyíti a használatát. A következő képen működés közben látható az RPM.
Ha az `aida` szót írjuk be az indítóképernyőn, az *AIDA16* rendszerelemző szoftver 2.12-es változatához jutunk. Ez egy *DOS* alapú, magyar fejlesztésű program, mellyel átfogó és részletes képet kaphatunk arról, hogy milyen hardver-



5. ábra A Ranish Partition Manager



6. ábra Processzorinformációk AIDA16-ban



7. ábra dban: jól jöhet, amikor eladjuk a merevlemezünket

8. ábra Néhány Enter, és eltűnik az adminisztrátori jelszó

összetevők találhatóak számítógépünkben. Sajnos az *AIDA16* fejlesztése végleg befejeződött, ugyanis a fejlesztő, *Miklós Tamás* a *Lavalys* állásajánlatának elfogadásával egyidejűleg bezárta a projekt honlapját. Jelenleg az említett cég oldaláról egy *Everest* nevű diagnosztikai program otthoni változata tölthető le ingyenesen – viszont kizárólag *Windows* rendszerre. Az alábbi fotó a *FreeDOS* alatt futó *AIDA16*-ot mutatja. A betöltő parancssorában a `dban` szó megadásával a *Linux* alapú *Darik's Boot and Nuke 1.0.3* fogad bennünket, amelynek már távolról sincs sok köze a rendszer-helyreállításhoz. Ezzel a szoftverrel végérvényesen és visszaállíthatatlanul törölhetők teljes merevlemezek, illetve partíciók. Ahhoz, hogy a hatás a névhez méltó legyen (a *nuke* angol ige nukleáris csapás mérését fejezi ki), a kiválasztott módszertől függően adott bitmintával írja felül az eszköz összes bájtját. Ezt tetszőlegesen állítható számú menetben teszi meg, és a végén kérésre ellenőrzi tevékenységét. Talán elnézi nekem a kedves Olvasó, hogy nem találok kellően nyomatékos szavakat annak hangsúlyozására, mekkora károkat lehet okozni ezzel. Kérem, hogy nagy körültekintéssel használja a programot, mert *visszavonhatatlan adatvesztést* okozhat, amiért a program készítője sem vállal felelősséget.

A 0.2.15-ös *System Rescue CD* egyik újdonsága a korábbi változatokhoz képest az `ntpass` rendszerkép megjelenése. Ezt beírva a parancssorba egy szintén *Linux* alapú *Windows NT* jelszó és regisztrációs adatbázis szerkesztőhöz jutunk. Segítségével a régi jelszó ismerete nélkül változtatható vagy törölhető minden felhasználói fiók jelszava, beleértve az adminisztrátorét is. A zárolt fiókok zárolásának megszüntetésére is van lehetőség. A regisztrációs adatbázis módosítása, új kulcsok és értékek felvétele úgyszintén támogatott. Így nem kell megijednünk, ha véletlenül elfelejtettük saját gépünk adminisztrátorának jelszavát.

A LiveCD használata

Térjünk vissza a *System Rescue CD*-ről futtatható *Linux* rendszerhez. A *LILO* parancssorába, mint azt korábban említettem, az alábbit írtam be:

```
boot: fb1024 nonet
```

Ezzel *1024x768*-as *FrameBuffer* módban indul egy 2.4.27-es rendszermag. Egyelőre a 2.6-os sorozatú magok nem támogatottak egyes, a *CD*-n található segédprogramra való tekintettel. A hálózati kártya felismerését kikapcsoltam, mivel tudtam a használandó modul nevét, és a felismerés eltarthat egy darabig. Rövid töltögetés után a használandó billentyűzetkiosztásra vonatkozó kérdés jelenik meg, amit a következő fotón láthatunk. Sokunk örömére magyar kiosztás is választható. Ezt követően kisvártatva egy rövid ismertető és egy rendszergazdai jogokkal futó parancshéj fogad bennünket. Alapértelmezésben a root felhasználó jelszava üres, de még bejelentkezni sem kell, mind a 6 terminálon egy-egy héjat találunk. Ezek között a megszokott *Alt+F1*, *Alt+F2*, stb. billentyűkombinációkkal válthatunk. A *Midnight Commander* az `mc` paranccsal azonnal indítható. Én elsőként a hálózati kártyám modulját töltöttem be. Ezután következhet az *IP*-cím és az átjáró beállítása. Ha nincs a hálózaton *DHCP* kiszolgáló, ez az alábbi parancsokkal történhet:

```
% ifconfig eth0 <saját_ip>
% route add default gw
➥<átjáró_ip>
```

A névkiszolgálókat a */etc/resolv.conf* fájlban a `nameserver` kulcsszóval lehet megadni. Ezekre nekem nem volt szükségem, ugyanis a háttérben futó *DHCP* ügyfél démon önműködően megtett minden hálózati beállítást. A `ping` parancs segítségével ezt rögtön ellenőriztem is, majd a `-g` kapcsolóval indított `links` böngészőprogrammal meglátogattam a magyar *Linuxvilág* magazin honlapját. Ezt mutatják be a következő képek.

A *CD*-n található egy teljes *SSH* csomag, ami távoli bejelentkezést, illetve állományok mozgatását teszi lehetővé számítógépek között. Ezen kívül a *System Rescue CD*-t futtató gépen *SSH* kiszolgálót is indíthatunk, ez esetben viszont már erősen ajánlott beállítani valamilyen jelszót a root felhasználónak a `passwd` paranccsal. A további hálózati lehetőségek között szerepel az SMB fájlrendszer rendszermag szintű támogatása. Így *windowsos* gépek megosztásai csatolhatók fel a `mount` parancs segítségével. A `lufsmount` használatával *FTP* és *SSH* megosztások is befűzhetők egy könyvtár alá, és úgy kezelhetők, mintha helyi állományokról és könyvtárakról lenne szó.

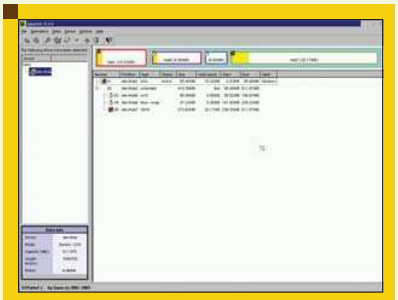
Fontosabb rendszereszközök

A *Partition Magic* szerelmeseinek két ingyenesen használható grafikus partícionáló segédprogramot kínál a *CD*. A *PartGui 0.2.3* és a *QtParted 0.4.4* egyaránt a *Qt* eszközkészletre épít. Mindkettő könnyen kezelhető, de a *PartGui* fejlesztése leállt, ezért érdemes a *QtParted* használatában

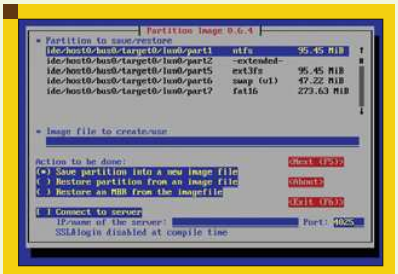
9. ábra Válasszunk billentyűzetkiosztást

10. ábra A hálózat beállítása

11. ábra www.linuxvilag.hu a grafikus links-ben



12. ábra A QtParted és egy 500 MB-os lemez



13. ábra Partimage: mentsük a menthetőt

elmélyedni. Ez a `run_qtparted` paranccsal indítható. Elsőként megkérdezi, hogy milyen kapura csatlakoztattuk az egeret. A felsorolásban a soros, PS/2 és USB kapu egyaránt szerepel, de az önműködő felismertetést is választhatjuk. Ezután a *QtParted* a képen látható felületen nyújt segítséget a partícionálásban és a formázásban.
Téved, aki ezek után azt gondolja, hogy a parancssorból használható partícionáló programok feleslegesek. Az `sfdisk` nevű segédeszközzel ugyanis biztonsági mentést készíthetünk partíciós táblánkról, amit aztán floppy lemezre, vagy egy hálózati megosztásra menthetünk. Azon a merevlemezen, amelynek a táblájáról készült a mentés, nyilván nincs sok értelme azt tárolni, mert ha megtörtént a baj, akkor épp az ezen levő adatokat nem fogjuk tudni elolvasni.
Ez leginkább ahhoz a leleményhez hasonlít, amikor valaki a széf kulcsának a másolatát a széfbe zárja. A következő paranccsal készíthetünk mentést:

```
% sfdisk -d <eszköz_neve> >
➥<mentés>
```

Az így készített állomány szöveges formátumú, így egy egyszerű szövegszerkesztővel is belenézhetünk. A visszaállítás hasonlóan könnyen végezhető:

```
% sfdisk <eszköz_neve> <
➥<mentés>
```

Ezzel viszont vigyázzunk, mert a parancs átírja a meglévő partíciós táblát. Kellő körültekintés mellett is mindig érdemes az első paranccsal menteni a táblát, mielőtt felülírnánk.

A partíciós tábla védelme mellett a partíciók tartalmának mentéséről is gondoskodhatunk a *Partimage 0.6.4* segítségével. Ez egy kliens-szerver alkalmazás, amelyben a kiszolgáló felelőssége a lemezképek tárolása, az ügyfelek pedig ide mentenek, és innen állítják vissza a partíciókat. Két változata érhető el a CD-ről. A sima szöveges átvitellel dolgozó kliens a `partimage`, a szerver a `partimaged` paranccsal indítható. Az *SSL* támogatással bíró, titkosított csatornát használó változat ügyfélprogramja a `partimagessl`, kiszolgálója pedig a `partimagedssl` nevet viseli. A *Partimage* felületét mutatja be a következő fotó.
A *Partimage* leginkább a jól ismert `dd`-hez hasonlít, mellyel már régóta készíthetünk nyers másolatot egy-egy partícióról, esetleg egy teljes lemezről. A legfontosabb különbség a hálózati működés mellett az, hogy míg a `dd` teljes másolatot készít, a *Partimage* felismeri a fájlrendszer típusát, és ügyel arra, hogy az üres területek ne kerüljenek bele a lemezképbe. A legtöbb fájlrendszert támogatja, sajnos azonban az *NTFS* partíciók kezelése még eléggé kiforratlan. További információért és a támogatott fájlrendszerek teljes listájáért érdemes ellátogatni a projekt honlapjára (➲ *http://www.partimage.org/*).
Sokan a mai napig a parancssoros `tar`-ral végeznek biztonsági mentést a rendszerről. Ennek az archiváló segédprogramnak a továbbfejlesztett változata a *DAR (Disk Archiver)*, amely szintén helyet kapott a *CD*-n. Az archívum elkészítésekor elődjéhez hasonlóan nem tesz mást, mint egyetlen állományba írja ki a mentésre kijelölt fájlhalmazt, ami már tetszőleges módszerrel tömöríthető. Ellenben a visszaállításnál már sokkal okosabb, mert képes egyetlen állományt is kibontani anélkül, hogy a teljes archívumot kicsomagolná. Hasonlóan jól kezeli a szeletelt mentéseket is.
Bármennyire is vigyázunk, akármikor áldozatai lehetünk egy kellemetlen vírusfertőzésnek, ami adott esetben a rendszer elindítását is lehetetlenné teheti. Ez esetben használhatjuk az ingyenes *Clam AntiVirust*, amelynek vírusleíró adatbázisa az Internetről frissíthető. Ez egyszerűen az alábbi paranccsal megtehető:

```
% freshclam
```

Ezután egy könyvtár annak összes alkönyvtárával együtt rekurzívan a következő paranccsal ellenőrizhető:

```
% clamscan -r <könyvtár>
```

Az Olvasóban bizonyára felmerül a kérdés, hogy egy *Windows* partíción mennyire használható ez a program. A *Clam AntiVirus* felismeri a *Windows*-ra írt vírusokat, viszont azok eltávolításához írási jogra van szükség, ami egy *NTFS* partíció esetében *Linux* alól veszélyes lehet. Sajnos a *Linux* rendszermag nem nyújt kiforrott támogatás az *NTFS* partíciók írásához. Ne essünk kétségbe, van megoldás. Igaz ugyan, egy kicsit körül kell táncolni a tüzet az esőhöz. A *Captive-NTFS* projekt nyújtotta eszközökkel teljes körű írás-olvasás valósítható meg *NTFS* formátumú fájlrendszereken is. Ez a megoldás a *Windows* saját meghajtóit használja a műveletek végrehajtásához. Bár az összes *NT* alapú operációs rendszer által kezelt partíció támogatott, a szükséges állományokat csak egy *Windows XP*-ből tudjuk megszerezni. A fájlok nincsenek rajta a *System Rescue CD*-n, mivel nem a *GPL licensz* alá tartoznak, ezért ezt a módszert akkor tudjuk alkalmazni, ha van egy legálisan telepített *Windows XP*-nk.
Első lépésként szerezzük be a meghajtókat. Két fájlra lesz szükségünk, amelyek az *XP* alapértelmezett telepítése mellett az alábbi elérési úton érhetőek el:

```
C:\WINDOWS\System32\drivers
➥\ntfs.sys
C:\WINDOWS\System32\ntoskrnl.exe
```

Mindkét állományt másoljuk be *Linux* alatt a */var/lib/captive* könyvtárba. Ezután a következő egyszerű paranccsal csatolhatjuk fel *NTFS* partíciónkat:

```
% mount.captive-ntfs
➥<eszközfájl> <könyvtár>
```

Egy helyi *XP* esetén megtehetjük, hogy először a hivatalos rendszermagban található *NTFS* modult használva csak olvasható módban csatoljuk fel a partíciót, majd a szükséges *Windows* meghajtók beszerzése után lecsatoljuk és a *Captive-NTFS* segítségével már írható-olvasható módban fűzzük be újra. Egy dologra azonban feltétlenül ügyeljünk minden `mount` parancsnál. A */mnt*-t ne használjuk csatolási pontként, mindig egy alkönyvtárába fűzzük be az eszközöket. Ellenkező esetben a rendszert lefagyaszthatjuk. Remélem, egy hasznos gyűjteményt ismert meg az Olvasó a *System Rescue CD* képében. Helyszűke miatt sok érdekes program nem kapott itt helyet, és arról sem esett szó, hogyan lehet telepíteni a *System Rescue CD*-t a merevlemezre, illetve miként lehet testre szabni a rajta található *Linuxot*. Ha a rendszer felkeltette a kedves Olvasó figyelmét, látogasson el a projekt honlapjára, ahonnan a *PDF* formátumú dokumentációt letöltve további hasznos információkhoz juthat. Kellemes pihenést a helyreállított rendszerek után!

Fülöp Balázs
(bigwig42@gmail.com)
21 éves, imádja a Túró Rudit, a Debian Linuxot és a teheneket. Kedvenc írója Slawomir Mrożek. Leginkább a számítógépes hálózatok biztonsága érdekli. A BME VIK műszaki informatikus szak hallgatója.

Az Ubuntu Linux telepítése és beállítása (1. rész)

Ez a magyar fülnek elég különösen hangzó nevű Linux terjesztés 2004. októberében jelentkezett első stabil kiadásával, a nem kevésbé különös elnevezésű „Warty Warthoggal”. A projekt mögött az angliai Man szigeten bejegyezett Canonical Ltd. áll, amely egy dél-afrikai milliárdos, Mark Shuttleworth tulajdona. A cég azt tűzte ki céljául, hogy a népszerű és a rendszergazdák körében igen elterjedt Debian Linuxra építve egy könnyen telepíthető asztali gépekre szánt rendszert hoz létre.

Miután ez év júliusában az *Ubuntu Alapítvány* 10 millió dolláros alaptőkével megalakult, erre a nagyszerű kezdeményezésre még nyilván sokáig számíthatunk.

Az *Ubuntu* pillanatok alatt a *Linux* felhasználók kedvence lett, olyannyira, hogy a *Distrowatch.org*-on jó ideje vezeti a népszerűségi listát. Ez többek között annak köszönhető, hogy a *Debianra* építkezik, amely deklaráltan szabad rendszer, óriási csomagkészlettel és könnyen használható csomagkezelővel is rendelkezik. Valószínűleg annak is jelentős része volt a sikerben, hogy a fejlesztők a felhasználók számára nagyon hízelgő ígéretet tettek: félévente új kiadást készítenek. Ezt pedig egyelőre be is tartották: az októberi első kiadást követte ez év áprilisában a második és ígéretünk van a harmadik kiadás októberi megjelenésére is. Ennek megfelelően az egyes kiadások számozása sem a szokásos módon történik, vagyis itt az 1.0-át nem a 2.0 követi és így tovább, hanem a verziószám mindig a kiadás dátuma. Az első *Ubuntu* tehát a 4.10-es számot kapta (lévén a 2004. év 10. havában látta meg a napvilágot), a második az 5.04-et és a következő, a harmadik az 5.10-es lesz.

De hagyjuk a számokat és foglalkozzunk a nevekkel. Honnan az *„Ubuntu”* elnevezés? A szó afrikai eredetű és ott széles körben ismert is.

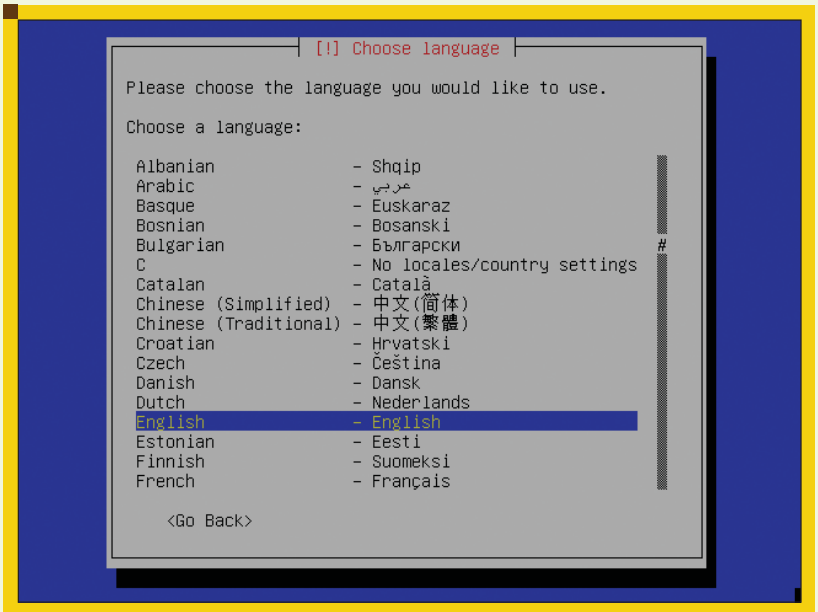
1. ábra Az Ubuntu Linux telepítőjének nyitóképe

Nehéz magyarra lefordítani, mert kifejez érzést, vágyat, cselekedetet, talán úgy mondhatnánk, ahogy a honlap is fordítja, hogy az *ubuntu „emberség másokkal szemben”*. Nem véletlenül afrikai szó, hiszen *Mark Shuttleworth* is afrikai származású.

Talán még érdekesebbek az egyes kiadások elnevezései (kódnevei). Ezek az afrikai kontinens egy-egy állatáról kapják nevüket, megtoldva egy vicces jelzővel is, ami így angolul egyben egy szójáték is.

A 4.10-es a *„Warty Warthog”* nevet kapta, azaz ő a *„Bibircsókos varacskos disznó”*, az 5.04-es lett a *„Hoary Hedgehog”*, vagyis *„Őszülő sündisznó”*, végül az 5.10-es fedőneve *„Breezy Badger”* ami magyarul körülbelül *„Szeles borz”* lehetne.

Az *Ubuntu* a *Gnome* ablakkezelőre építkezik. Az első kiadás idején a nem kevésbé népszerű *KDE*-t még egyáltalán nem támogatta, de mára már ez a felhasználói felület is teljes támogatottságot élvez. *Kubuntu* néven rész-



2. ábra Indulhat a munka

3. ábra A rendelkezésre álló terület felosztása

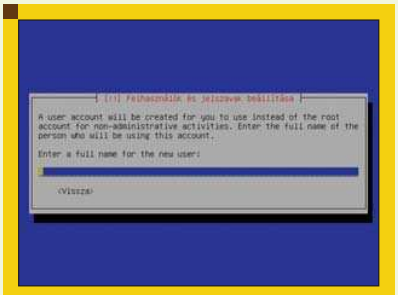
ben külön szálon fut a *KDE*-s *Ubuntu* fejlesztése, amelynek telepítése azonban gyakorlatilag megegyezik a *Gnome* alapú változatéval, így ezt külön nem is tárgyaljuk.

Az Ubuntu beszerzése

Az *Ubuntut* legegyszerűbben a hivatalos honlapról tölthetjük le, hiszen ez egy szabad szoftver, azaz mindenki számára szabadon és ingyenesen hozzáférhető. Keressük fel a ➲ *http://www.ubuntulinux.org/download/* címet és válasszunk egy szimpatikus letöltőhelyet. Innen a *CD*-képet (*.iso* formátumban) tudjuk letölteni majd kedvenc *CD*-író alkalmazásunkkal elkészíthetjük az *Ubuntu* telepítőlemezeit.
Akinek nincs szélessávú internet kapcsolata, az se keseredjen el, hiszen a ➲ *http://linuxcdiras.uw.hu* címen nagyon kedvező áron megrendelheti a megírt *CD*-t, de akár az én címemre is nyugodtan írhatnak azok, akik *Ubuntu CD*-t szeretnének.

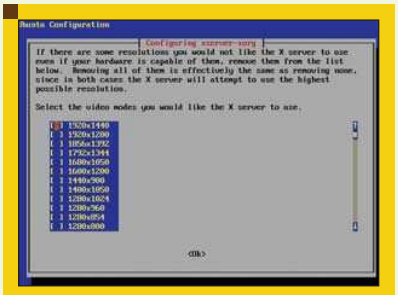
Telepítés

Egy új operációs rendszer telepítése mindig izgalmas és legfőképpen komoly dolog, ezért kellő elszántsággal és fokozott várakozással készüljünk az eseményre. Ne csak úgy, „két rohanás között”, hanem kellő időt rászánva, nyugodt körülményeket biztosítva tegyük ezt meg. Kellő odafigyelés nélkül, „erőltetett menetben” könnyen csalódás érhet minket, sőt ami még rosszabb, esetleg már meglévő adataink is elveszthetnek.
Ezért figyelmesen olvassuk el és hűen kövessük a telepítés lépéseit, különösen ha még járatlanok vagyunk a *Unix* rendszerek telepítése terén.
Bár – ahogy látni fogjuk – az *Ubuntu* telepítése nagyon egyszerű, néhány kérdésre mégis határozott választ fog várni tőlünk.
A legjobb, ha egy külön merevlemezt tudunk biztosítani az *Ubuntu* számára, de ha ez nem megoldható, akkor legalább 3-4 gigabájt szabad területet szabadítsunk fel. (A *Windows*-al ellentétben jó pár hasznos szoftvert kapunk együtt az alap telepítéssel, így több helyre lesz szükségünk, mintha csak egy „csupasz” operációs rendszert telepítenénk). Ha egy meglévő *Windows* partíciónk van, ami a teljes merevlemezt elfoglalja, akkor segíthet például a *Partition Magic* nevű szoftver, ami képes a partíció méretét lecsökkenteni.
Vigyázat! Fontos adatainkról mindig készítsünk mentést egy ilyen kényes művelet előtt és számítsunk rá, hogy esetleg elég sokáig fog tartani.
Ha nem csak próbálgatásra, de komoly munkára is szeretnénk használni az *Ubuntut*, ajánlatos nagyobb helyet biztosítani a számára, igényeinknek megfelelően.
Miután eldöntöttük, hová telepítjük a rendszert, tegyük be a telepítő lemezt CD-olvasónkba és indítsuk újra a számítógépet (ügyeljünk arra is, hogy bootoláskor a CD-ről induljon el gépünk).
Újra felhívom a figyelmet a mentés szükségességére: pótolhatatlan anyagainkat tegyük biztonságos helyre, ha esetleg fatális hibát ejtenénk a telepítés vagy partícionálás során!
Ha sikeresen bootoltunk a CD-ről, megjelenik az *Ubuntu* telepítő kezdő képernyője (*1. ábra*), ahol megadhatjuk a betöltési paramétereket is, ha szükségesnek látjuk. Ezeket közvetlenül beírhatjuk. Ha nem adunk meg semmit, hanem csak *ENTER*-t nyomunk, a szokásos telepítési beállításokkal indul el az *Ubuntu* telepítő. Ha minden részlet fontos számunkra, az *expert* szót írjuk be. Ennek az írásnak nem célja az expert telepítés bemutatása, haladóknak viszont nagyon hasznos lehet.
Egyes laptopoknál nem jól jelennek meg a telepítési képernyők. Ha ilyet tapasztalunk, indítsuk újra a gépet és adjuk meg a *vga=771* opciót. A telepítési paraméterek részleteiről az *F1* billentyű leütésével tájékozódhatunk.
Ez után elindul a telepítő rendszer betöltése, aminek a végén maga a telepítést segítő szoftver képernyő-

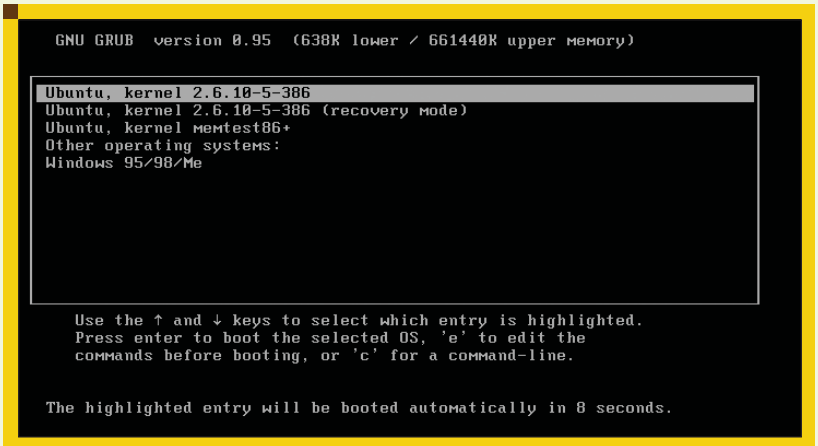


5. ábra Létrehozunk egy felhasználói fiókot

6. ábra Települnek a csomagok...



7. ábra A képernyő felbontásának beállítása



8. ábra Indulás a GRUB-bal

je fogad (*2. ábra*). Csak a billentyűzetet használhatjuk, az egér még nem működik. Az első teendők a nyelv kiválasztása. Ez nemcsak a telepítés nyelvét, hanem a telepítendő ***Ubuntu*** rendszerünk nyelvét is meg fogja határozni. Válasszuk ki a magyart (***hungarian***) úgy, hogy lefelé görgetjük a listát a kurzormozgató billentyűkkel vagy háromszor megnyomjuk a ***H*** billentyűt.

A következőben a billentyűzetkiosztást kell megválasztani. Ha magyar nyelvet választottunk, automatikusan a magyart fogja felajánlani, egyébként választhatunk egy listából is, sőt le is tesztelhetjük, helyes-e a választásunk.

Ezek után automatikus, a telepítés folytatásához szükséges műveleteket fogunk látni: először a különböző CD-meghajtók eléréséhez szükséges modulokat tölti be, majd a már olvasható cd lemezünket átvizsgálja, minden rajta van-e, majd a telepítés során szükséges további modulokat tölti be és felderíti, milyen hálózati kártyánk van. Ha minden sikeresen lezajlott, a hálózatot megpróbálja automatikusan konfigurálni ***DHCP*** segítségével. Ez egy olyan protokoll, amit arra használunk, hogy egy hálózatban gépünk hálózati beállításait a szervertől lekérdezzük. Ha ez sikeres, a hálózattal nincs több dolgunk, de ha nem, kézileg meg kell adnunk IP címünket, az alhálózati maszkot és az alapértelmezett átjárónk címét. Ha nem tudjuk, mik ezek, forduljunk rendszergazdánkhoz vagy egy hozzáértő szakemberhez.

A következő lépésben megadhatjuk gépünk nevét (***hostname***). Egyes hálózati szoftverek felhasználják ezt a nevet, így például ha rendszerünk egy ***Windows*** hálózatnak is része, akkor ez a név fogja gépünket azonosítani.

Ha ezzel végeztünk, el is érkeztünk a telepítés legkényesebb pontjához, a partícionáláshoz. A partíciók segítségével tudjuk felosztani merevlemezünket egymástól független részekre. Az ***Ubuntut*** természetesen legalább egy, más operációs rendszerektől független partícióra kell telepíteni.

A telepítő előbb felderíti, milyen merevlemezek vannak a gépben, betölti az ehhez szükséges modulokat, majd elindítja a partícionáló programot (*3. ábra*). Ha elrontunk valamit, ne ijedjünk meg: a változások csak akkor lépnek életbe, ha a partícionálás végén ezt engedélyezzük. Tehát ha belegabalyodnánk a partícionálásba, nyugodtan lépjünk ki és indítsuk újra a számítógépet.

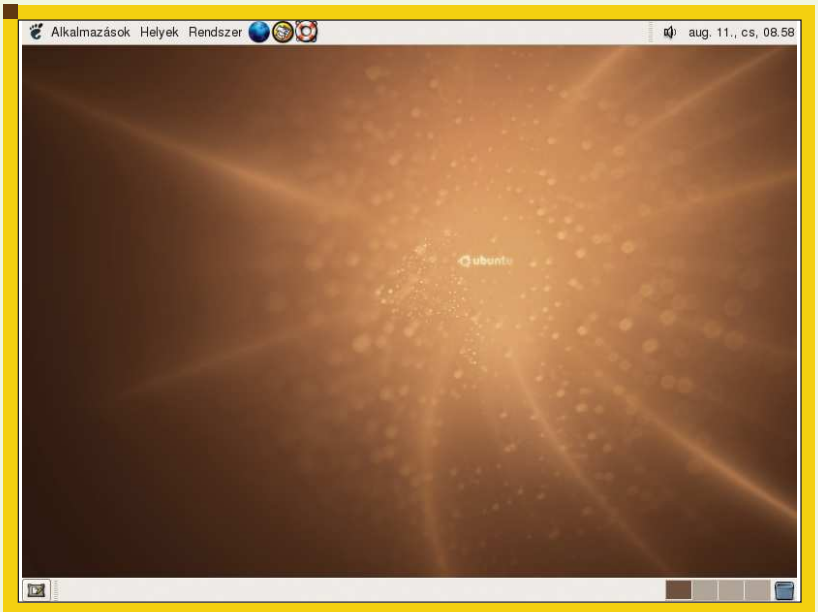
Ha független merevlemezre telepítjük az ***Ubuntut***, választhatjuk az ***„Egész lemez törlése”*** pontot. Itt azt a lemezt válasszuk ki, amelyiken nincs semmilyen értékes adat, mert ez nem visszavonható művelet. Zárójelben láthatjuk, melyik merevlemezről van szó. Az első a ***hda***, a második a ***hdb***, a harmadik a ***hdc*** és így tovább. Aki ***Windowshoz*** szokott, a ***hda***-t megfeleltetheti a ***C:***-nek, a ***hdb***-t a ***D:***-nek, a ***hdc***-t az ***E:***-nek és így tovább.

Amennyiben már van partíciós tábla a lemezen, ahová telepíteni kívánjuk, úgy a ***„Legnagyobb összefüggő szabad hely”*** lehetőséget is kiválaszthatjuk, ekkor az ***Ubuntu*** a fel nem használt részére telepedik, nem zavarva a többi partíciót.

Ha tudjuk, mit teszünk, választhatjuk a ***„Partíciós tábla kézi szerkesztését”*** is, de ezt csak haladóknak ajánlom.

Aki még nem gyakorlott a lemezek partícionálásában, csak akkor kísérletezzen, ha nincsenek pótolhatatlan bármelyik merevlemezen.

Miután ezen a lépésen is túl vagyunk, megjelenik, mely partíciók lesznek megváltoztatva és válaszolnunk kell arra a kérdésre, hogy ***„Lemezre írja a változásokat?”*** Itt kell eldöntenünk: az előző választásunk helyes volt-e, mert a lemezre írás után visszavonásra már nem lesz lehetőségünk. Megfigyelhetjük, hogy nemcsak egy, hanem



9. ábra Íme az eredmény

legalább két partíciót hoz létre a telepítő: a ***swap***, azaz ***cserepartíció*** hasonló a ***Windows*** virtuális memóriájához: a fizikai memória kiterjesztése ez a merevlemezre. Természetesen az az ideális, ha nem kell majd használat közben cserepartíciót használnunk (és a ***Linux*** igyekszik keveset is használni), de előfordulhat, hogy programjaink számára több memória kell, mint amennyi gépünkben rendelkezésre áll. Ilyenkor fog ***Linux*** rendszerünk cserepartíciót is használni, ami persze lényegesen lassabb, mint a fizikai memória.

Amennyiben igennel válaszoltunk az előző kérdésre, elkezdődik a partíciók lemezre írása, majd a megformázásuk és automatikusan elindul az ***Ubuntu*** alaprendszer telepítése is. A következő részben részletesebben fogok írni a linuxos fájlrendszerekről, de már most jó tudnunk, hogy ezek nem kompatibilisek a ***Windows*** fájlrendszerekkel, így az ubuntus partícióinkat ***Windows*** alól nem érhetjük majd el. Fordítva ennek semmi akadálya: mind a ***FAT***, mind az ***NTFS*** fájlrendszereket tudja használni az ***Ubuntu***. Alapértelmezésben az ***ext3*** fájlrendszert fogja használni, aki a kézi partícionálást választja, az mást is kiválaszthat, például a népszerű ***reiserfs***-t.

Az alaprendszer telepítése után újabb kérdésbe botlunk, ugyanis telepítő CD-nk nem tartalmazza a teljes magyar nyelvi támogatást. Ha van szélessávú internetkapcsolatunk, úgy a kérdésre, hogy letöltse-e ezt az internetről, válaszoljunk „igen”-el. Ha nincs, akkor a nemet válasszuk, hogy elkerüljük a bosszantó hibaüzeneteket, amikor az internetre szeretne majd kapcsolódni a telepítő. Ne aggódjunk: teljes nyelvi támogatás nélkül is használható rendszert kapunk, sőt a nyelvi fájlokat később is telepíthetjük.

A következő kérdésre, hogy a ***„Hardweróra GMT szerint jár?”*** általában a ***nem*** a jó válasz, hiszen a helyi idő szerint jár gépünk órája (a ***GMT*** a greenwichi középidőt jelenti). Ekkor nagyon helyesen felajánlja nekünk a budapesti időzónát, amit elfogadhatunk.

Ezek után a telepítő létrehoz egy alapfelhasználót (*5. ábra*), aki nemcsak egy szokványos felhasználó lesz, hanem olyan, aki a ***root***, azaz a rendszergazda nevében tud műveleteket végrehajtani. Az ***Ubuntu*** filozófiája szerint ugyanis a root felhasználó engedélyezése nem biztonságos. Így alapállapotban nem is tudunk majd rootként belépni gépünkre. Erről a beállításoknál még részletesebben lesz szó.

Most az új felhasználó valódi nevét kell megadnunk, majd a felhasználói fiókhoz rendelt nevét (amivel bejelentkezünk), végül a jelszót.

Ezután az ***apt*** csomagkezelőt állítja be (ha szerencsénk van, teljesen automatikusan), végül a ***GRUB*** bootkezelő (róla a következő számban lesz szó). Itt – ha már van operációs rendszer a gépünkön és azt fel is tudta deríteni – megkérdezi, hogy hová telepítse. Ha már van ***Windows*** partíciónk, ne aggódjunk, a ***GRUB***-ot nyugodtan telepíthetjük a fő rendszerindító rekordba, mert az ***Ubuntu*** beállítja, hogy bootoláskor kiválasztható legyen, melyiket kívánjuk elindítani (*8. ábra*).

Az újraindítás előtt a telepítő – nagyon helyesen – kiadja a ***CD***-t, amit most vegyünk is ki a tálcából és válasszuk a ***tovább*** pontot.

Ekkor gépünk újraindul, betöltődik az ***Ubuntu*** alaprendszer, majd megkezdődik a felmásolt csomagok kibontása és telepítése (*6. ábra*). Ez hosszú ideig eltart, ráadásul unalmas is, hiszen a képernyőn megjelenő információk menetébe nem avatkozhatunk be, érdemes ilyenkor egy időre magára hagyni gépünket és más tevékenység után nézni.

Egyszer azért mégis megkérdez a grafikus rendszer beállításakor, hogy milyen képernyőfelbontásokat engedélyezünk (*7. ábra*). Itt be kell jelölni, szerintünk (a telepítő által már bejelölteken kívül) mit tud a monitorunk. Jó esetben, csak egy ***ENTER***-t kell nyomni és mehet tovább a csomagok telepítése.

Türelmünk rózsát, akarom mondani ***Ubuntut*** terem és ha nem fogyunk ki a tárhelyből idő előtt, valószínűleg sikeresen elindul a ***Gnome*** rendszer bejelentkezés menedzsere, a ***gdm***. Választott nevünk és jelszavunk megadása után a várva várt ***Ubuntu Gnome*** asztal fogad minket (*9. ábra*).

A következő számban feltelepített rendszerünk alapelemeivel és beállításaival fogunk foglalkozni.

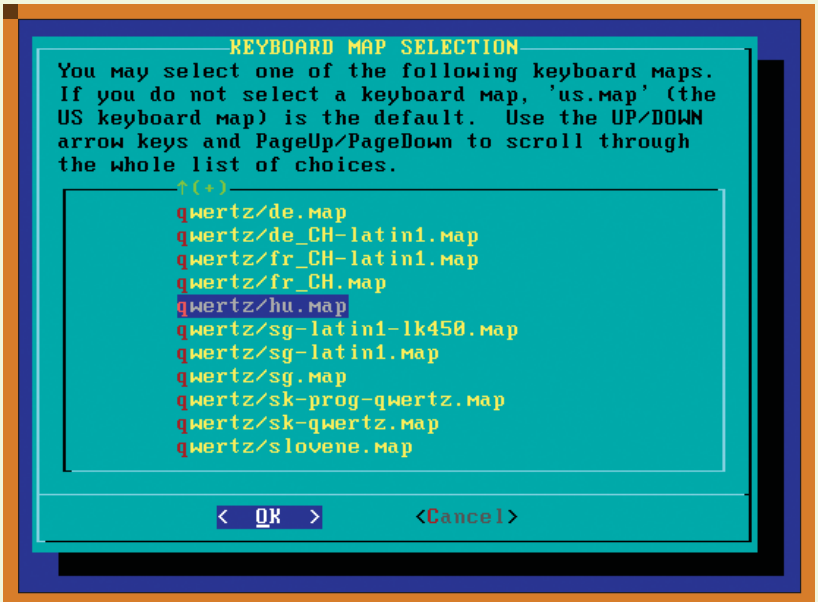
Addig is kellemes ***Ubuntuzást***!

Molnár Norbert
34 éves, rendszergazdaként dolgozik, 5 éve foglalkozik Linuxszal. Főként a szabad szoftverek és a számítógépes biztonság érdekli. Budapesten él, hobbija – ha éppen nem egy újabb disztribúciót tanulmányoz és nem utazik – a csillagászat és a filozófia – lehetőleg jó vörösbor mellett.
(molnar.norbert@gmail.com)

Pipázó pingvin – a Slackware telepítése

Mielőtt nekiállnánk felköltöztetni a Slackware-t a számítógépre, mi különbözteti meg ezt a veterán korú disztribúciót a többitől. Bár nemrég ünnepelte tizenkettedik születésnapját, még a jelenleg aktuális 10.1-es változata is maradéktalanul megfelel a létrehozásakor legfontosabbnak tartott szempontoknak: legyen egyszerű, átlátható, és legyen UNIX-szerű.

Így a *Windowsról* frissen áttérő „Linux-ifjoncoknak” nem biztos, hogy ez lesz a nyerő választás, de aki szeretné kívül-belül megismerni az operációs rendszerét és kedvenc időtöltése a beállítófájlok szerkesztgetése, az könnyen beleszerethet!
A rendszer nagy előnye, hogy igyekszik mindenből az eredeti, vagyis nem foltokkal telezsúfolt forrásprogramokat felhasználni. Ez igaz a kerneltől egészen az apró segédprogramokig. Így sokkal kisebb az esélye annak, hogy programfordításkor azon kell törni a fejünket, vajon melyik csomag melyik félresikerült módosításának következménye a „make error” üzenet. A *Slackware* telepítése sok tekintetben elüt a megszokottól (már amennyiben van megszokott módszer).

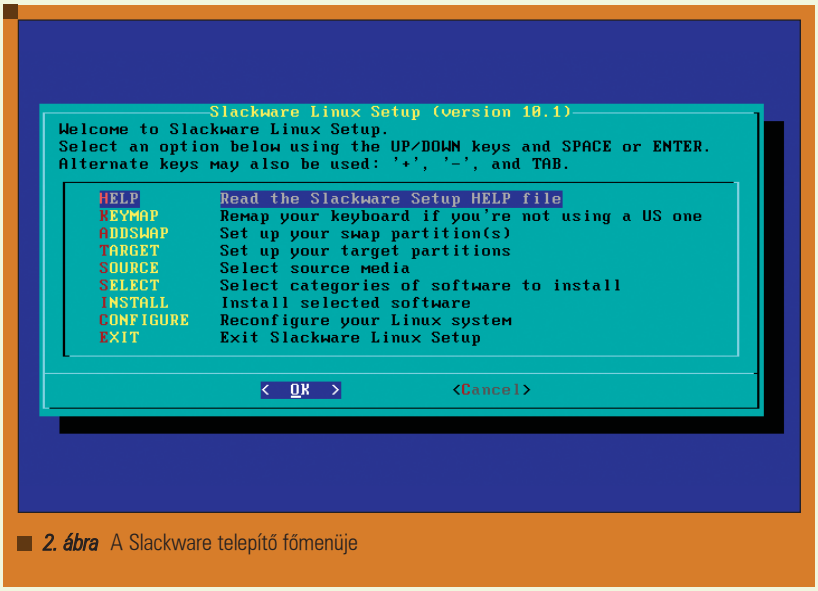


1. ábra Hátul keressük a magyarokat

Telepítés

Ennyi bevezető után kezdjünk neki a telepítésnek. Ehhez szükségünk lesz két (KDE és Gnome nélküli telepítés esetén csak egy) CD-re. Az *iso* formátumú lenyomatok (image) a ➲ *http://www.slackware.org* címen található webhelyről, vagy valamelyik tükörszerverről tölthetők le.
Minden kiadás megjelenésekor felkerülnek az oldalra *torrent* fájlok is, amelyek a disztribúció megjelenése után egy ideig sokkal gyorsabb letöltést tesznek lehetővé mint a túlterhelt szerverek.
A 10.1-es verziószámot viselő legifjabb kiadás négy lemezt tartalmaz. Ebből a telepítéshez csak az első kettőre lesz szükség, a másik kettő a forráskódokat tartalmazza.

A CD-ről történő indítás után azonnal egy promptot kapunk, ahol kiválaszthatjuk a telepítéshez használandó kernelt. Az alapbeállítás szerinti `bare.i` többnyire megfelelő lesz (vagyis az indításhoz egyszerűen csapjunk az *Enterre*). Ha viszont például *SCSI* kártyát vagy *ACPI*-t szeretnénk használni, az *F2* majd *F3* megnyomása után kapunk egy listát a választható kernelekről. A megfelelő nevet begépelve léphetünk tovább.
A következő lépés a billentyűzetkiosztás kiválasztása. Ha nem amerikai elrendezést szeretnénk használni, hanem – mondjuk – magyart, gépeljünk be egy 1-est, majd nyomjuk le az *Entert*. A felugró listából válasszuk a majdnem legalul található *qwertz/hu.map*-ot.
A következő ablakban ellenőrizhetjük, hogy a megfelelő kiosztást választottuk-e. Az ékezetes karakterek itt még nem fognak működni, ezen ne lepődjünk meg! A választott elrendezést egy 1-es begépelésével fogadhatjuk el, a módosításhoz 2-est írjunk be.
Ezek után egy bejelentkezési (login) prompt fogad bennünket, ahol a root felhasználónevet begépelve léphetünk be. Ez elég szokatlan az egyéb telepítőkhöz képest, de próbáljunk hozzá-



2. ábra A Slackware telepítő főmenüje

szokni, hogy a ***Slackware*** nem a hagyományos Next-Next-Finish megoldásokat követi, hanem az utasításainkra vár.

A belépést követően egy rövid tájékoztató szöveget olvashatunk a régebbi verziók frissítéséről, de akinek ilyesmi jár a fejében az inkább olvassa el a CD gyökerében található ***UPGRADE.TXT*** fájlt. Az utolsó néhány sor viszont a telepítéssel kapcsolatos teendőkről szól: ha a telepítéshez (tehát nem a működő rendszerhez) szükségünk van ***PCMCIA*** vagy hálózati támogatásra (például ha hálózaton keresztül szeretnénk telepíteni a rendszert), gépeljük be a `pcmcia` illetve `network` parancsokat.

A telepítéshez szükségünk lesz legalább két partícióra. Ezek létrehozásához adjuk ki a `cfdisk` (hardcore ***Unix/Linux*** guruk akár az `fdisk`) parancsot. Fontos megjegyezni, hogy mindkét particionáló eszköz az első merevlemezzel dolgozik, ha nem erre szeretnénk telepíteni a ***Linuxot***, ezt a megfelelő paraméterrel tudathatjuk. Ha például a második merevlemezt akarjuk felosztani adjuk ki a `cfdisk /dev/hdb` parancsot.

Üres merevlemez esetén egy kérdést kapunk, miszerint szeretnénk-e üres partíciós táblával indulni, erre feleljünk igennel (Y). Az első partíció amire szükségünk lesz, némi csereterület (swap). Ennek ajánlott méretéről megoszlanak a vélemények, de a mai merevlemez árak és méretek mellett nem érdemes ezen spórolni. Ha ugyanis később kiderül, hogy alulméreteztük ezt a lemezrészt, sokkal körülményesebb lesz segíteni a problémán. A ***New*** majd a ***Primary*** menüpontot választva hozzuk létre tehát a swap partíciót az üres hely elején (Beginning), a mérete pedig legyen mondjuk 512 MB.

Ezek után szükségünk lesz egy partícióra a Linux fájlrendszere számára. Hozzuk létre ezt is az előzőhöz hasonlóan (előtte a kurzormozgató billentyűk segítségével álljunk rá a ***Free space*** jelzésre). Méretnek most választhatjuk a program által felajánlott maximális méretet, de egy teljes telepítéshez legalább 4 GB szükséges. Hasznos lehet egy ***home*** partíció elkészítése is, főleg ha később le szeretnénk cserélni a használt ***Linuxot*** egy másikra, de szeretnénk a beállításainkat megőrizni. Ha létrehoztuk a partíciókat, állítsuk be a típusukat. A ***Type*** menüpontot választva listát kapunk a lehetséges kódokról, a cserterülethez válasszuk a 82-t, a fájlrendszerhez pedig a 83-t. Ezek után a ***Write*** parancsot kiválasztva, majd ***„yes”*** választ begépelve írjuk ki a lemezre a partíciós táblát, és lépjünk ki a ***Quit*** menüpontra lépve. Ha nem üres merevlemezre tesszük a ***Slackware***-t (hanem például egy már meglévő ***Windows*** mellé), legjobb lesz ha logikai (***Logical***) partíciókat hozunk létre.

Miután előkészítettük a terepet, a telepítőt a `setup` paranccsal indíthatjuk. Egy kilenc pontból álló menüt kapunk, melyen a telepítés folyamán szépen végig fogunk lépkedni.

Az első egy tájékoztató a telepítésről, a második a már ismert billentyűzetkiosztás-választó. A következő lépés az ***ADDSWAP***, a cserpartíció beállítása. Ha az imént létrehoztuk a megfelelő 82-es típusú partíciót, a telepítő ezt automatikusan megtalálja, vagyis csak az Entert kell leütnünk a jóváhagyáshoz.

A ***TARGET*** pontban azt a partíciót kell kiválasztanunk, ahova a rendszert telepíteni szeretnénk. Ha ez megvan, választhatunk gyors illetve lassú (hibás blokkokat is ellenőrző) formázást (***Format/Check***), vagy ki is hagyhatjuk ezt a lépést, ha korábban már megtörtént ez a művelet (***No***).

Mi most válasszuk az első lehetőséget (***Format***), és döntsük el, milyen fájlrendszert szeretnénk használni a felkínáltak közül. Az alapértelmezett a ***reiserfs***, de használhatunk ***ext3***-t is. ***Ext2***-t semmiképp ne válasszunk rendszerpartíciónak, mert egy esetleges meghibásodás vagy áramszünet után a másik kettőnél gyorsabb a helyreállítás.

A célterület kiválasztása és formázása után a telepítendő csomagok forrását kell kiválasztanunk. Ez legegyszerűbb esetben a meghajtóban levő ***CD***, de választható hálózati telepítés vagy merevlemezen található könyvtár is. ***CD*** használata esetén a telepítő felkínálja, hogy megkeresi a lemezt. Ezt nyugodtan hagyjuk rá, meg fogja találni.

A következő lépés a telepítendő csomagok kiválasztása. A csomagok csoportokba vannak szervezve. Ha nem szeretnénk teljes telepítést, itt gyorsan megszabadulhatunk a nem kívánt alkalmazásoktól, hiszen elég, ha kivesszük előlük az X jelzést. Alapesetben egy nekünk fontos csomagcsoport nincs bejelölve, nevezetesen az, amelyik a KDE kiegészítő nyelveit – köztük a magyart – tartalmazza.

Ha nincs szükségünk több tucat más nyelvre, hagyhatjuk a dobozt jelöletlenül, később is feltehetjük a magyar nyelvű támogatást. Továbblépve a csomagválasztás részletességét szabhatjuk meg. Legegyszerűbb a teljes (full) telepítés ami minden földi jóval ellátja rendszerünket, de akár teljesen testre is szabhatjuk a csomagválasztékot.

Arra viszont ügyeljünk, hogy a ***Slackware*** nem kezeli a csomagfüggőségeket, így könnyen kiránthatjuk

a rendszer lába alól a talajt, ha olyasmit hagyunk ki, ami nélkül Linuxunk el sem indul. A csomagok kiválogatása után elkezdődik azok merevlemezre telepítése. Ez el fog tartani egy darabig, a folyamat közben azonban folyamatos tájékoztatást kapunk az éppen telepítés alatt álló csomag nevéről, méretéről és elolvashatjuk a hozzá tartozó leírást is – igaz, ehhez néha Supermant megszégyenítő olvasási sebességre lenne szükség.

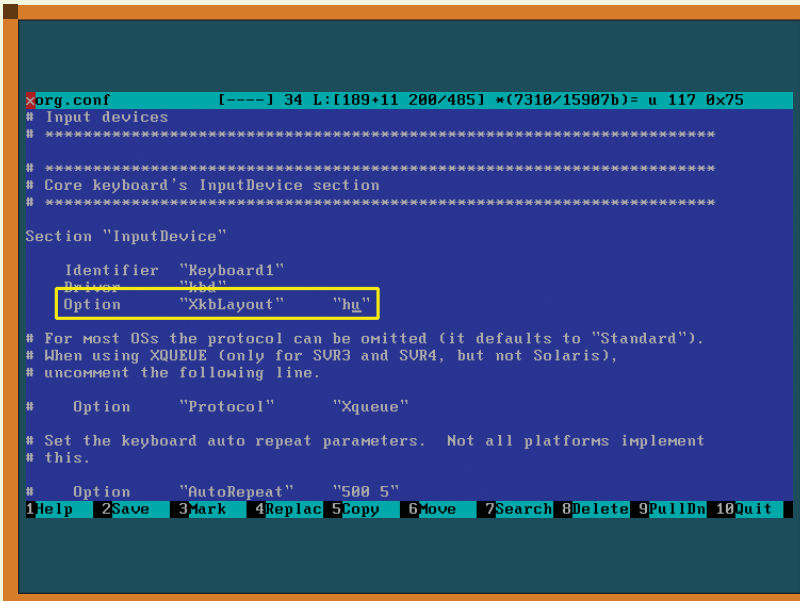
Egy lemezcserét követően a használandó kernelt kell megválasztanunk. A cikk elején említett kernellistából válogathatunk a második menüpontot (*cdrom*) választva. A telepítő felkínálja a lehetőséget egy rendszerindító lemez elkészítésére. Mivel a második telepítőlemez ***LiveCD***-ként is működik, így egy „halott” rendszer esetleges újraélesztése is elvégezhető vele, vagyis nem feltétlenül szükséges bootfloppyt készíteni. Viszont ha van kéznél egy felesleges üres floppy, egyszer még jól jöhet.

A következő kérdés a modemre vonatkozik. Ha van már ilyen eszköz a géphez csatlakoztatva, a helyének megfelelően válaszoljunk. Ha még nincs, akkor vagy válasszuk az első lehetőséget (ilyenkor nem keletkezik */dev/modem* szimbolikus link), vagy ha elképzelhető a jövőben egy modem érkezése, döntsük el, hogy valószínűleg hogyan csatlakozik majd a géphez és ez alapján válasszuk ki a megfelelő menüpontot.

Ezután a ***hotplug*** indításkori engedélyezéséről kell döntenünk. ha használunk például ***USB***-re csatlakozó ***pendrive***-ot, még jól jöhet ez a funkció, tehát válaszoljunk igennel. Cserébe a rendszer elindulása valamivel hosszabb időt igényel.

A rendszerbetöltő (ami a ***Slackware*** esetében tradicionálisan a ***Lilo***) a nevének megfelelően gondoskodik a rendszer indításáról, illetve lehetőséget biztosít több operációs rendszer (például ***Linux*** és ***Windows***) használatára, vagy akár egy rendszer több különböző verziójú rendszermagja közti választásra. Aki nem ismeri a ***Lilo*** konfigurációs fájljának felépítését, annak célszerű a ***simple*** opció választása, amivel ezzel a telepítő megpróbálja automatikusan felépíteni azt.

A következő kérdés a ***framebuffer*** használata. Ha ***nVidia*** chipes videokártyánk van, jobb nem kihívni magunk ellen a sorsot, és a standard opciót választva a hagyományos szöveges képernyőnél maradni (ezzel egyébként is sok bosszúságtól kímélhetjük meg magunkat), de ha valaki szeretné a rendszer indulása közben a pingvines logót látni, vagy fontos a 80x25-nél nagyobb felbontású konzol, ízlésének megfelelően válasszon képfelbontást.



3. ábra Grafikus felületen is legyünk magyarok

A ***Lilonak*** különféle paramétereket adhatunk meg (amiket aztán induláskor a rendszer értelmez), ezeket a következő dialógusablakban megjelenő sorba írhatjuk be. ***CD***-író és 2.4-es sorozatú kernel használata esetén általában ***SCSI***-emulációt szokás használni, ezt a telepítő által mutatott példának megfelelően adhatjuk meg, például a ***secondary masterként*** beállított, tehát ***Linux*** alatt `hdc` eszköz esetében a `hdc=ide-scsi` paraméterre lesz szükségünk.

A rendszerbetöltővel kapcsolatos kérdések közül az utolsó a ***Lilo*** helye. Mivel remélhetőleg nem szeretnénk minden alkalommal floppyról indítani a gépet, a ***Root*** vagy az ***MBR*** lesz a jó választás. Ha van már feltelepített rendszerbetöltőnk, akkor a gyökér fájlrendszer (***Root***), ellenkező esetben az ***MBR*** lesz a megfelelő.

A telepítő érdeklődik egerünk típusáról (görgővel rendelkező rágcsálónál válasszuk az ***imps2***-t) és az egeret kezelő program, a ***gpm*** indításáról (válaszoljunk igennel), majd a hálózat beállításáról (ez utóbbi elmulasztása esetén, vagy ha később kerül hálózatba a számítógép, a `netconfig` parancs lesz segítségünkre).

A következő képernyőn a rendszerbetöltéskor elinduló szolgáltatásokat kell kiválasztanunk. Figyelmesen olvassuk át ezek leírását, és próbáljuk eldönteni, hogy szükségünk van-e rájuk.

Ha esetleg később kiderül, hogy rosszul választottunk, a */etc/rc.d* könyvtárban található szkriptek futtatási attribútumának beállításával tehetjük a megfelelő szolgáltatást automatikusan indulóvá.

Ha nem szeretnénk különleges szöveges módú betűkészleteket használni (ugye nem...), a számítógép órájának beállítását kell tudatnunk a telepítővel. A legtöbb gép esetében (vagy ha ***Windowst*** is használunk) a helyes válasz nemleges lesz. Az ezután következő, időzónákat tartalmazó listában remélhetőleg mindenki megtalálja a ***Europe/Budapest*** sort.

Ha ez megtörtént, az alapértelmezés szerinti ablakkezelőt kell megválasztanunk. Windows-ről áttérők, vagy mutatós grafikus felületet kedvelők válasszák a telepítő által is felkínált KDE-t, de ha a számítógép viszonylag kis teljesítményű, akkor inkább az alatta találhatók közül jelöljük ki valamelyiket.

A következő kérdésre feltétlenül válaszoljunk igennel! Itt beállíthatjuk a *Linux* alatt mindenható root felhasználó jelszavát. Igyekezzünk nehezen megfejthető jelszót választani (ha túl könnyű, a rendszer figyelmeztet, de ha elég erőszakosak és kitartóak vagyunk, ezt is elfogadja). Ezen az akadályon túljutva nincs más hátra mint újraindítani a gépet a *CTRL-ALT-DEL* kombináció lenyomásával.
Az újraindulást követően ismét meglepődhet a gyanútlan *Linux* felhasználó: nincs sehol a látványos grafikus felület, csak egy egyhangú login prompt fogad. Nincs ok az aggodalomra, a Slackware hagyományosan szöveges módban indul. Ezen többféle módon is segíthetünk, talán a legegyszerűbb és legelegánsabb megoldás az alapértelmezett futási szint átállítása a */etc/inittab* fájlban. Nyissuk meg egy szimpatikus szövegszerkesztővel (például az `mc` paranccsal indítható *Midnight Commanderben* keressük meg és nyomjuk le az *F4*-et), és az `id:3:initdefault:` sorban található hármast írjuk át négyesre.

Ha a telepítéskor nem választottunk ki magyar nyelvet a *KDE*-hez, most feltelepíthetjük a megfelelő csomagot. Tegyük be a második *CD*-t, csatoljuk be a

```
mount /mnt/cdrom
```

paranccsal, majd telepítsük fel a csomagot az

```
installpkg /mnt/cdrom/
➥slackware/kdei/kde-i18n-hu-
➥3.3.2-noarch-1.tgz
```

parancs kiadásával. Ha magyar nyelven szeretnénk gépelni is, akkor alapértelmezetté tehetjük a magyar billentyűzetkiosztást a */etc/X11/xorg.conf* fájl szerkesztésével: az `InputDevice` szekcióba szúrjuk be az

```
Option "XkbLayout" "hu"
```

sort.
Slackware alatt többnyire nem okoz gondot a forráskódként beszerzett programok fordítása, de ha nem szeretnénk ezzel tölteni az időt, kedvenc *Slackware*-ünkhöz rengeteg előre fordított csomag található a ➲ *http://www.linuxpackages.net* címen. Ha saját magunk szeretnénk forráskódokat fordítani, rendkívül hasznos eszköz a `checkinstall`, mellyel a lefordított programból több más formátum mellett *Slackware* csomagokat is készíthetünk, lényegesen megkönnyítve ezzel az esetleges későbbi eltávolítást vagy frissítést. A már említett csomagfüggőségi ellenőrzések hiányára is létezik több alternatív megoldás, ilyen például a `swaret` nevű csomagkezelő.
A *Slackware* meglehetősen időigényes disztribúció, de ha valaki rászánja az időt a megismerésére, bizton állíthatom hogy nagyon megszereti.
Sok sikert hozzá!

Bokor Norbert
(doc@coder.hu)
Egy autóipari cégnél informatikus, emellett Győrbe, a Széchenyi Egyetemre jár. A számítógép mellett imádja a társasjátékokat. Most éppen gitározni tanul.

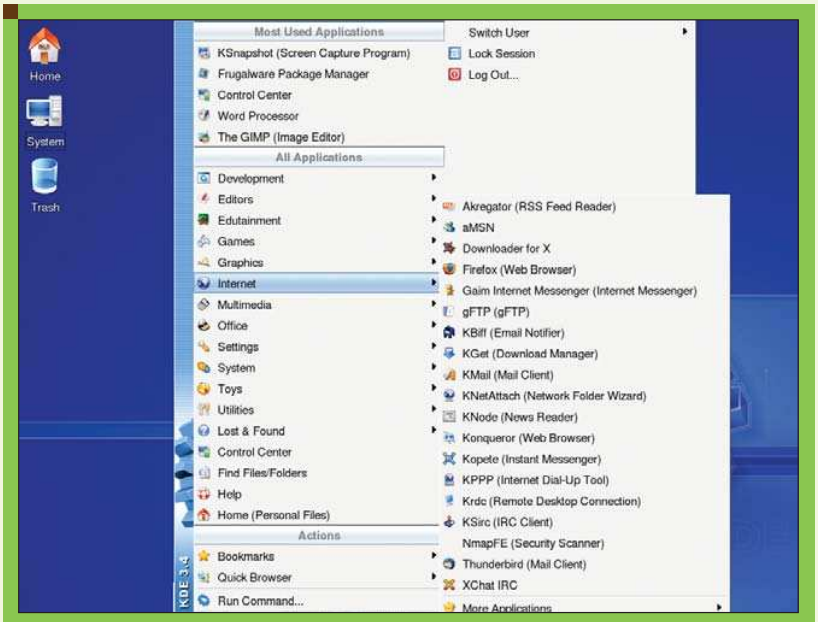
Frugalware – Egy egyszerű de nagyszerű magyar Linux terjesztés

Frugalware, vagyis „gazdaságos áru”. Vajon mit jelenthet ez a szó? A www.distrowatch.org szerint ez egy magyar Linux terjesztés, amely ráadásul benne van a TOP40-ben.

Valami végre elindult... Úgy tűnik, hogy lassan mi magyarok is kezdünk felébredni, hiszen immár több magyar fejlesztésű ***Linux*** terjesztéssel is büszkélkedhetünk.
Tulajdonképpen eddig is voltak olyan hazai fejlesztésű szabad szoftverek, amelyeket nem csak idehaza, hanem külföldön is elismertek, gondoljunk csak az ***Mplayerre***, hogy csak a legismertebbet említsem.
Bár a ***Frugalware Linux*** is a „nemzetközileg ismert” kategóriába tartozik, külföldön valahogy többet foglalkoztak vele, mint idehaza.
Ezt a hiányt kívánjuk most pótolni, és közel egy évvel az első stabil verzió megjelenése után górcső alá vesszük a a ***Frugalware 0.3pre2*** változatot, amely a közeljövőben megjelenő 0.3 stabil változat második béta verziója. A ***Frugalware Linux*** fejlesztése 2003-ban kezdődött. Fejlesztése teljesen önálló, vagyis egyik nagy disztribúció leszármazottjának sem tekinthető. Leginkább talán a Slackware-hez hasonlít, mivel a fejlesztőknek ezzel a terjesztéssel volt komolyabb tapasztalatuk.

Telepítés

A fejlesztők a ***Frugalware Linuxszal*** a középhaladókat célozták meg. Aki tehát éppen csak ismerkedik a szabad világgal, annak elveheti a kedvét a karakteres felületű telepítő, és hogy nem csillogó-villogó grafikus sávok jelzik, hogy mennyi idő van még hátra a miből.



1. ábra Felhasználói programok tekintetében bőséges választási lehetőségünk van

Ez az érzés esetleg később is megmaradhat, mivel a különböző beállítások elvégzése is némileg nehezebb két-három egérkattintásnál, sőt néha itt is karakteres felületet kell használnunk. Aki vállalja mindezt, az cserébe egy kiválóan működő, hajlékony rendszert kap.
A ***Frugalware Linuxot*** a ➲ ***www.frugalware.org*** oldalról tölthetjük le. Amennyiben nem kiszolgálón akarjuk használni, vagyis szükség van grafikus felületre is, akkor mindenképpen töltsük le mindkét ***CD***-t, vagy eleve a ***DVD***-s verziót szerezzük be. Aki szeretné kihasználni gépe 64 bites adottságait, annak sem kell másik terjesztés után néznie, hiszen a ***Frugalware*** elérhető az ***i686***-os architektúra mellett 64 bites változatban is.
A telepítés lényegében nem tér el az ismertebb disztribúciókétól. Talán érdemes megemlíteni, hogy bár a rendszer alapvetően magyar, támogatja az angol, német, olasz, és szlovák nyelveket is. Ez arra utal, hogy a projekt megpróbál szakítani a másik két magyar terjesztés filozófiájával, vagyis nem csak a hazai piacra készíti termékét. Nyilván éppen ezzel magyarázható, hogy a másik kettőnél jóval többen ismerik külföldön.

Azoknak, akiknek ezt a *Linuxot* szánták, a telepítés lépéseit valószínűleg teljesen fölösleges lenne felsorolni. A vállalkozó szellemű kezdőknek a legjobb tanács talán az, hogy ahol nem értik, hogy mit kérdez a telepítő, ott egyszerűen üssenek *Entert*, mivel a rendszer teljesen értelmesen megadott alapértelmezésekkel rendelkezik. Amúgy a *Frugalware* honlapján található egy magyar nyelvű telepítési segédlet, amely lépésről lépésre tartalmazza a folyamat leírását.

Az első találkozás

Miután elindult a rendszer a bejelentkező képernyőn kiválaszthatjuk, hogy melyik ablakkezelőt szeretnénk használni. Választék szerencsére van bőven. A *KDE* és a *GNOME* szinte alapfelszereltségnek számít, de megtalálunk még olyan közkedvelt ablakkezelőket is, mint az *Enlightment* vagy az *XFCE*. Nekem a *KDE* és az *XFCE* a kedvencem attól függően, hogy éppen milyen teljesítményű vason futtatom a *Linuxot*. Először a *KDE*-t próbáltam ki.

A menüből kitűnik, hogy impozáns mennyiségű alkalmazást sikerült rátenni a *Frugalware* csapatnak a mindössze két *CD*-re. Sajnos a menü angol nyelvű, pedig azt ember azt várná, hogy egy magyar terjesztés magyarul szól majd hozzá. Természetesen nincs veszve semmi! Amennyiben a *DVD*-s változatot választottuk, akkor a menük eleve magyarul lesznek, a *CD*-s változatot pedig a *kde-i18n-hu* csomag telepítésével lehet rávenni a szép magyar beszédre.

Kifejezett pozitívum, hogy – több amúgy népszerű terjesztéstől eltérően – kifejezetten sok program található a menüben. Tetszetős az a megoldás is, hogy a programok neve mellé zárójelben oda van írva, hogy mire valók Ez sokszor hasznos lehet, különösen a kezdő felhasználóknak, akik nem biztosak benne, hogy mi mire való. Persze az ilyen információ sokszor még a gyakorlott felhasználóknak is segíthet.

A legalapvetőbb elemek a következők: *2.6.12*-es kernel, *KDE 3.4*, *Gnome 2.10*, *GCC 4.0.1*, *X.org 6.8.2*.

Látható, hogy a kínálat elég friss, ami nem véletlen, hiszen a fejlesztőknek ez az egyik fő célkitűzése. Az első dolog, amit minden terjesztés tesztelésekor megpróbálok beállítani, az *init* folyamat, hiszen a futtatni kívánt szolgáltatások körét a felhasználási cél határozza meg.

A *System/Frugalware Runlevel Editor* segítségével könnyen, és gyorsan elvégezhetjük a szükséges módosításokat. Itt beállíthatunk egyszeri elindítást, illetve azt is, hogy a kérdéses szolgáltatás minden rendszerindításkor automatikusan elinduljon.

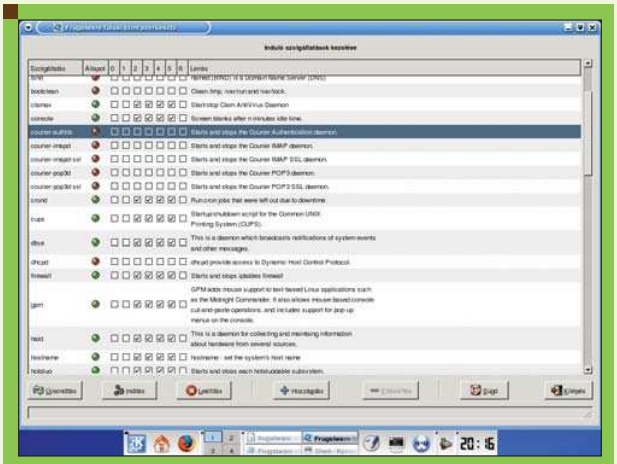
Ha azonban nem telepítettük az *X* felületet, akkor a konzolon keresztül kell mindezt megtennünk. Itt sem vagyunk azonban egyedül a `vi` editorral, mert segít a `service` programocska, amit persze csak adminisztrátorként futtathatunk. Ha például azt akarjuk, hogy a webszerver ne induljon el, adjuk ki a

```
service httpd del
```

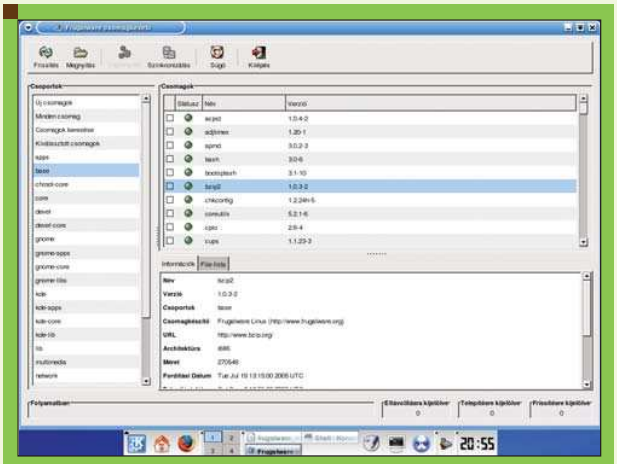
parancsot. Az eredeti állapotot bármikor visszaállíthatjuk a

```
service httpd add
```

paranccsal.



2. ábra Szolgáltatások kezelése a grafikus felületen



3. ábra Grafikus felületű csomagkezelés

Csomagkezelés

A *KDE* menüből a *System/Frugalware Package Manager* menüponttal tudjuk előcsalni a csomagkezelőt, amely első ránézésre leginkább az *Synaptic*-hoz hasonlít. Azzal ellentétben azonban a *Frugalware* csomagkezelése nem `apt` hanem `pacman` alapú.
Mivel valószínűleg nem sokan ismerik a `pacman` csomagkezelőt, talán nem árt néhány egyszerű utasítást itt is ismertetni, ámbár a hivatalos *Frugalware Linux* dokumentációban részletesebb leírás is található.
A `pacman` amúgy *C*-ben íródott, ezért érezhetően gyorsabb a szkript alapú csomagkezelőknél. A csomagadatbázist frissíteni a

```
pacman -Sy
```

paranccsal lehet, magukat a csomagokat pedig a

```
pacman -Su
```

utasítással.

Ha egy adott csomagot szeretnénk feltelepíteni, akkor a

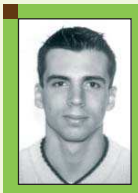
```
pacman -S csomagnév
```

formát használjuk. Ez a kérdéses alkalmazást annak függőségeivel együtt telepíti, ütközés esetén pedig rákérdez arra, hogy eltávolíthatja-e a konfliktust okozó másik csomagot.
A további lehetőségeket a `pacman -h` paranccsal jeleníthetjük meg.

A hivatalos *Frugalware Linux* dokumentációból azt is megtudhatjuk, hogyan érdemes nekiállni a *Frugalware* csomagok készítésének. Természetesen a fejlesztők örömmel veszik az ilyen segítséget, mivel a fejlesztés teljes mértékben nonprofit módon folyik.

A puding próbája

Egy terjesztéssel az ember akkor ismerkedik meg igazán, amikor használni kezdi. Internetezésre, zene lejátszására, videó nézésére, és egyéb mindennapi feladatokra, vagy éppen kiszolgálóként egy kisebb irodában.
A *Frugalware Linux* jól tűrte a megpróbáltatásokat, és gazdag programválasztéka révén szinte az összes, a napi munka során felmerülő feladatot sikerült megoldanom vele. Ráadásul számos esetben több program közül is válogathattam. Találkoztam természetesen kisebb hibákkal, hiányosságokkal is a tesztelés folyamán, de mivel ez egy béta verzió, a fejlesztőknek ezeket még van idejük kijavítani.
Összességében elmondhatom, hogy a *Frugalware* nagyon jó kis disztribúció, és jóval több megbecsülést érdemelne idehaza is.
Azt hiszem eddig inkább csak külföldön figyeltek fel rá, idehaza nem kapott kellő elismerést, pedig „magyar termék”.
Telepítése egyszerű, a dokumentáció pedig bár rövid, teljesen használható az első lépések megtételénél. Megjelenése tetszetős, sok téma, és ablakkeret közül választhatunk.
A karakteres terminálok is grafikus felületen érhetőek el, így elég kellemes és jóval átláthatóbb a megjelenése. Ráadásul az *mc* alatt sem esik szét a kép, mint ahogy tapasztaltam néhány másik terjesztésnél.
A rendszer gyors, ellenben a grafikus felületű csomagkezelőt elég lassúnak éreztem, például a *synaptic*-kal szemben.
Az általam talált legnagyobb hiányosság a rendszerben az, hogy sem a *CD-ROM*, sem a *Pendrive* nem csatolódik fel automatikusan, ami egy desktop rendszernél szerintem alapvető szolgáltatás. Szerencsére a fejlesztők is tisztában vannak ezekkel a problémákkal, így nyilván orvosolni fogják azokat.
Teljesen kezdő felhasználóknak egyelőre nem tudom jó szívvel ajánlani, ellenben akinek már van némi tapasztalata a *Linux* rendszerekkel, és egy jól összeállított, ugyanakkor még sok helyen konfigurálható rendszerre vágyik, az nyugodtan vágjon bele a dologba.



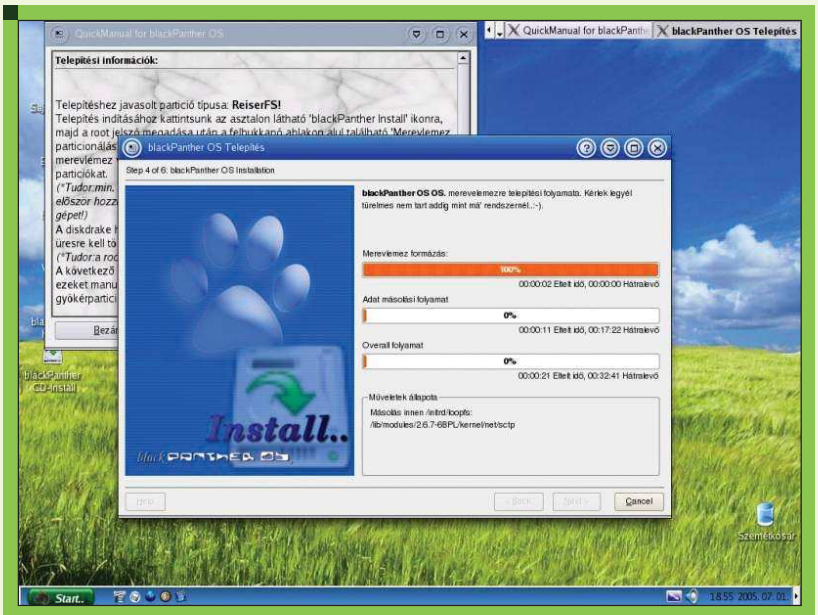
Horváth Ernő
ernohorvath@gmail.com

24 éves, műszaki informatikus. Három évvel ezelőtt ismerkedett meg komolyabban a Linux rendszerekkel és emellett érdeklődik még a robotika és a biztonságtechnika iránt is. Ha lenne szabadideje sokat kirándulna, biciklizne és filmeket nézne.

blackPanther OS – kicsi, fekete, a miénk...

Talán kevesen tudják, hogy mi magyarok immár három Linux terjesztéssel büszkélkedhetünk. Ezek egyike, a blackPanther OS, amely még soha nem szerepelt a Linuxvilág hasábjain. A hiányosságot ezennel pótoljuk.

A ***blackPanther OS***, amely „érkezési sorrendben" a második magyar ***Linux*** terjesztés volt, jelenleg a 4.2-es verziójánál tart. Amikor az első magyar ***Linux*** változat megjelent, sokan csóválták a fejüket, hogy miért, és ugyan minek kell egy ilyen kis országnak külön utakon járnia, és saját ***Linux*** disztribúciót fejlesztenie, ahelyett, hogy bekapcsolódna valamelyik nagy nemzetközi projektbe. Aztán sokan elkezdték használni a magyar ***Linuxot*** és a moraj elült. Amikor azonban a ***blackPanther OS*** megjelent, ismét lehetett hallani olyan hangokat, hogy minek még egy magyar ***Linux***. Ha nem lenne magyar ***Linux***, akkor valószínűleg az lenne a baj...
Egyetlen összeállítás se lehet jó mindenkinek, de éppen a választás lehetősége adja a ***Linux*** szépségét. Aki minden egyes beállítást maga szeretne elvégezni az ebben a világban ugyanúgy megtalálja a neki szánt terjesztést, mint az, aki 10 perc alatt telepíthető rendszerre vágyik. Ami tehát a hitvitákat illeti, éljen a pluralizmus!



■ ***1. ábra*** Megkezdődött a rendszer telepítése

Beszerzés, telepítés

A telepítő *CD*-t a ➲ ***ftp://ftp.blackpanther.hu/blackPanther.OS/All/v4.2(route)/*** helyről tölthetjük le. A blackPanther 4.x különlegessége, hogy a telepítőlemez egyben ***Live CD*** is, tehát úgy is kipróbálhatjuk ezt a terjesztést, hogy nem telepítjük. Egyszerűen csak be kell bootolnunk a *CD*-ről, és máris kapunk egy teljes, működő rendszert. Bootolás közben az ***F2*** gombot megnyomva figyelhetjük, hogy a rendszer éppen milyen feladatokat hajt végre. Elindulás után automatikusan beléptet bennünket a rendszer ***bpuser*** felhasználói névvel. Kapunk egy részletes magyarázatot is arról, hogyan telepíthetjük a ***blackPanther***-t a számítógépünkre. Mielőtt azonban ezt megtennénk, nézzük meg, mi is található pontosan az első lemezen.

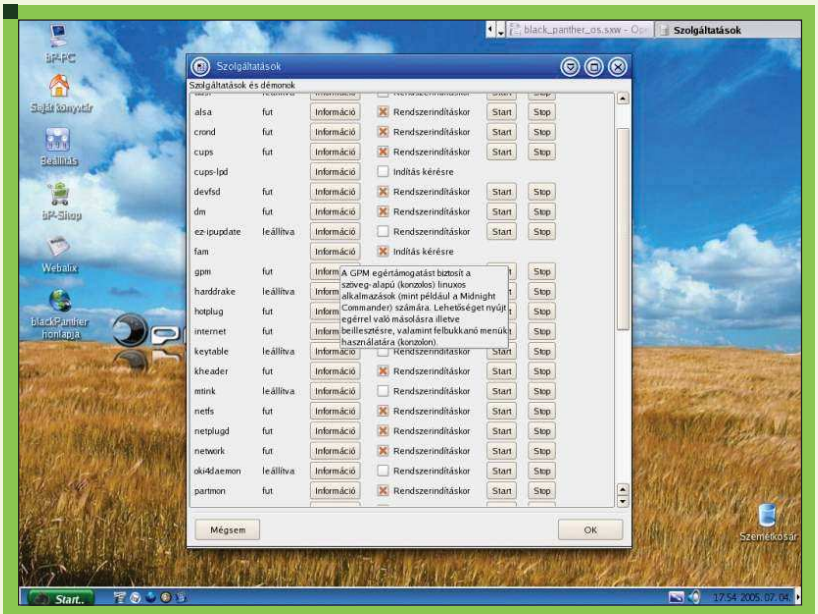
Mit tartogat számunkra a fekete párduc?

Az első benyomás igencsak pozitív. Szép zöld búzatábla a háttér, az ablakok árnyaltak, bár az ablakkeretek kicsit nagyok, a rajta található gombok szépek, és animáltak. Aztán hirtelen bebarnul a búzatábla, és én csak nézek, hogy mit rontottam el. Később rájövök, hogy semmit, a háttérkép periodikusan változik. Sajnos az ablakok mögötti árnyak az ablak mozgatása közben nem jelennek meg. Ez egyértelműen az ablakkezelő hibája.
De lássuk, mit is kapunk a ***blackPanther OS 4.2***-vel. Először is a ***Start...*** menüben találunk egy ***CrossOver Office*** rendszert. Ez kereskedelmi termék, így a disztribúció csak egy 30 napos próbaverziót tartalmaz belőle. A ***CrossOver Office*** segítségével ***Windows*** alá írt irodai alkalmazásokat tudunk használni ***Linux*** platform alatt.
Meglepetésként ért, hogy a ***blackPanther*** már a 2.0-ás ***OpenOffice.org***-gal készült, pontosabban a ***1.9.107***-es magyar nyelvű build-et tartalmazza.

A felhasználói programok terén amúgy nagy meglepetés nem ért: a legismertebb programokat (***Mozilla Firefox***, ***Mozilla Thunderbird***, ***Evolution***, ***Mplayer***, ***XMMS***, ***Gimp 2.0***, ***K3b***, ***Krusader***) megtalálhatjuk az első *CD*-n. Bár sok választási lehetőségünk nincs azzal kapcsolatban, hogy milyen feladatra melyik szoftvert akarjuk használni, mindegyik típusból a talán legismertebb rendelkezésünkre áll. A többi *CD*-n, illetve a csomagokat netről letöltve azonban kedvünkre válogathatunk zenelejátszók, és böngészők között.
Még a telepítés előtt gyorsan ki is próbáltam az operációs rendszert. Pendrive-ból többfélét is gond nélkül felismert, de az ***USB***-s külső merevlemezemet nem. Az ***MP3*** lejátszás sem sikerült elsőre, mert a hangerők alapból 0-ra voltak állítva, de ez csak apróság. Nagyon tetszett az a megoldás, hogy az ***XMMS*** bizonyos információkat ideiglenesen kitett a képernyő aljára zöld színnel. Amikor például az egyik szám véget ér, akkor megjeleníti a következő adatait.

A Párduc-lak kialakítása

Amennyiben nem zártuk be a telepítést segítő ablakot, akkor ennek segítségével végezzük el a telepítést. Először is kattintsunk az asztalon található ***blackPanther CD-Install*** ikonra, és gépeljük be a root felhasználó jelszavát, ami alapértelmezésben ***root***. Válasszuk ki a kívánt nyelvet, és szükség esetén végezzük el a particionálást a ***DiskDrake*** segítségével. Természetesen a telepítés előtt egy másik program segítségével is megtehetjük ezt, illetve meg is formázhatjuk a partíciókat. Aki nem nagyon ismeri a ***DiskDrake***-et, annak ezt a megoldást javaslom.
Ha már előre elkészítettük a partíciókat a ***blackPanther*** számára, akkor a ***DiskDrake***-en belül a ***Leválasztás*** gombbal válasszuk le őket, mivel a rendszer valószínűleg automatikusan felcsatolt mindent, amit felismert. A megfelelő csatolási pontokat a ***Csatolási pont*** menüben állíthatjuk be.
Ha már van egy szabad partíciónk a rendszer számára, de nincsen megformázva, akkor először váltsunk át szakértői módba, itt válasszuk ki a gyökérpartíció típusát (javasolt a ***ReiserFs)***, majd formázzuk meg. Ha már van már valamilyen Linux a gépünkön – ahogyan az enyémen is volt – akkor nem kell feltétlenül egy másik swap partíciót létrehoznunk. A két ***Linux*** rendszer nyugodtan használhatja ugyanazt a csereterületet, hiszen úgysem futnak egyszerre a gépünkön.



■ ***2. ábra*** A szolgáltatások beállítása sem nehéz feladat

A következő lépésben kijelölhetjük, hogy akkor melyik partíciókat is szeretnénk felhasználni a merevlemezünkön a ***blackPanther OS*** számára. Itt megadhatjuk még a */home*, */var* könyvtáraknak szánt partíciókat is.
Még a rendszer egyszer rákérdez, hogy akkor biztosan ezekkel a beállításokkal szeretnénk-e telepíteni az operációs rendszert, és elindul a telepítés (*1. ábra*), ami nekem 23 percet vett igénybe egy 1,5 ***GHz***-es ***Centrino*** processzorral szerelt notebookon.
Utána be kell állítani, hogy a rendszert betöltő LILO a Master Boot Recordba, vagy csak a saját partícióra települjön, ha már van valamilyen boot-managerünk, akkor célszerű azt meghagyni, és az új ***blackPanther OS***-t felvenni bele, ha még nincs ilyen, akkor mindenképpen azt válasszuk, hogy a ***LILO*** a ***Master Boot Recordba*** telepítse magát.
Amennyiben már telepítve van a ***GRUB***, akkor a telepítés után a ***blackPanther OS*** nem fog szerepelni a menüben. Ilyenkor a */boot/grub/menu.lst* fájlban megtalálhatjuk a ***GRUB*** menüjét, ezt egészítsük ki a következő három sorral:

```
title blackPanther OS
kernel (hd0,XX-1)/boot/vmlinuz
➥root=/dev/hdaXX
initrd (hd0,XX-1)/boot/
➥initrd-2.6.7-6BPL.img
```

Természetesen `XX` helyére a megfelelő partíció számát írjuk, `XX-1` helyére pedig értelemszerűen ***XX-1*** kivonás eredményét helyettesítsük be. Ezután adjuk ki a

```
grub-install /dev/hda
```

parancsot, és máris bootolhatjuk a ***blackPanther OS*** rendszerünket.
A telepítés utolsó szakaszában adjuk meg az új root jelszót, majd kattintsunk a Jelszó írása gombra. Utána felvehetünk még tetszőleges számú felhasználót, és indítsuk újra a frissen telepített rendszerünket. Az újraindítás ugyanaz a kép fog fogadni minket, mint a ***Live CD*** indítása után, ezen ne lepődjünk meg, még el kell végezni egy kis módosítást ahhoz, hogy a feltelepített rendszerünk teljes értékű legyen. A telepítést segítő ablakban keressük meg az utolsó előtti lépését a telepítésnek, innen megtudhatjuk, hogy ahhoz, hogy egy működő rendszert kapjunk a ***Takarítás*** gombra kell kattintanunk. Ezt a műveletet a rendszer gyorsan el is végzi, majd pedig kszedődhet a beállítás.

Szelídítsünk párducot!

A takarítás után a rendszer automatikusan felajánlja a rendszer beállításait, melyeket természetesen később is beállíthatunk a ***Rendszer/Beállítás/blackPantherOS Control Center*** menüpontban. Először is a futó szolgáltatásokat van lehetőségünk beállítani (*2. ábra*).

A szolgáltatások beállítását egyszerűen el tudjuk végezni, minden egyes szolgáltatás mellett az ***Információ*** gombra kattintva megtudhatjuk, hogy a szolgáltatás mire jó, a ***Start/Stop*** gombbal lehetőségünk van leállítani, illetve a ***Rendszerindításkor*** négyzet mellett lehetőségünk van arra, hogy az adott szolgáltatást minden rendszerindításkor elindítsuk-e vagy sem.

A következő lépés a hardverfelismerő, mely a számítógépünk összetevőit próbálja meg felkutatni. Ha valami nem működik megfelelően, akkor a baloldalon válasszuk ki az eszközt, és a jobb oldalon megjelennek az eszköz paraméterei. Itt alul található a ***Modul beállítása*** gomb, melyre kattintva elérhetjük, hogy az adott eszközt beállítsuk. Igaz én itt arra számítottam, hogy ha a merevlemezt akarom beállítani, akkor nem egy particionáló program ugrik fel, bár mégis ez történt. A beállítás sok helyen nem érthető, sok helyen pedig túl bonyolult dolgokat kérdez a rendszer, ezért szerintem egyelőre ne változtassunk itt semmin, hanem haladjunk tovább.

A Rendszerindítás beállításai pontot én kihagytam, mivel a rendszert egy már létező ***GRUB***-bal töltöttem be. Amennyiben viszont a ***blackPanther OS LILO***-ját használjuk, akkor mindenképpen érdemes itt beállítani, hogy egyéb operációs rendszereket is el tudjunk indítani. A következő ablakban lehetőségünk van megváltoztatni a root felhasználó jelszavát, amennyiben ezt még nem tettük meg, akkor most mindenképpen fordítsunk erre is figyelmet, a legjobb ha a jelszavunk minimum 8 karakter, és nem csak betűket, és számokat, hanem speciális karaktereket is tartalmaz, például `#bJi8!Hj` egy jó jelszó.

Tovább, tovább, tovább...

Most hogy már áll a rendszerem nem hagyott nyugodni, hogy hogyan oldjam meg a külső ***USB***-s merevlemezemet, illetve szerettem volna bekonfigurálni a hálózatot, hogy letölthessem a többi csomagot az internetről.

Az első meglepetés akkor ért, amikor bedugtam a hálózati kábelt a csatlakozóba. Nálam a ***192.168.1.1***-es címen egy előre beállított ***ADSL Router*** csücsül, ezért gondoltam, hogy szépen beállítom majd, és akkor menni fog a net.

3. ábra A telepítendő szoftverek letöltése folyamatban van

Hát nem így lett, bedugtam a csatlakozót, bebootoltam a rendszert, és a ***DHCP*** szerver tette a dolgát. Pillanatok alatt volt netem.

A hálózat tehát sikerült, bár sokat nem tettem az érdekében, úgyhogy már csak az ***USB***-s merevlemezem volt hátra. A telepítés során egyszer már találkoztam a ***Hardware felismerővel*** úgy gondoltam, hogy segíthet majd most is. Meg is találtam a ***Start..*** menü, ***Rendszer/Beállítás/Hardware/HardDrake*** menüpontban található a ***Hardware felismerőt***. A Lemezek között meg is találtam a hiányzó `sdb` eszközömet, az alul található Beállítóprogram indítása után pedig könnyen sikerült felcsatlakoztatni az eszközt a számomra tetsző helyre.

Ez még mindig nem oldotta meg teljesen a problémámat, mert most meg kézzel kellett csatolgatni az eszközt, így ha például az eszköz az `sdb1`, akkor a következő parancs root jogokkal való kiadása segít a problémán:

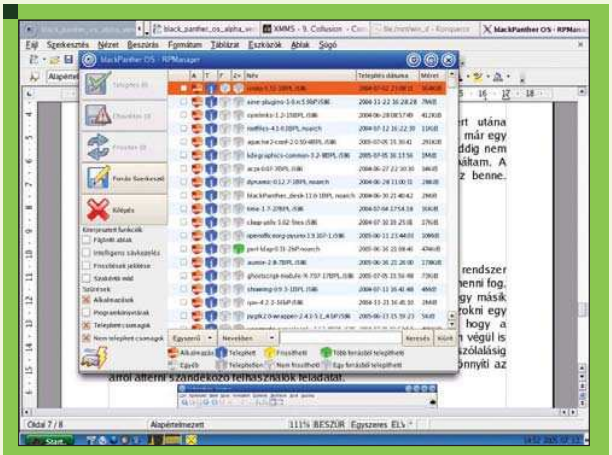
```
drakupdate_fstab --auto --add
➥ /dev/sdb1
```

Miután a net is megy, gondoltam, hogy kipróbálom a csomagkezelőt, hiszen egy rendszernek ez az egyik legfontosabb része. ***Start..*** menü ***Rendszer/Beállítás/Csomagoló/Szoftvertelepítés*** és nem történt semmi, látszott, hogy megpróbál elindulni a program, de nem sikerült neki. Gyorsan rá is jöttem a hibára, mert furcsálltam, hogy nem kérte a rendszer az ilyenkor kötelező root jelszót. Ellenben ha a ***blackPanther OS Control Centeren*** keresztül hívjuk meg a programokat akkor azok működnek. Innen már elérhető a Szoftvertelepítés menüpont. Hívjuk meg tehát ezt, és próbáljuk ki (*3. ábra*).

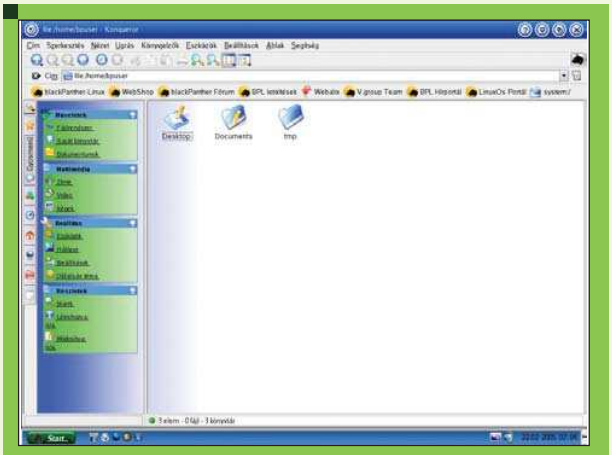
Első látásra nekem túl egyszerűnek tűnt a felülete. Mint minden esetben célszerű most is először a már frissíthető csomagokat feltelepíteni, mielőtt bármilyen más új csomagot telepítenénk. A frissíthető csomagokat a legördülő menü Összes csomag, frissítés elérhetősége alapján a listából ki tudjuk választani, hogy az összes frissíthető csomagot frissítsük. Ha ezzel készen vagyunk, akkor kedvünkre telepíthetünk új szoftvereket.

A telepítő figyel a függőségekre is, vagyis ha a telepíteni kívánt csomag másik csomagokra épül, akkor azokat is telepíti. Sajnos néhány esetben nem tudom, hogy miért, de nem sikerült a programhoz kapcsolódó ***lib**** csomagokat megtalálni, így bizonyos programokat nem tudtam feltelepíteni, meg lehetett volna azt is tenni, hogy ilyenkor a ***Mandrake (Mandriva)*** adott csomagjait telepítem helyette, hiszen a ***blackPanther OS Mandriva*** kompatibilis.

A telepítésnél viszont célszerű odafigyelni arra, hogy milyen csomagokat telepítünk, bár vannak ***bPL*** és ***mdk*** végű csomagok is, ez különösebb problémát nem jelent mivel a fejlesztők valószínűleg törekednek a kompatibilitásra. Persze ez nem azt jelenti, hogy a ***Mandrake FTP***-ről letöltött csomagokat össze-vissza telepíthetjük, mert a végeredmény akár az is lehet, hogy végül ***Mandriva*** rendszerünk



■ *4. ábra* Munkában az új csomagkezelő



■ *4. ábra* A Saját könyvtár megjelenése is megkönnyíti az áttérést

lesz. A forrásokban található csomagokat nyugodtan felhasználhatjuk vagy ha rendelkezünk némi jártassággal és csomagismerettel akár külső csomagokat is használhatunk.
A csomagkezelő nekem valahogy nem esett kézre, ezért utána érdeklődtem annak, hogy mi a helyzet vele, és kiderült, hogy létezik már egy teljesen új, de egyelőre csak tesztverzió belőle, a neve ***RPManager***, természetesen addig nem nyugodtam, amíg meg nem sikerült szereznem, és ki nem próbáltam. A ***blackPanther OS 5.x***-es verziójában már ez a csomagkezelő lesz benne. Felülete sokkal szebb, és átláthatóbb (*4. ábra*), mint az előzőnek, de még meg kell belőle várni a végleges verziót.

Szubjektív...

Úgy álltam neki a szoftver tesztelésének, hogy ***Linux*** és ***Linux*** rendszer között nem lehet olyan nagy különbség, úgyhogy minden pikk-pakk menni fog. Közben rájöttem, hogy azért ez korántsem így van. Közel két éve egy másik disztribúciót használok, ezért első látásra kicsit furcsa volt, hozzászokni egy másik rendszerhez. Már a ***Live CD*** indítása közben feltűnt, hogy a ***blackPanther OS*** csapat sokat ad a szoftver külső megjelenésére, hiszen végül is ez az amit a felhasználók először látnak. A munkaterület megszólalásig hasonlít egy Windows operációs rendszer munkaterületére (*5. ábra*), ami nyilván megkönnyítheti az arról áttérni szándékozó felhasználók feladatát. Nagyon tetszik, hogy a telepítő ***CD*** egyben ***Live CD*** is, így először lehetőségünk van kipróbálni a rendszert, és csak ha már megszoktuk, tetszik, akkor kell telepítenünk, pontosan ebből kifolyólag viszont elég kevés program fért el az első ***CD***-n, sajnos vagy nem sajnos a ***Cross Over Office*** demója elég sok helyet elfoglal teljes értékű, ingyenes programok elől. Meglepetésemre az ***OpenOffice 2.0***-ás változata van rajta a CD-n, ezzel ellentétben sajnos például az ***Acrobat Reader***-nek csak az 5.0-ás verzióját tartalmazza a disztribúció, gondolom ennek is vannak okai.
Új felhasználókat könnyen hozzávehetünk a rendszerhez, de ezeknek a felhasználóknak az alapértelmezett munkaasztaluk teljesen másképpen fog kinézni mint a ***bpuser*** felhasználónak. Az, hogy a rendszer bootolás után rögtön a ***bpuser*** felhasználót jelentkeztette be engem egy kicsit zavart, de van benne ráció, hiszen aki most ismerkedik az operációs rendszerrel, annak így nem kell a belépéssel bajlódnia. Habár jó lenne megszokni a felhasználóknak azt, hogy mindig azonosítaniuk kell magukat a számítógép számára. Picit zavart engem az is, hogy a ***Krusader*** alapértelmezett képnézegetője a ***Gimp2***, természetesen van ennél kisebb képnézegető is rendszerben, nem értem, hogy miért nem az az alapbeállítás. Mondjuk át lehet állítani könnyen, csak nem biztos, hogy egy kezdő felhasználó tudja, hogy mit kell ilyenkor tenni, márpedig egy kép megtekintése miatt a ***Gimpet*** elindítani kicsit luxus. Az ***USB***-s eszközök csatlakoztatásával is volt még egy kis problémán, mert ha ellenkező sorrendben dugtam be ***Pendrive***-ot, és az ***USB***-s külső merevlemezemet, akkor a rendszer az utóbbit nem csatolta fel megfelelően.
Ez engem egy kicsit zavart. De nem szabad elfelejteni, hogy több napot vett igénybe, amíg a jelenlegi ***Linux*** rendszeremet is úgy összeállítottam, hogy minden stimmeljen benne, a ***WLan*** kártya meg még egy külön kör volt. Természetesen jó ***Linuxhoz*** mérten azóta szinte semmit sem kellett rajta átállítanom, és megbízhatóan működik. Pontosan ezért egy kis kutatómunkával ***blackPanther OS*** alatt is szerintem mindent sikerülne beállítanom, de ez túlmutatna a cikk terjedelmén, és mivel gondolom nem mindenkinek vannak ilyen gondjai, ezért nem untatnám az olvasókat feleslegesen ezzel.
A ***blackPanther OS*** számomra összességében kellemes élmény volt. Érzésem szerint nagyszerűen használható desktop rendszerként, szinte minden általános felmerülő feladatra alkalmazható csak az első ***CD*** telepítése után is vagy mint ***Live CD***, de vállalati környezetben való alkalmazásával én inkább megvárnám az 5.0-ás verziót, hiszen a 4.2 csak a 4.0-ás frissített változata, míg a következő verzióban már sokkal több minden lesz benne, például az új csomagkezelő amely nekem sokkal jobban kézre áll. Remélem, hogy hamarosan az új rendszert is bemutathatjuk itt a Linuxvilág hasábjain.

Horváth Ernő
(ernohorvath@gmail.com)

Ide nekem az internetet – wget

A szélessávú netkapcsolatok elterjedésének hála, ma már ezerféle anyaghoz – szoftverekhez, dokumentációkhoz, képekhez, videókhoz – hozzájuthatunk, csak le kell tölteni őket. Böngészőnkben általában elég egy kattintás és a kiválasztott anyag máris töltődik le gépünkre. Ez azonban csak igen korlátozott lehetőségeket nyújt számunkra.

Ha egy valóban kezes segítőtársat szeretnénk letöltéseinkhez, használjuk a ***wget***-et. Nagyon hatékony segédeszközünk lehet hálózatról való letöltésre ***ftp*** és ***http*** (vagy ***https***) protokollokon keresztül. Régi segítőtársunk ő, bár lehet, hogy a szekrény mélyén lapult, így vegyük elő, töröljük le róla a port és ismerkedjünk meg vele!
A ***wget*** minden ***Linux*** és ***BSD*** terjesztésnek része, de ***MacOS X***-en és ***Windowson*** is használhatjuk, sőt létezik ***VMS***-re írt változata is.
A ***wget*** karakteres felületen vagy terminálban használható, tehát ismernünk kell a kapcsolókat, de mint látni fogjuk, ez igen hasznosnak fog bizonyulni mindennapos munkánk vagy egy-egy komolyabb letöltés során.

A wget képességeinek áttekintése

Az internet alapvetően nem megbízható közeg, így ha nagyobb anyagot, vagy sok kicsit kell letöltenünk, gyakran kell bosszankodnunk a folyamat megszakadása miatt. Ha pedig egy honlapot mentenénk le minden kellékével együtt, néha a böngészőnk nem elég intelligensen oldja ezt meg ahhoz, hogy a tükrözött helyet lokálisan is jól tudjuk majd használni. A ***wget*** minderre egyszerű megoldást nyújt. Hogy lássuk, mennyire hatékony eszközről van szó, tekintsük át röviden a program fontosabb tulajdonságait, képességeit.

- Szinte bármilyen ****nix*** környezetben megtalálható, vagy lefordítható, sőt szabadon használhatjuk Windows vagy ***MacOS X*** alatt is.
- Képes a háttérben dolgozni, így csak akkor kell odafigyelünk rá oda, amikor szeretnénk, akár ki is jelentkezhetünk.
- Lassú, vagy nem megfelelő kapcsolat esetén egy bizonyos határon belül addig próbálkozik, míg a letöltés jó nem lesz.
- Tudja folytatni a megszakadt letöltéseket (***ftp*** kapcsolat esetén persze csak akkor, ha ezt a kiszolgáló is támogatja).
- Proxy kiszolgálóval is használhatjuk a letöltés gyorsítására, a hálózati terhelés csökkentésére.
- Teljes könyvtárrendszereket letölthetünk vele a távoli gépről.
- Letöltési célként megadhatunk fájltípusokat, vagy meghatározhatjuk a kihagyni kívánt csoportot is.
- Konvertáltathatjuk vele a linkeket a helyi gépre, azaz automatikusan kialakíthatjuk az „offline böngészésre” alkalmas anyagot.
- Megszabhatjuk, hogy a linkek követése során mennyire „kalandozhat el” más helyekre, honlapokra.
- A letöltés során nyomon követhetjük a folyamat menetét, de mindezt át is irányíthatjuk egy fájlba, ahol később tanulmányozhatjuk.
- Minden beállítást elmenthetünk egy konfigurációs fájlba, így nem kell minden egyes alkalommal megadnunk őket. Ezeket egyedileg felül is bírálhatjuk, ha éppen erre van szükségünk.
- Végül, de nem utolsó sorban teljesen szabad, nyílt forráskódú szoftverről van szó, ami jelentős érv a ***wget*** mellett a rengeteg érdekes, de a legkevésbé sem olcsó letöltésvezérlők korában.

Azt hiszem, ennyi információ épp elég annak eldöntéséhez, hogy szeretnénk-e közelebbről megismerkedni a ***wget***-tel...
Mivel valószínűleg megtaláljuk kedvenc terjesztésünkben, ezért letölteni sem szükséges. Ha mégis szeretnék a legfrissebb változathoz hozzájutni (ez jelenleg, 2005. július elején az 1.9.1-es), akkor a ➲ http://www.gnu.org/order/ftp.html oldalon található ftp szerverek listájáról válasszunk egyet.

Alapvető tudnivalók

Ennyi bevezető után vágjunk a közepébe! A használat nagyon egyszerű:

```
wget [opciók] [URL]
```

Amennyiben az opciókat kezdésként elhagyjuk, a kérdés az marad, mi is pontosan az „URL”? Ez egy hálózati erőforrás azonosítója, ami lehet dokumentum, kép, hang vagy bármilyen más fájl. Az ***URL*** rövidítés az ***„Uniform Resource Locator”***-t takarja, amit talán „Egységes erőforrás-azonosítónak” magyarosíthatnánk, az internetes erőforrások szabványosított címe, tartalmazza a protokollt (például HTTP), a használt portszámot, a gazdagép nevét és az erőforrás elérési útját a gépen belül, esetleg még az a név és jelszó-párost is, ami az eléréshez szükséges. Nézzünk egy példát!
A ***http://hu.wikipedia.org*** például egy nagyon egyszerű ***URL***, tulajdonképpen csak egy protokollt, a HTTP-t tartalmazza és az internetcímet. (A magyar ***Wikipédiát*** a ***hu.wikipedia.org*** címen találjuk, ha az információkra éhes olvasó esetleg még nem ismerné, akkor sürgősen keresse fel.)
Az ***ftp://pisti:pst32z@pistinet.hu/fajlok/doksi.zip*** már bonyolultabb. Itt már megadtuk a nevet és a jelszót is, ami az ftp eléréshez szükséges (név: ***pisti***, jelszó: ***pst32z***) és a teljes elérési utat a ***„pistinet.hu”*** szerveren belül.
Megadom a ***wget*** számára értelmezhető ***URL*** mintát általános módon is. Ez a következő:

```
protokoll://név:jelszó@gazdagép
➥:port/könyvtár/fájl
```

A `protokoll` lehet `http`, `https` vagy `ftp`. Ennek megadása kötelező. A név és jelszó kettősponttal van elválasztva, amit akkor kell csak megadnunk, ha azonosítanunk kell magunkat. Névtelenül is használható (anonymous) ***ftp*** szerverek esetén ezt a ***wget*** elvégzi helyettünk. A név és jelszó megadását kukac jellel zárjuk le, ezután megadjuk a gazdagép domén nevét vagy ***IP*** címét, ezután kettősponttal elválasztva a portszámot, ha az nem a szabványos (a szabványos portszámok: 80-as ***http*** esetén, ***443 https***-nél és ***21 ftp*** szolgáltatásnál) majd az elérendő erőforrás útvonala következik.
Most már tudjuk, hogy az ***URL***-ek meglehetősen bonyolultak is lehetnek, viszont a jó hír az, hogy szerencsére ritkán kell a szabványostól eltérő porton lévő szerverre felcsatlakoznunk, publikus szervereknél pedig a nevet és jelszót sem kell megadnunk. Persze amikor egy hosszú ***URL***-t kell megadnunk, könnyen eltéveszthetjük. Ezért érdemes a címet a böngészőből kimásolni, hiszen az esetek többségében valószínűleg az interneten bukkanunk rá egy-egy izgalmas helyre, ahonnan azonnal le szeretnénk valami tölteni.
A ***wget*** igazán akkor kezd érdekessé válni, ha megismerkedünk az opciókkal. Ezek segítségével szabályozhatjuk, hogyan töltsön le, mit vegyen figyelembe és mit nem és miként jelenítse meg a letöltés folyamatát. Ne feledkezzünk meg arról sem, hogy a kis- és nagybetűk bizony számítanak az opciók megadásakor, így például a „V” és „v” egészen mást jelentenek.

Alapvetően három lehetőségünk van:

- `v`: Ez a verziószámot írja ki, ami hasznos lehet megállapítani, hogy vajon a legújabb verziót használjuk-e vagy pedig letöltőnk már frissítésre szorul
- `h`: A ***wget*** helpjét (magyarosan a súgóját) kapjuk meg
- `b`: Megadásakor a ***wget*** a háttérben fog futni, mindenféle visszajelzés nélkül. Ez nagyon hasznos lehet, ha a terminálban tovább szeretnénk dolgozni. Ami normálisan a képernyőre kerülne, az most egy ***„wget.log”*** nevű fájlba íródik bele. Így a letöltés után végigkövethetjük, mi történt, felderíthetjük az esetleges hibákat is.

Ha be akarom mutatni a további lehetőségeket is, akkor az opciókat csoportosítanom kell, mert nagyon sok van, hiszen ahogy említettem, a ***wget*** nagyon rugalmas szoftver, szinte minden, a letöltéssel kapcsolatos óhajunkat képes teljesíteni.

Én négy csoportra osztottam fel a funkciók listáját, bár nyilván lehetne máshogyan is:

- A ki- és bemenet szabályozása (megjelenítési opciók, naplózás, ***URL*** listafájl megadása)
- A letöltés szabályozása (proxy szerver magadásától a letöltés folytatásának lehetőségén át a letöltés sebességéig sok minden belefér ide)
- ***HTTP*** opciók (a ***http*** protokollon keresztüli letöltés szabályozása)
- ***FTP*** opciók (az ***ftp*** protokollon keresztüli letöltés szabályozása)

A ***wget*** opciók általában az url megadása előtt következnek, de semmi sem kötelez erre bennünket – ha úgy jobban tetszik, egyeseket átcsoportosíthatunk utána is. A következőkben tehát az opciókkal foglalkozom, de nem fogom mindet felsorolni, inkább csak bemutatni a ***wget*** lehetőségeit.

A ki- és bemenet szabályozása

```
--progress = tipus
```

A „típus” helyére írt szöveggel megadhatjuk, milyen legyen a letöltés folyamatának a kijelzése. Ha nem használjuk ezt a beállítást, akkor az alapértelmezett mintát fogjuk látni, ami jelenleg a `bar`. Ez a következőképpen fest:

```
32% [=====>                ]
➥14,968,576     118.56K/s
➥ETA 04:22
```

Az első szám a letöltött mennyiséget mutatja százalékban, ezután két kapcsos zárójel között (ez a `bar`, azaz „téglalap” egysoros ASCII formája) egy folytonosan növekvő nyilat láthatunk. A zárójel után a letöltött mennyiség bájtokban, majd az aktuális sebesség (kilobájt per másodpercben) és az `ETA` látható. Az `ETA` az ***„Estimated Time of Arrival”***, azaz a “becsült várakozási idő” rövidítése. Használhatjuk még `dot`-ot is, ennek további altípusai is vannak. Írhatjuk önmagában is, ami az alapértelme-

zett *„dot”* stílust fogja jelenteni, amikor is egy pont 1 kilobájtot fog jelenteni, minden sorba pedig 50 pontot ír ki. Ez nyilván a régi, lassú hálózatok maradványa, amikor még csak kis dolgokat tölthettünk le belátható időn belül, így ezzel kényelmesen nyomon követhettük a letöltés folyamatát. Egy szélessávú hálózaton ez elviselhetetlenül szétaprózza a kijelzést. Az „elavult” jelzőt ráakaszthatjuk a `dot:binary` típusra is, ami érdekes lehet, az a `dot:mega`, ahol nevével ellentétben nem egy megabájt egy pont, hanem 64K. Minden sorba 48 pontot ír, így egy sor 3 megabájtnak fog megfelelni:

```
wget –progress=dot:mega
➥ftp://ftp.uhulinux.hu/ISO/
➥uhu-linux-1.2-cd1.iso
```

Ez valahogy így fog kinézni *(1. kód)*
A `dot` stílus kiválóan alkalmas arra, hogy nyomon kövessük, az egyes egységekben (itt ez 3 megabájt) mekkora volt az átlagos letöltési sebesség.
A `bar`-ral ellentétben sokkal finomabb egységekben jelzi a letöltés folyamatát is. Ez főleg egyenetlen letöltési sebességnél jöhet jól nekünk.

```
-o logfile
```

A ***wget*** alapértelmezésben a képernyőre ír mindent. Ezt átirányíthatjuk egy fájlba ezzel az opcióval. A `-b` opciónál említettem hogy itt automatikusan a ***wget.log*** fájlba kerül a program kimenete. Az `-a` opció csak annyiban különbözik az `-o`-tól, hogy amennyiben létezik a megadott fájl, nem törli azt, hanem folytatja.
A ***wget***-et teljesen el is „hallgattathatjuk” a `-q` vagy `--quiet` opcióval. Ezzel ellentétben a `-v` vagy `--verbose` opció „szószátyárrá” teszi a ***wget***-et, ami egyébként az alapbeállítás is.
A kettő között van az `-nv` vagy `--non-verbose` opció, ami egy általában hallgatag, de a hibáknál „beszélő” szoftvert eredményez.

Előfordul, hogy több fájlt kell letöltenünk, amik nem nem egy helyen vannak (mert akkor a rekurzív letöltést alkalmazhatjuk) hanem az interneten szétszórva, több gazdagépen. Ilyenkor alkalmazhatjuk a `-i file` vagy `–input-file = file` opciót, amelynél egy megadott fájlból fogja venni a letöltendő elemeket.

A letöltés szabályozása

Ha megszakad a letöltés és a távoli szerver ezt engedi, a ***wget*** tudja folytatni azt, erre szolgál a -c opció. Amennyiben a hálózati kapcsolat szakad meg és nem a ***wget*** működése, ez az alapértelmezése, tehát automatikusan folytatni fogja a már megkezdett letöltést. Előfordulhat azonban, hogy kézileg állítjuk le a ***wget***-et vagy valamilyen más hiba történik, ekkor az opció megadása nélkül újból elkezdi letölteni a fájlt, meghagyva az eredetileg félbehagyottat is.
Nagyon hasznos funkció lehet a letöltés sebességét szabályozó `--limit-rate` opció. Ezzel beállíthatjuk, mekkora az a legnagyobb sebesség, amivel a ***wget*** letölthet. Így ha közben szeretnénk internetezni vagy a belső hálózaton lévő más felhasználóktól nem szeretnénk a teljes sávszélességet elvenni, ezzel szabályozhatjuk a nekünk tetsző sebességet.
Bájt, kilobájt és megabájt per másodperc mértékegységeket használhatunk. Tehát a

```
wget –limit-rate=35k
➥ftp://ftp.fsn.hu/pub/linux/
➥distributions/knoppix/
➥KNOPPIX_V3.9-2005-05-27-EN.iso
```

azt jelenti, hogy maximum 35 kilobájt/másodperc, azaz 280 kbit/s sávszélességgel fog menni a ***Knoppix*** disztribúció letöltése. Így ha például egy 512 kbit/s adsl vonalunk van, akkor a többi felhasználónak vagy más letöltéshez még mindig marad 232 kbit/s.

1. kód

```
   0K ........ ........ ........ ........ ........ ........  0%  112.62 KB/s
3072K ........ ........ ........ ........ ........ ........  1%  114.58 KB/s
6144K ........ ........ .....
```

Ha ***FTP*** kiszolgálóról töltünk le, használhatjuk a helyettesítő karaktereket is:

```
wget ftp://ftp.szerverem.hu/*.zip
```

De több könyvtár tartalmát is letölthetjük a `-r` (rekurzív letöltés) opcióval. Ez nemcsak FTP-n, hanem weboldalakon is alkalmazható.
Ha ugyanis enélkül akarunk egy honlapot letölteni:

```
wget http://www.linuxvilag.hu
```

akkor csupán egy ***index.html***-t fog letölteni, ugyanis ezt kapta vissza a ***wget*** a ***Linuxvilág*** web szerverétől. A honlap további részeinek letöltéséhez a

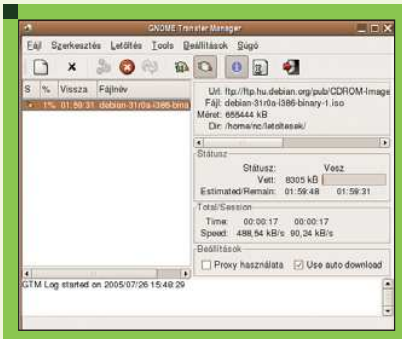
```
wget -r
➥http://www.linuxvilag.hu
```

formát kell használnunk.

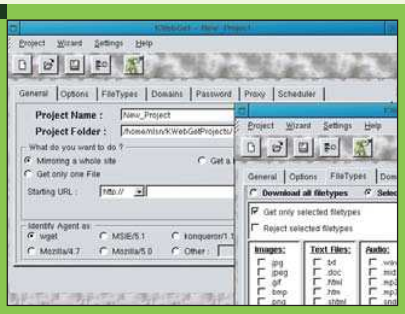
Ez már egy ***www.linuxvilag.hu*** nevű könyvtárat is létrehoz és ebbe menti le a honlap tartalmát. Web oldalakon ez az egyes linkeket is megvizsgálja, így a letöltés ezekre is kiterjed. Alapértelmezésben 5 szint mélységig fog letölteni, de ezt az `-l` (vagy `--level=`) opcióval megváltoztathatjuk. Például a `-l 2` megadásával két szint mélységig fog vizsgálódni. Végtelen mélységet a `-l 0` opcióval adhatunk meg, de ezzel nagyon vigyázzunk, mert lehet hogy bizonyos helyekről több gigabájt adatot fogunk letölteni.
Ha teljes tükrözést szeretnénk, ezt az -m opcióval érhetjük el (ekkor az `-r` és `-l` nem kellenek).
A weboldalak letöltésének további lehetőségeiről a következő fejezetben lesz szó.
Végül, érdemes megemlíteni a következő három opciót:

`-nc` (vagy `--no-clobber`)
`-N` (vagy `--timestamping`)
`-K` (vagy `--backup-converted`)

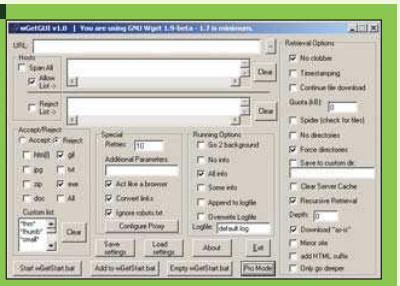
Ezek a rendszeres letöltésnél használhatók jól. Ha például egy webhelyet rendszeresen frissítünk a gépünkön, fontos a `-N` opciót használni, így csak a változásokat fogja letölteni. Ha ezt a `-K` -val együtt



■ *1. ábra* A Gnome Transfer Manager



■ *2. ábra* KwebGet



■ *3. ábra* wGetGUI

használjuk, akkor a régi fájlokról egy biztonsági mentést (backup) is készít, azaz átnevezi őket, hogy ne kerüljenek felülírásra. A „no clobber” opció (`-nc`) viszont megakadályozza, hogy a létező fájlokat felülírhassuk, így ezek le sem töltődnek. Ez akkor hasznos, ha mindig csak az új fájlokat szeretnénk egy helyről letölteni (és az új fájlok megnevezése is eltérő a régiekétől).

HTTP opciók, weboldalak letöltése

A weboldalak letöltéséhez számtalan lehetőséget, beállítást kínál nekünk a *wget*. Mint említettem, ha nem adunk meg a letöltésre vonatkozó opciót, akkor csak az index oldalt fogjuk visszakapni.
Rekurzív letöltés esetén (`-r` opció) viszont a megadott (vagy az alapértelmezett 5) szintig letölti a webtartalmat, követve a hivatkozásokat is (már amihez jogosultságunk van, természetesen *HTTP* protokoll esetén is megadhatunk bejelentkezési (login) nevet és jelszót akár az *URL*-ben, akár a *–http-user=user* és `--http-password=password` opciókat használva). A letöltött oldalak és webtartalom ezután böngészőben megnézhető... lenne, ha a hivatkozások a helyi gépre mutatnának. De alapértelmezésben mindent úgy töltünk le, ahogy a webhelyen található. Ekkor csak a relatív címzések fognak működni, az abszolútak nem. Persze lehet, hogy nem szeretnénk az eredeti tartalmat megváltoztatni. Viszont az is előfordulhat, hogy a weboldalon egyes linkek más helyekre is mutatnak, a *wget* így végül akár a szándékunkkal eltérő tartalmakat is letölthet vagy a letöltés a végtelenségig folytatódik.

Ha offline böngészésre szeretnénk használni a letöltött website tartalmat, mindenképpen konvertálni kell a hivatkozásokat a `-k` (vagy `--convert-links`) opcióval. Ekkor minden link a helyi gépre fog mutatni.
A *wget* szerencsénkre más tartományokba (domain) nem „kalandozik el”, azaz ha a letöltendő weboldalakon kívülre mutató link van, azt nem dolgozza fel. Tehát a

```
wget -r http://www.debian.org
```

csak a `debian.org` tartományra mutató linkeket fogja követni.
Ellenben ha a `-H` opciót is megadjuk, akkor minden linket követni fog, bárhová is hivatkozzon. A `-D` opcióval tudjuk korlátozni ezt a funkciót, megadva, mely tartományokra korlátozzuk a letöltést. Tehát a

```
wget -rH -Ddebian.org
➥http://www.debian.org
```

letölti a *debian.org* altartományai lévő tartalmat is, így a *packages.debian.org*-ot vagy a *lists.debian.org*-ot is.
Természetesen csak a legritkább esetben szeretnénk akkora tartalmakat letölteni, mint a *packages.debian.org*.
Ezért a nem kívánt helyek listáját megadhatjuk a `--exclude-domains` opcióval:

```
wget -rH -Ddebian.org
➥-exclude-domains
➥packages.debian.org,lists.
➥debian.org http://www.debian.org
```

Az is egy jó módszer a letöltés kordában tartására, ha csak a relatív linkeket engedjük követni a *wgetnek* a `-L` opcióval. A *www.debian.org* tipikusan ilyen jól megszerkesztett oldal, ahol relatív linkek mutatnak a tartományon belüli tartalomra.
Végül, a webes letöltések bemutatását fejezzük be egy nagyon hasznos lehetőséggel, a ***tükrözéssel (mirroring)***.
Ez természetesen a `-m` opcióval adható meg.
Egy webhely (vagy akár *FTP* hely) tükrözése során a *wget* rekurzív módon mindent letölt az adott tartományból (természetesen *HTTP* esetén a linkek követésével), ami a legutóbbi tükrözés óta megváltozott.

FTP opciók

FTP szerverről letöltésnél a leggyakoribb opciónk valószínűleg a megfelelő név és jelszó megadása lesz:

```
--ftp-user=user
--ftp-password=password
```

Ezek hiányában a *wget* *anonymousként* próbál belépni.
Használhatjuk a speciális behelyettesítő karaktereket is: a * vagy ? szimbólumokat.

```
wget ftp://szerverem.hu/*.jpg
```

Így természetesen csak a *jpeg* képeket fogja letölteni számunkra.
Általában passzív módban töltünk le *FTP* szerverekről. De előfordulhat, hogy csak az aktív mód engedélyezett, ekkor kell használnunk a `--no-passive-ftp` opciót.
Gyakran alkalmaznak szimbolikus linkeket az *FTP* szerverek könyvtárstruktúrájában. A *wget* alapértelmezés szerint ezeket nem követi, tehát gépünkre is csak a link kerül. Ha szeretnénk, hogy a szimbolikus linkek által

hivatkozott helyekről is letöltsön, alkalmaznunk kell a `--retr-symlink` opciót.

A wgetrc

A *wgetrc* egy nagyon hasznos beállítófájl, ahol a *wget* általános működését tudjuk szabályozni. Több helyen is megtalálhatjuk. A globális beállítások az */etc/wgetrc* fájlban találhatóak (egyes disztribúciókban ez az */usr/local/etc/wgetrc*). A felhasználótól függő beállításokhoz a saját könyvárunkban kell létrehoznunk egy *.wgetrc* fájlt. Ez felülbírálhatja az általános *wgetrc* beállításait.
Az egyes opciók megadása természetesen bármely *wgetrc*-t felülbírál. Ezek a beállítások addig érvényesek, míg a parancssorban felül nem bíráljuk őket.
A *wgetrc* szintaxisa nagyon egyszerű:

```
változó = érték
```

Minden egyes változó-érték párt új sorba kell írni.
Rengeteg lehetőségünk van a *wget* szabályozására. Ha belenézünk a globális *wgetrc*-be, láthatjuk, hogy egypár érték már előre be van állítva számunkra, amik így a *wget* alapértelmezései. Ha használunk proxy szervert, itt egyszer s mindenkorra megadhatjuk azt, elkerülve hogy minden letöltéskor be kelljen írnunk:

```
http_proxy = server1.ceg.hu
ftp_proxy = server2.ceg.hu
proxy_user = "janos"
proxy_password = "jelszavam"
```

A *http://ftp.gnu.org/pub/gnu/Manuals/wget-1.8.1/html_node/wget_27.html#SEC27* oldalon egy teljes listát kaphatunk a *wgetrc* parancsokról.

Grafikus felületek

A *wget* használatának megkönnyítésére grafikus felületek is készültek hozzá. A *Gnome Transfer Manager* (*1. ábra*) egyszerű, grafikus letöltésmenedzser a *Gnome* ablakkezelőhöz, és természetesen a *wget*-en alapul. Ennél részletesebb beállítási lehetőségeket kínál *Frank von Daak KwebGet*-je (➲*http://www.kpage.de/en/index.html*) (*2. ábra*).

A *wget* windowsos változatát is letölthetjük a *http://gnuwin32.sourceforge.net/packages/wget.htm* oldalról. Ezen alapul a *wGetGUI* (*http://www.jensroesner.de/wgetgui/*) is (*3. ábra*), amellyel nagyon részletesen szabályozhatjuk a *wget* számtalan lehetőségét.
Egyetlen cikkben természetesen lehetetlen bemutatni a *wget* minden tulajdonságát és szolgáltatását, tehát a sikeres felfedezésekhez bátran használjuk a *man wget* parancsot vagy a *www.gnu.org/software/wget/manual/* oldalon található részletes ismertetőt.

Molnár Norbert
34 éves, rendszergazdaként dolgozik, 5 éve foglalkozik Linuxszal. Főként a szabad szoftverek és a számítógépes biztonság érdekli. Budapesten él, hobbija – ha éppen nem egy újabb disztribúciót tanulmányoz és nem utazik – a csillagászat és a filozófia - lehetőleg jó vörösbor mellett.
(molnar.norbert@gmail.com)

FreeBSD – a szomszéd vár (12. rész)

A szomszéd szomszédai

Néha szükséges, hogy a gépünkre más operációs rendszer is telepítsünk, s itt elsősorban Linux rendszerre gondolok. Valami oknál fogva általában egy gépen a Linux mellé kerül egy FreeBSD, esetleg a FreeBSD mellé egy Linux. Ennek oka vagy az új rendszer megismerése, vagy a hiányzó funkciók kihasználása.

Ha már két operációs rendszer van egy számítógépen, akkor ésszerű a lehető legkevesebb eltérést beállítani, amely ***FreeBSD*** és ***Linux*** viszonylatában – bizonyos korlátokat betartva – megoldható. Érdemes úgy kialakítani a rendszert, hogy legyen egy közös ***home*** könyvtár, esetleg a ***swap*** partíció is legyen közös használatban, s egy ablakkezelő helyes működése sem árt. Ha a legtöbb program képes mind a két rendszer alatt megfelelően működni, akkor már semmi egyéb kívánságunk nem lehet.

A közös fájlrendszer

Fájlrendszer tekintetében ez a két rendszer néhol oly mértékben nem tud együttműködni egymással, hogy ilyet még a ***Linux*** és ***Windows*** viszonylatban is ritkán látni. Ez főleg a teljesen szabadon hozzáférhető fájlrendszer specifikációk esetén tűnik furcsának, ugyanis a ***FreeBSD*** egyedül az ***Ext2*** fájlrendszert írja-olvassa, az ***Ext3*** esetén sokszor olvasni sem képes, s gyakorlatilag csak a naplózás egyszerűsítésével képes írni. Ha mégis kezeli, akkor az alábbi beszédes üzenettel képes őrületbe kergetni a gyanútlan felhasználót:

```
$ mount_ext2fs /dev/ad0s4 /mnt/
mount_ext2fs: /dev/ad0s4: Invalid argument
$ dmesg
WARNING: mount of ad0s4 denied due to
➥unsupported optional features
```

Ez például csak annyit jelent, hogy az adott fájlrendszer nincs rendbe téve, érdemes lenne egy `fsck` futtatása:

```
$ fsck.ext3 /dev/ad0s4
e2fsck 1.38 (30-Jun-2005)
/: recovering journal
/: clean, 120706/555968 files, 765563/1110493
➥blocks
$ mount_ext2fs /dev/ad0s4 /mnt/
```

Ne bízzuk el magunkat túlságosan, ugyanis hosszabb ***FreeBSD*** alatti munka után – amikor a Linuxot választjuk a következő rendszerindításkor – szinte biztos, hogy elindul és valami hibát talál az `fsck`. Persze ez fordított esetben is így történik, ugyanis a ***Linux*** sem kezeli rendesen a ***FreeBSD*** „új” (vagyis lassan három éves) ***UFS2*** fájlrendszerét, így hosszas munka után itt is azt veszem észre a ***FreeBSD*** indulásakor, hogy a háttérben elindít egy `fsck` folyamatot is a hibák javítása ügyében. Ráadásul a ***SuSE 9.3*** „gyári” rendszermagja csak olvasni tudja az ***UFS2*** fájlrendszert, külön rendszermagot kell fordítani, ha írni is szeretnénk azt. Ezen túlmenően én az ***UFS2*** fájlrendszer javaslom közös nevezőnek, mert a ***Linux*** rendszermag ***UFS2*** kezelő része jobban van megírva (bár tudtommal nem ismeri/kezeli a ***SoftUpdate*** lehetőséget), mint a ***FreeBSD*** ***Ext3*** kezelő modulja.
Ezt még tetézi, hogy ha nem kézzel csatoljuk le a ***FreeBSD*** leállítása előtt a felcsatolt ***Ext2*** vagy ***Ext3*** fájlrendszereket, akkor ez a leálláskor problémát okoz. A ***FreeBSD*** rendszermagja ugyanis hibásan kezeli az említett fájlrendszerek gyorstárazását, s úgy érzékeli, hogy nem tudja kiírni

a lemezre az összes módosított blokkot. Ennek következménye az, hogy egyik fájlrendszert sem zárja le helyesen, és lecsatolás nélkül eldobja őket, s ez okból a következő indításkor egy *fsck* vár minket néhány teljesen hibátlan fájlrendszeren. Bosszantó és zavaró hiba, bár problémát vagy adatvesztést nem okoz. Kissé érdekes, hogy a világhálón található egyetlen *Linux+FreeBSD HowTo*-ban mindössze a következő sor található ezzel a – még az 5.4-es *FreeBSD*-ben is megtalálható – hibával kapcsolatban: *„The bug has been fixed in FreeBSD 3.x."*
További hiányosság, hogy az újabb *Linux* kiadások már *UTF-8* fájlrendszert használnak, míg a *FreeBSD* ezt nem képes használni, nincs felkészítve erre a feladatra. Érdemes a *Linux* rendszerben beállítani például az *ISO8859-2* kódlapot a fájlrendszerre is. Érdemes tehát egy *UFS2* fájlrendszerre létrehozni a */home* könyvtárat, amelyet többnyire mind a két rendszer jól kezel.

Telepítési sorrend

Gyakorlatilag teljesen mindegy, hogy melyik rendszer telepítjük fel először, majd ezek után melyikkel folytatjuk. Javaslom azonban először a *FreeBSD*-t feltelepíteni, mivel ennek az igényei a kevésbé rugalmasak, hiszen csak elsődleges partícióra tud feltelepülni, illetve indulni. A *Linux* rendszer már rugalmasabb lehet, főleg egy jól bejáratott *GRUB* használatával még a két rendszer indítása is sokkal szebb lehet.
A *GRUB* beállítása egyszerű, a *FreeBSD* teljesen azonos módon indul, mint egy *Windows*, vagyis a megfelelő állományba (*SuSE* esetén a */boot/grub/menu.lst*) kell a következő sorokat beszúrni (megfelelő adatokkal):

```
title FreeBSD
    root (hd0,2)
    chainloader +1

A LILO is teljesen hasonlóképpen állítható be:

other=/dev/hda3
    table=/dev/hda
    label=FreeBSD
```

Közös swap partíció

A két rendszer képes – hellyel-közzel közös – *swap* partíciót használni, ha *Linux* alatt minden induláskor lefuttatjuk az *mkswap* parancsot a megfelelő fájlrendszerre. A *FreeBSD* ezt megteszi a *swap* aktiválásakor (nincs is *mkswap* parancsa). Ha *Linux* alatt használjuk a „hibernálást", akkor a közös *swap* használatot nyugodtan elfelejthetjük, mivel a *Linux* rendszer ide menti le a memória tartalmát, mielőtt lekapcsolná a gépet: a *FreeBSD* pedig minden indulásakor lazán – különösebb üzenet és figyelmeztetés nélkül – felülírja ezeket az adatokat.

Felhasználók

A *passwd* és *shadow* (*FreeBSD* esetén a *master.passwd*) állományok különbözősége okán a felhasználókat külön-külön fel kell vennünk, lehetőleg azonos jelszóval és azonos *UID*/*GID* számokkal. Érdemes a megnevezéseket is közös nevezőre hozni, mivel a *FreeBSD sysinstall* mindenese a kitöltetlen csoport esetén létrehoz egy felhasználói névvel azonos nevű csoportot. Ha a */home* közös (és miért ne lenne az), akkor az egyik rendszer problémázni fog a már létező felhasználói könyvtár miatt.
Érdemes először *Linux* alatt létrehozni a felhasználókat, mivel itt több állomány kerül be a felhasználó könyvtárába kezdésképp, amelynek nagy részére szükség is van.
A *FreeBSD* már képes ezeket megfelelően használni, vagy figyelmen kívül hagyni.

Grafikus felület, KDE

A két rendszer grafikus felületét külön-külön beállíthatjuk, hiszen az alaprendszer külön helyen található. Igyekezzünk azonos felbontást és tulajdonságokat beállítani, különben azonos programokat használva a két rendszer esetén apró kis bosszantó hibák kerülnek napvilágra (kilógó ablakok, eltünedező ikonok, stb.).
A *KDE* (és *GNOME*) esetén különösen ügyeljünk arra, hogy a két operációs rendszer alá feltelepített programok azonos verziószámmal bírjanak, különben képesek apró hibákat okozni a konfigurációs állományokban, amelyeket kibogozni – ha nem is lehetetlen – de időrabló és hosszadalmas feladat (velem többször is előfordult, hogy inkább letöröltem és újra beállítottam az adott programot).
Sajnos a *Linux* terjesztők saját fejlesztései kismértékű problémát okozhatnak, ha belefutunk egy-egy ilyen programba. Például a modemes tárcsázáshoz a *SuSE* egy saját *Kinternet* nevű programot használ, amelyet nehézkes lenne *FreeBSD* alatt használni, így marad a *Kppp*, amelyet viszont *SuSE* alatt nehéz jól beállítani (egyszerűbb átmásolni a *FreeBSD* alatti beállítófájlokat).
Érdemes *FreeBSD* alatt kezdeni a beállításokat, mert nagyobb az esély arra, hogy *Linux* alatt is lesz ilyen program, mint fordítva (bár akadnak kivételek).
Ezzel véget ért a *FreeBSD* operációs rendszer néhol részletesre, néhol szűkre sikerült bemutatása. Sok sikert és örömet kívánok mindenkinek a szomszéd vár további felfedezéséhez.

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

A FreeBSD projekt honlapja: ➲ http://www.freebsd.org

A magyar FreeBSD honlap: ➲ http://www.freebsd.hu

A magyar BSD honlap: ➲ http://www.bsd.hu

A kézikönyv magyar fordítása

➲ http://www.enaplo.hu/FreeBSD/handbook/

A pingvin hidegben érzi jól magát...

A szervereknek állandó 20-22 fokos hőmérséklet szükséges a megbízható működéshez. Ezt a hőmérsékletet komolyabb helyeken klímaberendezések biztosítják. Murphy idevágó idézete: „Ami elromolhat, az el is romlik...” És ilyenkor jön jól egy rendszer, amellyel akkor is értesíteni tudjuk a klímát szervizelő céget, ha történetesen a szerverszobától messze vagyunk.

A hőmérséklet folyamatos figyelésére számos lehetőségünk nyílik, melyek között olcsó és egészen drága is akad. Mi az alábbi lehetőségeket mérlegeltük:

Az egyik szerver alaplapi szenzorának figyelése
- előny: nem igényel újabb beruházást
- hátrány: nem megbízható, ugyanis nem biztos, hogy ott a baj, ahol mér

Webkamerás megfigyelés
- előny: az esetleges áramszünet is látszik (később lesz szó róla, hogyan)
- hátrány: kell egy webkamera, egy normál hőmérő és egy asztali lámpa

Független digitális hőmérő (beágyazott rendszer, saját webszerverrel)
- előny: egyszerűen felügyelhető és akár több ponton is figyelhető vele a hőmérséklet
- hátrány: drága az eszköz

Tanszékcsoportunk a második alternatíva mellett döntött, így azt ismertetem részletesebben, de azért kitérek a harmadikra is.

Szükséges kellékek

Kell egy olyan webkamera, ami működik ***Video4Linux*** szabvány szerint is, vagy a ***Gphoto*** tudja kezelni. Szükségünk lesz aztán egy közönséges hőmérőre, egy asztali lámpára (ami megvilágítja a hőmérőt) és végül, de nem utolsósorban kell egy asztali gép ***Linuxszal***, amire a webkamera kerül. Ez utóbbi tulajdonképpen lehet az egyik szerver is. Nekünk egy egy ***Labtec*** kameránk volt, ami ***Logitech Quickcam*** kompatibilis, így a 2.4-es kernel is egyszerűen kezeli. Más jellegű webkamerával is érdemes kísérletezni, hiszen a ***Gphoto2*** rengeteget ismer már most is.

A kernel beállítása

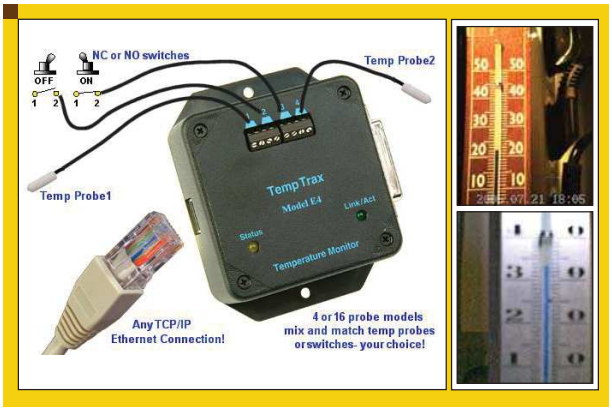
Ellenőrizzük, hogy a webkamera kezeléséhez használt rendszer kernelében engedélyezve vannak-e a következő opciók (ha nem, akkor új kernelt kell fordítanunk):

```
Character devices -> I2C Support
     M  I2C Support
     M  I2C bit-banging interfaces
     M  I2C device interface
     M  I2C /proc interface
Multimedia devices -> Video4Linux
     M  V4L information in proc filesystem
     M  CPIA Video for linux
     M  CPIA USB Lowlevel Support
USB Support
* Support for USB
* Preliminary USB device filesystem
* UHCI Alternate Driver
```

Mivel kameránk egy ***USB*** portra csatlakozik, az `lsusb` parancs segítségével célszerű ellenőrizni, hogy a rendszer valóban „lát-e”.. ***Debian*** alatt ez az `usbutils` csomagban található. Ha mindent jól csináltunk, valami ilyen lesz a végeredmény:

```
Bus 002 Device 001: ID 0000:0000
Bus 001 Device 001: ID 0000:0000
Bus 001 Device 002: ID 046d:0870 Logitech, Inc.
➥QuickCam Express
```

Amint az utolsó sorból látható, az ***USB*** alrendszer észrevette a kamerát. Igaz ugyan, hogy a miénk nem ***Logitech*** kamera, amint azt a szöveg állítja, de ez most mellékes. Itt az ideje, hogy megnézzünk egy élőképet. Ehhez be kell tölteni a `mod_quickcam` és `videodev` modulokat. Ha nincs más ***V4L*** eszközünk (például TV tuner), akkor az `xawtv`-vel megnézhetjük a kamera által szolgáltatott képet. Ha van TV tunerünk is (ez egy szerverszobában elhelyezett gépnél talán nem túl gyakori), akkor elképzelhető, hogy az `xawtv` ezt fogja látni. Ilyenkor néhány – itt nem részletezett – trükköt kell bevetnünk.



Automatizálás

Ha idáig eljutottunk, érdemes a ***cron*** segítségével automatizálni a szükséges műveleteket. Erre ajánlom a ***Videodog*** nevű programot, ugyanis segítségével nagyon egyszerűen le lehet menteni a pillanatképeket és a feliratozás se bonyolult. Az alábbi szkriptecske végezheti ezt a feladatot:

```
#!/bin/bash
# pillanatkep-mento

DATUM=`date +"%d%H%M"`
/usr/local/bin/videodog -x 352 -y 288 -w 3 -d
➥/dev/video0 -j -f /home/webcam/cam${DATUM}.jpg
➥-S "%Y.%m.%d %H:%M" -X 300 -Y 270
```

Ez a szkript lekérdezi az aktuális dátumot – ez lesz a fájlnév egy része – valamint a már korábban említett `videodog` segítségével kiolvas egy képkockát a webkamerából és feliratozza `év.hónap.nap óra:perc` formában. A formázó karaktersorozat részletesen a `videodog` leírásában található meg. Már csak egy webszerver kell és kész.
Hogy kicsit mozgalmasabb legyen a dolog, készült egy másik szkript, ami egy nap képeit összefűzi egy ***AVI***-ba. Így lehet látni a hőmérséklet növekedését/csökkenését. Íme a szkript:

```
#!/bin/bash
# a pillanatkepekbol mozgokep
# TODO: popcorn-t hozni ;-)

NAP=`date +%d`
/usr/local/bin/mencoder "mf:///home/webcam/
➥cam$NAP*.jpg" -mf fps=15:type=jpg:w=352:h=288
➥-o /home/webcam/cam$NAP.avi -ovc lavc -lavcopts
➥vcodec=mjpeg:vbitrate=100 -quiet
```

Ez a videó 15 képkockát fog tartalmazni másodpercenként, ami egy nap alatt körülbelül 20 MB adatot jelent.

Nem Video4Linux-os kamera használata

Lehetőség van olyan kamera használatára is, amely nem kompatibilis a ***Video4Linux*** rendszerrel. Ekkor a ***Gphoto2***-t kell használnunk. A feliratozásra a kicsit bonyolultabban használható `convert` parancsot ajánlom, amely az *ImageMagick* csomag része. A kamera „meglétét” megint az `lsusb` parancs segítségével ellenőrizhetjük:

```
Bus 001 Device 001: ID 0000:0000
Bus 001 Device 004: ID 2770:9120 NHJ, Ltd Che-ez!
➥Snap / iClick Tiny VGA Digital Camera
```

A képet most a következőképpen töltjük le:

```
ketchup $> gphoto2 --auto-detect --capture-preview
Modell                         Port
----------------------------------------------------
Argus DC-1510                  usb:
Fájl mentése mint sq_cap.ppm
ketchup $>
```

Ebből a részből is tisztán látszik, hogy az eszközt az `lsusb` teljesen másnak ismerte fel, mint a `gphoto2`, de ettől nem kell különösebben megijedni. A `gphoto2`-t azért is ajánlom mindenki figyelmébe, mert nagyon sok kamerát támogat, sőt napról-napra többet.
Említettem a cikk elején, hogy ezzel a megoldással az áramszüneteket is figyelhetjük, már persze ha a hőmérős – pardon, webkamerás – szerver szünetmentes táplálást kap. Annyi csak a dolgunk, hogy a lámpát nem a szünetmentes tápegységre tesszük. Ilyenkor ha van egy kimaradás, az a napi mozgóképnél látszani fog – sötétebb lesz a kép, vagy a feliraton kívül egyáltalán nem látunk semmit.

A profi megoldás

Végül, de nem utolsósorban szeretném megemlíteni az igazán profi megoldást, ami egyben a legdrágább is. Az ilyen minimum négy hőmérő szondát tudnak fogadni és komplett webszerver lakik benne. A négy szondának általában 10 méteres hatósugáron belül kell lennie. A dobozkához soros és ***Ethernet*** porton is csatlakozhatunk. Soros portra csatlakoztatható (egy szondát fogadó) dobozka ára körülbelül 20 ezer forint, de az ***Ethernet*** portot is tartalmazó (négy szondát fogadó) már 60 ezerbe kerül, ami nem mindig tűnik soknak, pláne akkor nem, ha több tízmilliós szerverparkot kell felügyelnünk. Ilyen „szerkentyű” például a linkek között szereplő ***WeatherShopban*** is kapható.

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.

KAPCSOLÓDÓ CÍMEK

VideoDog: ➲ http://paginas.terra.com.br/informatica/gleicon/video4linux/videodog.html

Mencoder: ➲ http://www.mplayerhq.hu/

WeatherShop: ➲ http://www.weathershop.com/temptrax.htm

Csak most, csak neked – Spamszűrők és a spam lélektana

Ma már senkinek nem kell elmagyarázni a kéretlen levél fogalmát. Ha valakinek publikus email címe van, az szinte biztosan kapott már szexuális ajánlatot elektronikus levélben, és mivel a spammerek a szükséges mennyiségű empátiaval is rendelkeznek, rögtön ajánlottak némi Viagrát is.

Valaki kiszámolta, hogy egy átlag felhasználó csak a spamek törlésével órákban mérhető időt veszít el évente. Ha már te is unod a ***Del*** gomb mindennapi kötelező ujjgyakorlatát, akkor a spamszűrők lehetnek segítségedre, amelyek a beérkező levél bizonyos jellemzői alapján, különféle módszerek alkalmazásával döntik el, hogy az spam (szemét; kéretlen levél) vagy ham (hasznos levél). Az alábbiakban különféle módszereket ismertetek a spam felismerésére. A spamszűrő alapvetően két helyen működhet: kliens- vagy szerveroldalon.

Mi a spam?

A spam olyan elektronikus levél, amelynek számunkra semmi értelme, és fölöslegesen foglalja erőforrásainkat (a gép és ember idejét), arról már nem is szólva, hogy ezek tartalma sok esetben félrevezető, pl. a nigériai milliókról sajnos többen bedőltek. Az előbb említett meghatározásból az is következik, hogy ami számomra kéretlen és bosszantó reklám (spam), az másnak esetleg értékes levél (ham). Ez pedig alaposan megnehezíti a spam szerveroldali szűrését.

Kulcsszó alapú szűrés

A spam szűrésének legegyszerűbb módja az, ha bizonyos szavakat keresünk a levélben (például ***szex***, ***VIAGRA***), és ha megtaláljuk a levél fejlécében vagy a törzsében, akkor azt spamnek tekintjük. A módszert gyakorlatilag minden alkalmazás támogatja mind kliens- (például ***Thunderbird***), mind pedig szerveroldalon (például ***procmail***, ***maildrop***, ***postfix***). Azonban az egyszerűsége egyúttal hátrány is. A spammerek ugyanis úgy kerülték meg ezeket a szűrőket, hogy például a ***VIAGRA*** szóból lett ***V1AGRA***, ***V.I.A.G.R.A***, stb, amit a szűrő már nem azonosított spamként – de mi még igen. A variációk száma pedig olyan nagy, hogy egy kulcsszó listát manuálisan karbantartani képtelenség a gyakorlatban. További hátrány, hogy ha így akarunk pl. a szex hirdetésektől megszabadulni, akkor az olyan levél is a kukában végzi, amely pl. a szextáns szót tartalmazza és nem spam, arról nem is beszélve, hogy a ***szex*** szó legitim levelekben is előfordulhat.

A kulcsszó alapú módszer változata a reguláris kifejezésre történő szűrés, amivel összetettebb és hatékonyabb szűrés végezhető. Ez a módszert – korlátozottan – akár férgek szűrésére is használható, ha szerepel azokban egy ismétlődő minta.

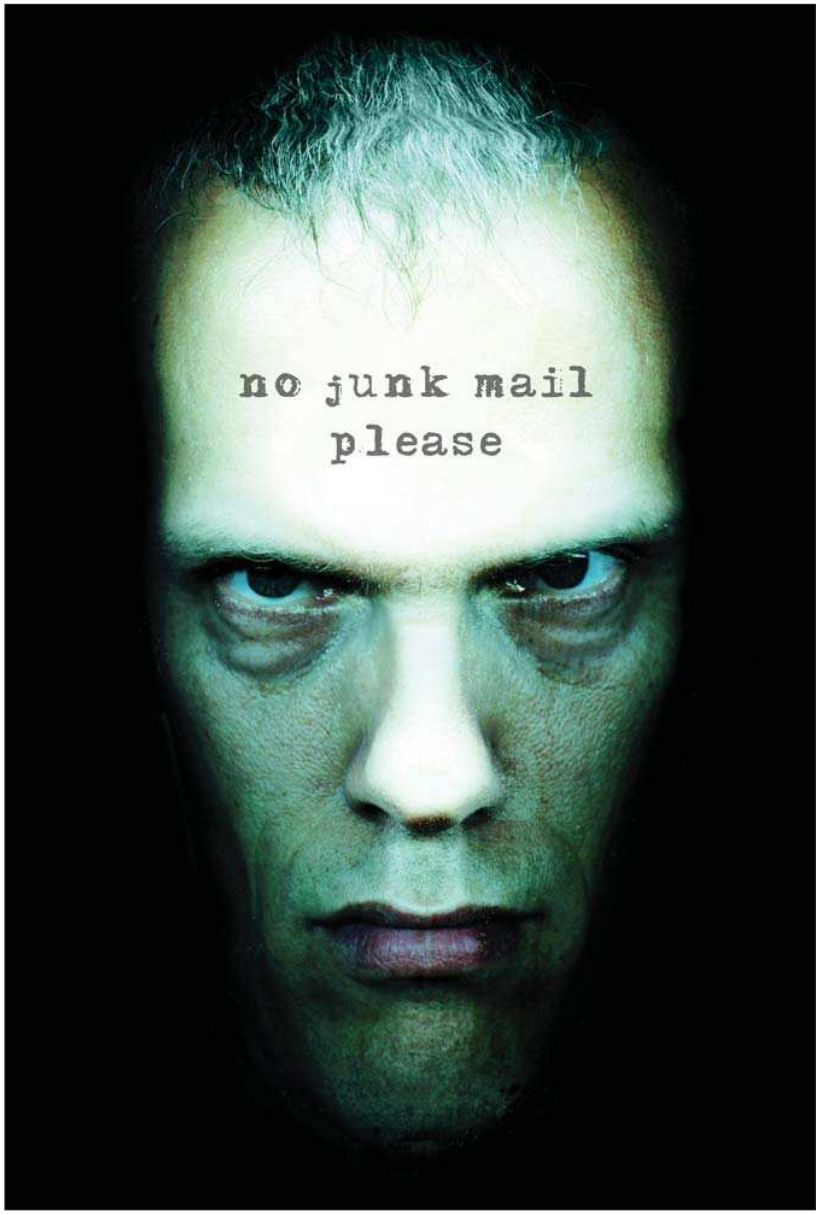
Feketelista

Ez az eljárás a levél fejlécében szereplő ***IP*** címeket veti össze különféle ***RBL*** listák, például ***MAPS*** (➲ ***http://www.mail-abuse.com/***),

ORDB (➲ *http://www.ordb.org/*) adatbázisával, és megvizsgálja, hogy azok szerint a minket megszólító gép már ismert spammer vagy levéltovábbító (***open relay***). (Ilyen lehet például egy megfertőzött zombi.) A módszer a gyakorlatban egy egyszerű *DNS* lekérdezést jelent, és a kapott válasz alapján lehet eldönteni, hogy az ***RBL lista szerint*** az adott gép spam forrás vagy sem. A feketelisták úgynevezett fehérlistákkal (whitelist) is kombinálhatók, amely listán szereplő *IP* címeket mindig átenged a szűrő. Ez a technika egyszerű, viszonylag kevés erőforrást használ, széles körben implementált, és szinte minden modern levelezőkliens és -szerver támogatja. Természetesen nem csak a levél fejlécében szereplő címeket, neveket lehet feketelistákkal összevetni, de akár a levél törzsében levőket is, sőt az *SMTP* kapcsolat során megadott neveket, címeket is (pl. a *HELO*, *MAIL FROM*, *RCPT TO* utáni paramétereket), illetve a kliens *IP* címét is, és ha annak nincs korrekt *PTR* rekordja, az minimum gyanús.

Nem mindenki elégedett azonban a feketelisták hatékonyságával. Előfordul, hogy egyes *RBL* listákon egy teljes C-osztályt tiltanak ki akkor is, ha csak egyetlen renitens küld spamet az adott hálózatból. Ez különösen a szerverbérlők (colocation szolgáltatás) számára lehet igen kellemetlen, mert így ártatlanul is tiltólistára kerülhetnek, a helyzet megoldása pedig sokszor nem kevés utánajárást igényel. A spammerek pedig igen ritkán használják saját azonosítható gépüket kéretlen levelek küldésére, inkább feltört zombikat vagy korlátozás nélküli továbbítószervereket (***open relay***) használnak erre. Ezekből meg éppen elég van ahhoz, hogy ne nagyon viselje meg őket, ha pár darab tiltólistára kerül. Paul *Graham* a *„Filters vs Blacklists”* című írásában (➲ *http://www.paulgraham.com/falsepositives.html*) rámutat, hogy például a *MAPS* feketelista úgy fogta meg a spamek 24%-t, hogy közben az ártatlan levelek 34%-t spamnek tekintette (false positives).

A feketelista egyik változatában nem konzultálunk *RBL* listákkal, hanem a tiltást *IP* szinten végezzük, mondjuk egy csomagszűrő alkalmazás (például *ipf*, *iptables*) segítségével, esetleg az adott



IP címet a ***loopback*** interfész felé irányítjuk. Ebben az esetben azonban nekünk kell kézzel – vagy más módon – karbantartanunk az ehhez szükséges ***netfilter*** illetve ***route*** bejegyzéseket.

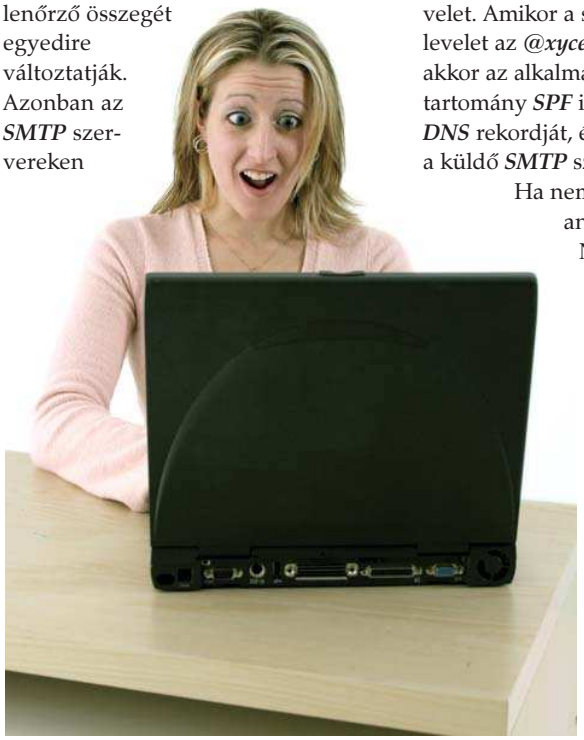
Szürkelista (greylist)

A szürkelista a spammereknek azt a szokását használja ki, hogy azok az első sikertelen próbálkozás után általában feladják, és továbblépnek a következő címzettre. Amikor az *SMTP* kiszolgálónk levelet kap, azt először 450-es hibaüzenettel (átmeneti hiba) elutasítja. A szabványoknak megfelelően működő *SMTP* szerverek egy bizonyos idő múlva újra próbálkoznak. A mi szerverünk azonban feljegyezte a küldő *SMTP* szerver azonosítóját, a feladó és a címzett email címét, továbbá az időt (egy bizonyos idő múlva ez a bejegyzés lejár), és a küldő *SMTP* szerver második próbálkozását már elfogadja. Így a spammerek többsége nem jut be, míg a legitim levelezőpartnerek igen. Ezt a megoldást támogatja például a *Postfix* egy egyszerű ***policy démon*** segítségével (➲ *http://www.postfix.org/addon.html*). A *Postfix* levelező listán többen igen kedvező tapasztalatokról számoltak be, míg néhányan kifogásolták a levelek késleltetését. Itt is lehetőség van fehérlista definiálására, ahol meg lehet adni azokat az *IP* címeket/feladókat/címzetteket, amelyeket azonnal

elfogad a szerver. Jelenleg ez a módszer jó hatásfokkal dolgozik, de ez könnyen a múlté lehet, ha a spammerek alkalmazkodnak hozzá, és újraküldik a spamet. Ehhez azonban nagyobb teljesítményű infrastruktúrára van szükségük, ami nekik több pénzbe kerül, és ez nekünk – akik már torkig vagyunk a spammel – jó.

Distributed Checksum Clearinghouse (DCC)

A spameket általában nagy tömegben küldik ki, nem ritka, hogy némelyik több milliós példányszámot is elér. A *DCC* lényege az, hogy az ebben résztvevő minden *SMTP* szerver minden egyes beérkező levélből egy ellenőrzőösszeget képez, amit elküld a *DCC* szervereknek. A *DCC* szerverek nyilvántartják, hogy az egyes ellenőrző összegekhez hány találat tartozik. Az *SMTP* szerverek az elküldött ellenőrző összegre kapnak egy választ (egy számot). Ha ez a szám meghalad egy értéket, akkor nyilvánvalóan tömegesen kiküldött levélről, azaz spamről van szó.
A spammerek azonban ezt az eljárást is igyekeznek megkerülni (a spam fejlődik) úgy, hogy az egyes levelekbe véletlenszerű tartalmat tesznek, így kvázi testre szabják a leveleket (amihez egyetlen szó is elég), ezzel minden egyes levél ellenőrző összegét egyedire változtatják. Azonban az *SMTP* szervereken futó *DCC* kliens is úgy módosult, hogy többféle ellenőrző összeget képez, amivel képes kiküszöbölni, hogy a csupán 1-2 apró részletben (pl. idő, véletlenszerű azonosító) eltérő spamek különböző ellenőrzőösszeget (*checksum*) eredményezzenek.
A módszer annál hatékonyabb, minél többen vesznek benne részt. Eredményesen képes felismerni a nagy tömegben kiküldött spamet. Hátránya, hogy külön alkalmazás szükséges hozzá, és viszonylag bonyolult együttműködésre bírni a levelező alkalmazásunkkal. Ilyen alkalmazás pl. a *DCC* (➲ *http://www.rhyolite.com/anti-spam/dcc/*) és a *Vipul's Razor* (➲ *http://razor.sourceforge.net/*)



Sender Policy Framework (SPF)

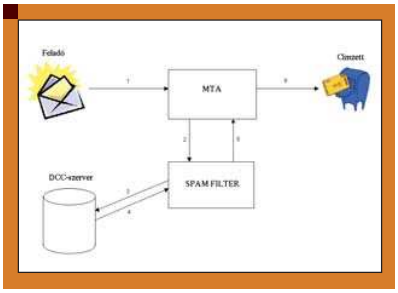
Megnehezíti a spammerek felderítését, hogy meghamisítják a levél fejlécében szereplő email címeket. Semeddig nem tart egy olyan levelet készíteni, amelyik pont úgy néz ki, mintha az X cég vezérigazgatója küldte volna. Mennyivel könnyebb lenne az életünk, ha biztosak lehetnénk abban, hogy egy levél feladója garantáltan az *@xyceg.hu*. Pont erre találták ki az *SPF*-et. Az adott tartományban el kell helyezni egy rekordot (jelenleg a tartomány *TXT* rekordja), amely definiálja, hogy az adott tartomány nevében mely *SMTP* szerverek küldhetnek levelet. Amikor a szerverünk kap egy levelet az *@xyceg.hu* tartományból, akkor az alkalmazás lekérdezi az adott tartomány *SPF* információt tartalmazó *DNS* rekordját, és ellenőrzi, hogy a küldő *SMTP* szerver szerepel-e ott. Ha nem, akkor nyilvánvalóan spamről van szó.
Noha a módszer működik, még nem terjedt el széles körben, így az *SPF*-et támogató *SMTP* szervernek egyelőre el kell fogadnia azokat a leveleket, amelyekben a feladó tartománya nem tartalmaz *SPF* információt. Továbbá vannak még meg nem oldott kérdések ez ügyben, pl. levelező listák kezelése.
Az *SPF* honlapján (➲ *http://spf.pobox.com/*) varázsló segíti a megfelelő *DNS* rekord elkészítését, amit érdemes publikálni, hogy minél kevesebb esélyt adjunk arra, hogy a spammerek a mi tartományunkat hamisításra használják fel.

Kérdés-válasz (challenge-response) szűrők

Amikor a szerver egy ismeretlen feladótól kap egy levelet, akkor automatikusan visszaküld egy kérdést (challenge) tartalmazó üzenetet. Ez lehet csupán annyi, hogy „válaszoljon erre a levélre", de igényelheti egy komplex web űrlap kitöltését is. Ha a kért műveletet végrehajtja a feladó, akkor a rendszer elfogadja az emailt. Mivel a spammerek általában hamisított feladó email címet használnak, ez a megoldás minden ilyen levelet blokkol. Ha pedig mégis az ő saját, élő címüket használják, akkor sem igen létezik olyan tömeges levélküldő alkalmazás (*bulk mailer*), amely nagy mennyiségben lenne képes ezekre a kérdésekre válaszolni.
Noha ez a módszer hatékony a spammerek ellen, van azonban néhány kellemetlen mellékhatása is, pl. kérdéses, hogy levelező listák esetében mennyire járható ez az út, és ez érvényes minden olyan levélre, amit nem ember adott fel (például űrlap kitöltésének e-mailben történő automatikus visszaigazolása).
A módszer egyik változata egy olyan speciális „bélyeg" meglétét követeli meg a levélben, amelynek elkészítése relatíve számításigényes. A legitim alkalmazások számára ez nem okoz számottevő terhelést, de tömeges levélküldésnél már olyan sok gépidőre van szükség a bélyegek előállítására, hogy a *spamware* program teljesítménye (a kiküldött levelek száma) erősen lecsökken.

Bayesian szűrők

Bár az előbb ismertetett módszerek több-kevesebb hatékonysággal működnek, van egy még jobb megoldás, mégpedig a levelek tartalmának elemzése. Az eljárás *Thomas Bayes* matematikus után kapta a nevét, aki egy valószínűségelmélettel kapcsolatos képletet dolgozott ki. A módszer lényege az, hogy amikor egy levelet kapunk, a *Bayesian* program azt szavakra (precízebben szólva tokenekre)



bontja. Két adatbázis (*HAM* és *SPAM*) segítségével minden egyes tokenhez meghatároz egy valószínűséget, mekkora az esélye, hogy a levél spam, ha az adott token szerepel benne. Ezután a tokeneket sorrendbe állítja aszerint, hogy valószínűségük mennyire tér el egy semleges középértéktől (amiről nem tudjuk eldönteni, hogy ham vagy spam). Végül veszi a középértéktől leginkább eltérő N darabot, és egy összetett képlet segítségével kiszámítja azok összesített spam valószínűségét. Az alkalmazás általában a levél fejlécébe beszúr egy extra mezőt, amiből kiderül, hogy az adott levél szerinte spam vagy sem, így el tudjuk dönteni, mit tegyünk ezek után a levéllel.
A *Bayesian* szűrőt azonban használat előtt tanítani kell, azaz létre kell hoznunk a spam illetve a ham adatbázist, amelyhez a leveleknek egy olyan halmazát kell képezni (például egy mailbox spoolfile), amelyben csak spam illetve csak ham van.
Ez a szűrés is végezhető a felhasználó oldalán, például a *Thunderbird* beépített *Bayesian* szűrővel rendelkezik, és már viszonylag kevés (<100) üzenetnél is jól osztályozza a beérkező leveleket. Több alkalmazás létezik, amelyik szerveroldalon valósítja meg ezt a funkciót. A *bogofilter* parancssorból futtatható, így egyszerűen használható például *maildroppal* együtt a levelek osztályozására. Szerveroldalon is számos alkalmazás támogatja a *Bayesian* algoritmust, ill. annak különböző módosított változatait. Ezen alkalmazások közül némelyik *SMTP* protokollon kommunikál az *MTA*-val. A *Postfix* ezekkel is képes együttműködni.
A módszer előnye, hogy a téves pozitív azonosítások száma minimális, véleményem szerint a legjobb megoldás a felsorolt eljárások közül. Hátránya, hogy az összes közül a legerőforrásigényesebb, továbbá tanítani kell, mielőtt használható.
Ha valaki többet szeretne megtudni a *Bayesian* algoritmus lelkivilágáról, ajánlom figyelmébe a korábban említett *Paul Graham* honlapja (➲ *http://www.paulgraham.com/*) mellett *Gary Robinson* két írását: *„Gary Robinson's Rants"* (➲ *http://radio.weblogs.com/0101454/stories/2002/09/16/spamDetection.html*) és *„A Statistical Approach to the Spam Problem"* (➲ *http://www.linuxjournal.com/article/6467*)

Heurisztikus szűrők

Ez a módszer azt használja ki, hogy a spamek szokatlan jellegzetességekkel rendelkeznek. Ezért egy sor tesztet végez el a leveleken, és minden egyes szokatlan, spamre jellemző dolgot pontoz. Ha a pontok összege elér egy határt, akkor a levelet spamnek tekintjük. Büntetőpont jár azért, ha pl. hiányzik a `From:` vagy `To:` mező; ha ezek tartalma érvénytelen; ha a címzett nem szerepel sem a `To:` sem pedig a `Cc:` mezőkben; ha a `From:` és a `To:` mezők azonosak; ha több, mint 10 cím szerepel a `To:` vagy `Cc:` mezőkben; ha hiányzik vagy érvénytelen a `Message ID:`; ha a levél csak *HTML* részt tartalmaz, közönséges szöveget nem. Bizonyos levelekben csak egy – azonosítót is tartalmazó – link szerepel, amire kattintva lehet elolvasni a spamet. A cikk írása napján lettem figyelmes egy ➲ *www.xxxx.spammerdomain.com* alakú linkre, ahol az *xxxx* egy több karakterből álló egyedi azonosító, azaz egy wildcard tartománynévről van szó. Egy másik változatban valamelyik ismert keresőgép felparaméterezett *URL*-je szerepel, amelyre kattintva jelenik meg a spammer oldala. Érdemes ezeket is pontokkal „jutalmazni". A *SpamAssassin* (➲ *http://spamassassin.apache.org/*) ilyen típusú szűrésre (is) képes.

Egyéb szűrési lehetőségek

Előfordul, hogy csak egyetlen csatolt képet kapunk, amely tartalmazza a kéretlen üzenetet. Ez ellen például képfelismerő (*OCR*) alkalmazásokkal lehet védekezni, amelyek képesek kinyerni a szöveget a képből, aztán jöhet az elemzés.
A *Bayesian* elv egyik továbbfejlesztett változata a szavak kombinációját is figyeli, amely még jobb eredményt adhat. Ilyen például a *CRM114* nevű diszkriminátor (➲ *http://crm114.sourceforge.net/*).
A feketelista egy további változata esetén a gyanús *IP* címekre sebességkorlátozást (*traffic shaping*) lehet végezni. Ebben az esetben a vélt spammerek eleve csak korlátozott mennyiségű adatot vihetnek át időegység alatt. Ennek egy másik változatában az *SMTP* kiszolgálónk az ismeretlen kliensekkel szándékosan lassan kommunikál, pl. több másodpercet is vár, amíg egyáltalán `220 SMTP` bannert ad. Ez az ötlet arra épít, hogy a spammerek türelmetlenek, minél rövidebb idő alatt minél több levelet akarnak elküldeni.
Ha szerencsénk van, a spammer program leidőzít, és odébb áll.

Melyiket használjam?

E sorok írója szerint a legjobb választás, ha kombináljuk a fentebb ismertetett módszereket, ill. mind szerver-, mind pedig kliensoldalon alkalmazunk spamszűrést. Szerveroldalon a *Bayesian* szűrést javaslom egyesíteni a szürkelistával és a heurisztikus szűréssel, míg kliensoldalon jó döntés a *Thunderbird* adaptív spamszűrője. De a kulcsszó szerinti szűrés is jól használható, ha pl. keleti spammel kell megbirkóznunk. Azt is figyelembe kell venni, hogy a spam fejlődik, így az a technika, amely ma hatásos ellene, az holnap már lehet, hogy nem működik. Minél többféle szűrőt használunk, annál biztosabban tudjuk eldönteni egy levélről, hogy az spam vagy sem. De annál több erőforrást is igényel.
Egy jól működő spamszűrő kombináció esetén felmerülhet az az igény is, hogy immár ne egy kijelölt folderbe gyűjtsük a spamet, hanem egyszerűen dobjuk el, a felhasználókhoz már meg se érkezzen. Azonban úgy vélem, hogy ez nem jó ötlet, mert időnként mindegyik szűrő hibázik, és egy elvesztett levél akár sok elvesztett pénzt is jelenthet.

Sütő János
(jsuto@freemail.hu)
1997 óta használ Slackware Linux-ot. Szabadidejében a postfix clapf nevű vírus- és spamszűrőjét polírozza.

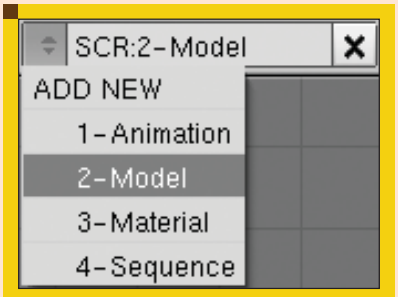
Térhatás

A Blender használata (2. rész)

A felhasználói felület, és az Edit Mode alapjai

Az elmúlt hónapban a 3D View kezeléséről és az alapvető transzformációkról volt szó, most a felhasználói felület kerül kicsit részletesebben bemutatásra, de az Edit Mode-ot is górcső alá vesszük.

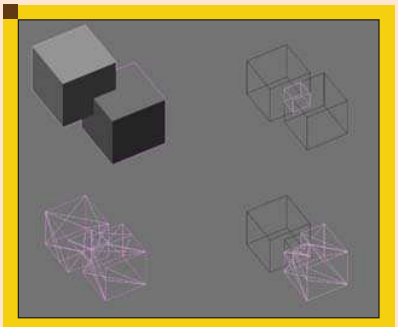
A *Blender* teljes felhasználói felülete különböző ablakokra tagolódik. A lenti gombokat tartalmazó panel is ablak, és a fenti menüsor is egy ilyen ablak fejléce. Mind rendelkezik egy úgynevezett fejléccel (*Header*), ami lehet az ablak tetején (*Top*) vagy alján (*Bottom*), de akár el is rejthetjük (*No Header*). Az adott ablak fejlécére jobb egérgombbal kattintva kiválaszthatjuk annak új helyét. Egy eltüntetett fejlécet egyszerűen visszavarázsolhatunk, ha az ablak felső vagy jobb oldali szegélyére kattintunk jobb egérgombbal, majd kiválasztjuk az *Add Header* opciót.
A másik két opció (*Split Area* és *Join Area*) az ablakok kettéosztására, illetve két ablak összeillesztésére használható. Így tetszőleges elrendezésű felhasználói felületet hozhatunk létre magunknak. Maga a *Blender* is rendelkezik néhány különböző elrendezéssel. Ezek között a fenti menüsorban válthatunk (*SCR:2-Model* az alapméretezett), vagy a *CTRL+Jobb* és *CTRL+Bal* billentyűkombinációkkal lépkedhetünk. Természetesen létrehozhatunk saját felületeket is és elmenthetjük őket.



1. ábra Néhány előre beállított ablak elrendezés

Most lássuk kicsit részletesebben a legfontosabb ablakokat, és azok feladatait.
A *3D View*-ról már a múlt hónapban is szó volt. Ez az egyik legtöbbet használt ablak (és persze a legtöbb helyet foglalja el a képernyőn). A 3D-s objektumaink megjelenítésére, szerkesztésére szolgál. Alapméretezésként ezt láthatjuk középen.
A másik gyakran használt ablak a *Buttons Window*, ami – a nevéből is könnyen kitalálható – gombokat tartalmaz. Ráadásul tekintélyes mennyiséget, funkciónként csoportokra bontva, amik között a fejlécen található gombokkal válthatunk. Az objektumok finombeállításától kezdve a speciális funkciókig

2. ábra Különböző ablak típusok



3. ábra Boolean: Intersect (jobbra, fent), Union (ballra, lent) és Difference (jobbra, lent)

4. ábra Dobókocka boolean műveletekkel

5. ábra Téglafal duplicate művelettel

6. ábra Objektum áthelyezése másik Layer-re

mindent elérhetünk itt. Ahogy a program újabb verziói kijönnek és az elérhető funkciók bővülnek, a gombok elhelyezkedése változhat, de a tematikus elrendezésnek köszönhetően mindig könnyen megtalálhatjuk amit keresünk (feltéve, hogy tudjuk mit is keresünk pontosan). A program indításakor ez az ablak alul van.

A ***User Perefrences*** nevű ablakból a program indításakor csak egy fejlécet látunk a képernyő tetején, azonban lefelé húzva előtűnik az ablak tényleges funkciója: a különböző felhasználói beállítások. Ha van elég időnk és kreativitásunk, akár egy teljesen új színsémát is létrehozhatunk a ***Blenderhez***.

A fenti három ablakon kívül még számos, számunkra egyelőre kevésbé fontos ablak létezik, amelyeket indításkor nem is látunk. Lássunk közülük néhányat a teljesség igénye nélkül.

Ablakok

Az animációk időbeli finomhangolását az ***IPO Curve Editor*** nevű ablakban végezhetjük el. Egy grafikonon, görbék segítségével követhetjük nyomon az objektumok mozgását.

A ***Blender*** rendelkezik egy egyszerű, beépített szövegszerkesztővel (***Text Editor)*** is. Ezt legtöbbször ***Python*** szkriptek írására használhatjuk, így innen rögtön futtathatjuk is kis programocskáinkat.

Az ***Outliner*** (régebben ***OOPS Schematics***) ablakban az objektumok felépítését és egymáshoz való kapcsolatát követhetjük nyomon. Összetettebb jeleneteknél jól jöhet, ha átlátjuk a teljes hierarchiát.

A ***File Browser*** és ***Image Browser*** ablakok funkciója és használata pedig valószínűleg mindenkinek egyértelmű. Az előbbi mentéskor, illetve megnyitáskor, az utóbbi képek keresésekor használatos.

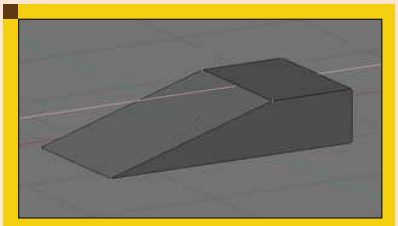
A továbbiakban szeretnék bemutatni néhány módszert, amivel időt és energiát takaríthatunk meg Object módban való munkánk végzése közben.

Lehetőségünk van arra, hogy objektumainkkal ***Boolean*** műveleteket végezzünk, összeadjuk vagy kivonjuk őket egymásból, esetleg vehetjük a közös részüket... Így új objektumokat kaphatunk. Működésük hasonló a matematikai értelemben vett halmazműveletekhez (unió, közös rész, stb).

Állítsunk be két kockát a *3. ábrán* látható módon. Ezzel a két kockával fogjuk kipróbálni a ***Boolean*** transzformációkat. Jelöljük ki mindkét kockát, majd nyomjuk le a ***W*** billentyűt. Innentől kezdve három lehetőségünk van. Mindhárom esetben egy új objektum keletkezik, azonban a régiek is megmaradnak, azokat nekünk kell törölnünk, ha többé nincs rájuk szükségünk.

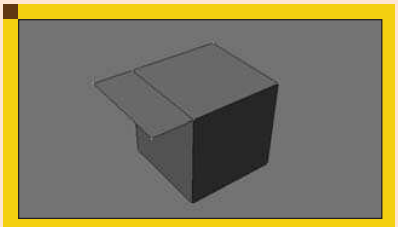
- Ha az ***Intersect*** opciót választjuk, a kiválasztott két objektum közös részéből egy új objektum keletkezik.
- Az ***Union*** opcióval összeadjuk a két objektumunkat, így egy nagyobbat kapunk, ami mindkettőt tartalmazza.
- Egy objektumot egy másikból kivonhatunk, ha a ***Difference*** opciót választjuk. Ebben az esetben figyeljünk arra, hogy mindig az utoljára kijelölt modellt (a világosabb rózsaszínűt) vonjuk ki a másikból.

Ezzel a módszerrel könnyen és gyorsan tudunk dolgozni. A *4. ábrán* látható dobókocka például kizárólag ***Boolean*** transzformációkkal készült. Az előző számban ígéretet tettem,



7. ábra Téglánk eltorzítva

10. ábra Editing Buttons



8. ábra Extrude

9. ábra Törlés Edit Mode-ban

hogy – az akkor még egyke – téglánkból falat építünk. Az egyetlen problémánk, hogy egy ilyen építményhez bizony sok tégla kell, jóval több, mint amennyi nekünk van.
A megoldás: sokszorosítsunk. Jelöljük ki a téglánk, majd nyomjuk meg a ***SHIT+D (Duplicate)*** billentyűkombinációt. Téglánkból rögtön kettő lett. Az újat helyezzük el a régi mellé, majd ismételjük meg a műveletet, de ezúttal jelöljük ki mindkét téglánkat. Természetesen addig folytathatjuk, ameddig csak szeretnénk (vagy amíg számítógépünk memóriája bírja). Ha már rendelkezünk egy legalább 8 téglából álló sorral, elkezdhetünk felfelé is építkezni. Az *5. ábrán* látható fal is ilyen módon készült.
Előfordulhat, hogy egyszerre túl sok objektum van előttünk, mi azonban csak néhánnyal szeretnénk foglalkozni. Erre is van megoldás. A ***Blender*** rendelkezésünkre bocsájt úgynevezett rétegeket (***Layer***), amiken elhelyezhetjük objektumainkat, majd ki-be kapcsolhatjuk őket. Így félretehetünk egy zavaró objektumot anélkül hogy törölnénk, vagy megváltoztatnánk a helyét. Egyszerűen jelöljük ki, nyomjuk ki meg az M billentyűt, és válasszuk ki, hogy melyik rétegre szeretnénk áthelyezni. Alapméretezésként az első rétegen dolgozunk, és csak azt látjuk. A ***3D View*** fejlécén tudjuk változtatni, hogy épp melyik az aktuális. Ha a ***SHIFT*** billentyűt nyomva tartjuk egyszerre több ***Layert*** is kijelölhetünk. Ezennel elérkeztünk a tényleges 3D-s szerkesztéshez, a ***Blender Edit Mode***-jához. A továbbiakban leírom az ***Edit Mode*** alapjait, és további varázslatokat viszünk véghez téglánk segítségével. Ehhez jelöljük ki a téglánkat, majd nyomjuk meg a ***TAB*** billentyűt. Ezzel ***Edit Mode***-ba léptünk. Az ***Object*** módba való visszalépés szintén ilyen egyszerű.
Ahhoz azonban, hogy elkezdhessük téglánkat csonkítani, előbb tisztáznunk kell néhány alapfogalmat ***Vertex***-nek, (vagy ***Vertice***) nevezzük a 3D-s térben X, Y és Z koordinátákkal megadott pontokat. Ezeknek a pontoknak nincs kiterjedése, így a renderelt képen is láthatatlanok lesznek, kizárólag szerkesztés közben látjuk őket. Az éppen kijelölt ***Vertexek*** színe sárga, míg a kijelöletleneké rózsaszínű. Ne tévesszük össze őket azonban objektumunk középpontjától, ami szintén rózsaszínű, viszont szemmel láthatóan nagyobb. Téglánk 8 ***Vertexet*** tartalmaz.
A ***Vertexeket*** összekötő szakaszokat élnek (***Edge***) nevezzük. Éleink szintén láthatatlanok a renderelt képen, tehát egy oldalak nélküli „drót” kocka sem jelenne meg (van azonban mód mind az élek, mind pedig a ***Vertexek*** megjelenítésére). Téglánknak 12 éle van.
Az élek által bezárt területeket – például kockánk egy oldalát – ***Face***-nek (magyarul talán lap-nak lehetne fordítani) nevezzük. Nem törvényszerű azonban, hogy minden – élekkel határolt – terület ***Face***. Gondoljunk például egy olyan kockára, aminek egyik oldala hiányzik, tehát egyetlen lap kivételével az összes ***Vertex*** és él megtalálható benne. Egy ***Face***-t 3 vagy 4 él határolhat, tehát maximum 3 vagy 4 ***Vertexet*** tartalmazhat.
Téglánk tehát ***Vertexekből (Vertice)***, élekből (***Edge***), és lapokból (***Face***) épül fel, egy él több lapot is határolhat, illetve egy ***Vertex*** több él végpontja is lehet. A renderelt képen azonban csak a lapok láthatóak.
Elkezdhetjük tehát téglánkat ténylegesen is átalakítani. Dolgozni régebben csak ***Vertexekkel*** tudtunk, de a ***Blender*** újabb verzióiban lehetőségünk van élek, és lapok kijelölésére is. A ***3D View*** fejlécén válthatunk a kijelölési módok között (***Vertex Select***, ***Edge Select*** vagy ***Face Select Mode***). A kijelölt részeket ugyanolyan könnyen transzformálhatjuk, mint ahogy azt a teljes objektumainkkal tettük. Használhatjuk a mozgatást (***G***), a forgatást (***R***) illetve az átméretezést (***S***). Az A billentyűvel kijelölhetjük az összes ***Vertexet*** vagy levehetjük a kijelölést, használhatjuk a ***Border Select*** funkciót is, sőt a ***B*** billentyű egymás utáni kétszeres lenyomásával egy újabb kijelölési módot is elérhetünk.
Csonkoljuk meg téglánkat a *7. ábrán* látható módon: Jelöljük ki (jobb egérgombbal) az egyik élét, vagy a két végén lévő ***Vertexet*** (ügyeljünk rá, hogy más ***Vertex*** ne legyen kijelölve), nyomjuk meg a ***G*** billentyűt (mozgatás), majd mozgassuk el a számunkra megfelelő irányba. Itt is használhatjuk az ***X***, ***Y*** és ***Z*** billentyűket, ha csak adott irányba szeretnénk elmozdulást.
A billentyűk egyszeri lenyomásával

a globális, kétszeri lenyomásával pedig a lokális (mindig az adott objektum sajátja) koordináta rendszerhez képest transzformálhatunk. Néhány kivételtől eltekintve ugyanúgy transzformálhatjuk ***Vertexeinket***, mint ***Object*** módban objektumainkat.
Fontos azonban, hogy ha Edit módban új objektumot hozunk létre, akkor az hozzáadódik az éppen szerkesztett objektumunkhoz, így tudjuk objektumunkat bővíteni. Ezek után, ha visszaváltunk ***Object*** módba, a régi és az új, hozzáadott test egy objektumként jelenik meg. Ezért, ha külön objektumot szeretnénk létrehozni, mindig váltsunk vissza ***Object*** módba.
Egy másik mód a bővítésre az ***Extrude*** eljárás (***E*** billentyű), amivel a kijelölt részeket „kihúzva” bővíthetjük objektumunkat. Attól függően, hogy mit jelöltünk ki, több választási lehetőségünk van. Kihúzhatjuk csak az éleket, de akár az egész lapot is. Ezzel a módszerrel könnyen bővíthetjük a primitívek tárházát szinte tetszőleges módon.
Edit módban a törlés sem olyan egyszerű, hiszen törölhetünk ***Vertexeket***, éleket és lapokat is. Próbáljuk meg: Jelöljünk ki például kockánk egyik sarkát, majd nyomjuk meg az ***X*** billentyűt. Rögtön láthatjuk, hogy az ***Object*** módban látott egyszerű megerősítés helyett itt ki kell választanunk a megfelelő törlési módot:

- ***Vertices*** – Töröljük a kijelölt ***Vertex(ek)et***, vele együtt természetesen a belőle kiinduló éleket és lapokat is.
- ***Edges*** – Minden kijelölt élt törlünk. Ez maga után vonja, hogy az élek által határolt lapok is eltűnnek. Az olyan kijelölt ***Vertexeket*** is törli, amelyekhez nem kapcsolódik több él.
- ***Faces*** – Minden kijelölt lapot törlünk. A lap nélkül maradt kijelölt éleket és az árva ***Vertexeket*** is eltávolítja.
- ***All*** – Kijelöléstől függetlenül mindent töröl. Maga az objektum azonban megmarad, csak nem tartalmaz ***Vertexeket***.
- ***Edges And Faces*** – Az ***Edges***-hez hasonlóan töröl, az egyetlen különbség, hogy itt a ***Vertexek*** minden esetben megmaradnak.
- ***Only Faces*** – Csak a lapokat törli, minden más megmarad.

Néhány egyszerű beállítással könnyebben átláthatóvá tehetjük az ***Edit*** módban való szerkesztést. A ***Buttons Window*** fejlécén váltsunk át az ***Editing*** gombokra (balról az ötödik). (Ha az egeret pár másodpercig felette tartjuk, ki is írja, hogy ***„Editing (F9)”***) Ha még mindig ***Edit Mode***-ban vagyunk, akkor jobb oldalt láthatunk egy ***Mesh Tools 1*** nevű gombokat tartalmazó panelt. Itt kapcsoljuk be a ***Draw Faces*** és ***Draw Edges*** beállításokat. Rögtön láthatjuk a különbséget a ***3D View***-ban. Az előbbi a kijelölt lapokat, míg az utóbbi az éleket emeli ki szerkesztés közben.
A következő számban még mélyebbre merülünk az ***Edit Mode***-ban, bemutatásra kerül néhány – a munkát segítő – apróság és végre igazi képeket is renderelünk.

Szalai András (sly87@freestart.hu)

Jelenleg középiskolába jár, ahol informatikát tanul. Jövőre érettségizik. Hobbija a programozás és a biztonságtechnika, és a továbbtanulási szándékai is ilyen irányúak.

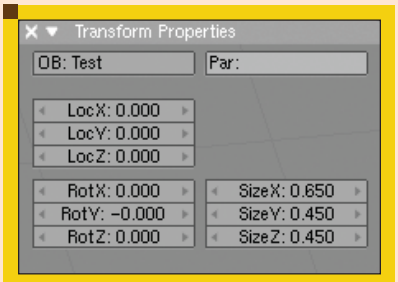
Animáció készítése a Blenderrel

Ha beülünk a moziba és megnézünk egy jó filmet, számtalan digitális trükköt látunk, amelyek már annyira valóságosak, hogy szinte el is hisszük. A most következő oldalakon egy mozgó pókot fogunk elkészíteni, amelynek működési elve azonos a filmekben szereplő óriáspókokéval, akik házakat döntenek le, vagy hatalmas hálóikba embereket szőnek. De azért a mi pókunk barátságos lesz.

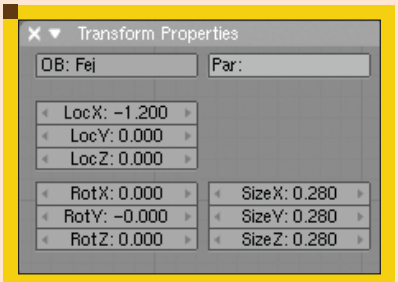
Először is el kell készítenünk a pókot. Mint általában a pókoknak, a mi pókunknak is két „része” lesz: a fejtor és a potroh. E kettő közül állnak ki a lábai, a fejtoron pedig lesz két csápja és két szeme. A ***Blenderben*** alapértelmezésként szerepel egy kocka, egy lámpa és egy kamera. A kockát töröljük ki így: jobb gombbal rákattintunk az eltávolítani kívánt objektumra, majd az ***X*** billentyűvel töröljük azt. Ez után adjunk a jelenetünkhöz egy gömböt, így: nyomjuk le a ***Space*** billentyűt, majd az előugró menüből válasszuk az ***Add*** menüpontot, itt a ***Mesh***-t, majd az ***Uvsprehe***-t. A továbbiakban, ha egy objektumot hozzáadunk a jelenethez, azt kicsit rövidebben fogom leírni. Az előbbi gömb hozzáadását például így: ***Space*** >> ***Add*** >> ***Mesh*** >> ***Uvsprehe***. Miután ez kész, megjelenik egy csomó sárga pont, ezek az úgynevezett vertexpontok, amelyeket most nem használunk, ezért lépjünk ki a szerkesztőmódból a ***TAB***-bal. Lapítsuk ki így: nyomjuk le az ***N*** gombot, majd írjuk be azokat az adatokat, amiket az *1. ábra* mutat!

Ezzel kész is van a pókunk teste, a fejét is hasonlóan kell elkészíteni: ***Space*** >> ***Add*** >> ***Mesh*** >> ***Uvsprehe***, majd ***N*** gomb, és a táblázatba azok az adatok kerüljenek, amelyeket a *2. ábra* mutat!

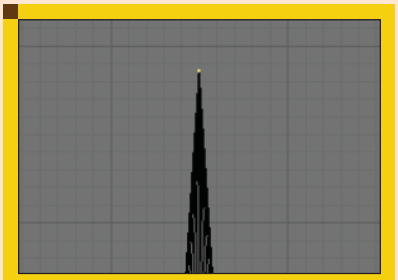
Most a lábakat adjuk hozzá: Kattintsunk a ***View*** menüpontra, majd itt a ***View Properties***-re, és a 3D kurzort toljuk el ***X*** irányba ***-0.85***-tel. Adjunk hozzá a jelenethez egy objektumot,



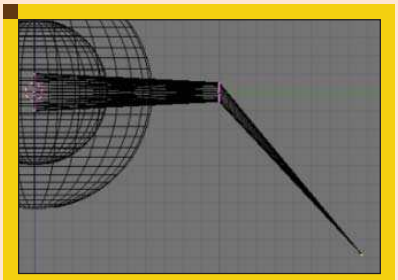
1. ábra A kilapításhoz szükséges adatok



2. ábra A fej adatai



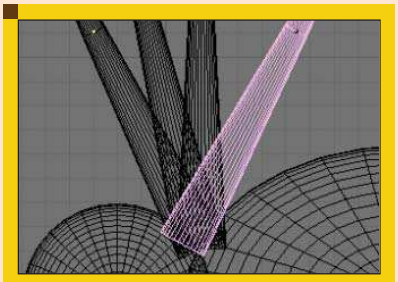
3. ábra A lábak „finomhangolása”



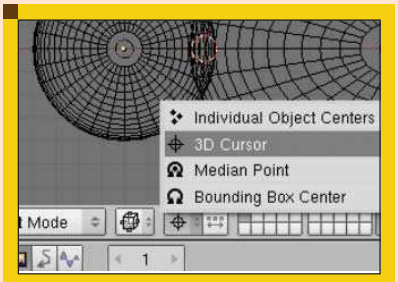
4. ábra A „térd” elkészítése

amelyből a láb lesz: ***Space*** >> ***Add*** >> ***Mesh*** >> ***Tube***. Ezt aztán forgassuk el az X tengelyen -90 fokkal (Nyomjuk le az ***N*** billentyűt, majd az előugró táblázatba a RotX bejegyzést módosítsuk -90.000-re). Nyomjuk meg a ***TAB***-ot, és a következő módon igazítsuk a vertexpontokat: A gomb, ettől elvesztik a kijelöltségüket, majd ***B*** gomb, ami egy keretek aktivál, és a benne lévő pontok lesznek kijelöltek. Ha sikerült, húzzuk össze teljesen az ***S*** billentyű segítségével őket. Körülbelül úgy nézzenek ki, mint a *3. ábrán*!

Most csinálunk neki egy térdet: Kétszer nyomjuk meg az A billentyűt, hogy minden pont ki legyen jelölve, majd a ***K*** gombot, a menüből pedig válasszuk a ***Knife (Midpoints)*** menüpontot. Húzzunk most egy felező vonalat, majd nyomjunk ***Entert***. Ettől a láb felénél további vertexpontok jöttek létre. Nyomjuk meg a 3-mast a numerikus billentyűzeten (figyeljünk, hogy a ***NumLock*** be legyen kapcsolva!), majd jelöljük ki a csúcspontot és a ***G*** billentyűvel a *4. ábrán* látható módon mozgassuk el!

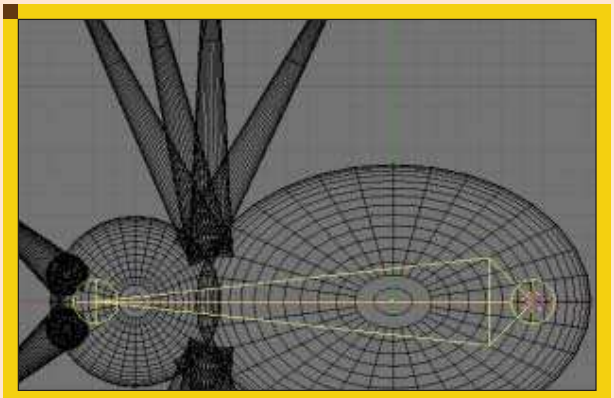


5. ábra A lábak elkészítése

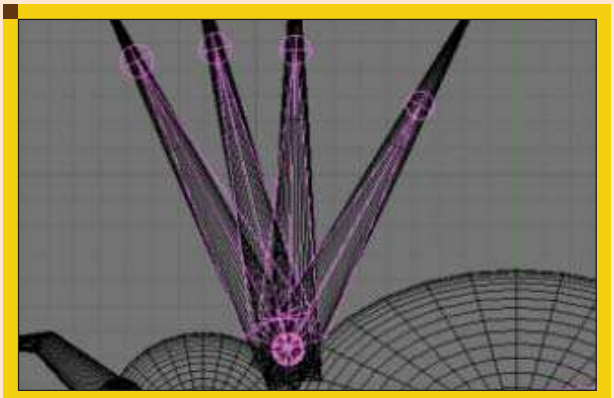


6. ábra A referenciapont átállítása

7. ábra A kész pók

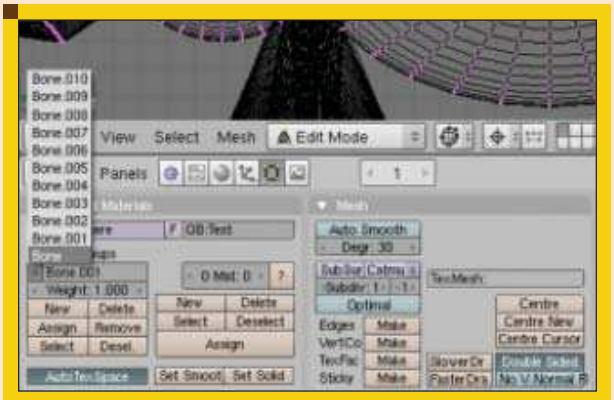


8. ábra Az első csont

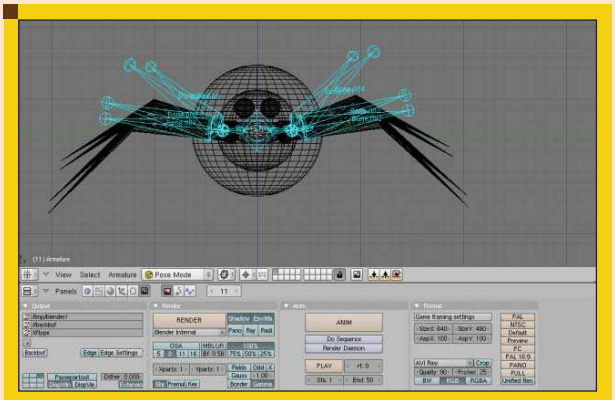


9. ábra A többi csont

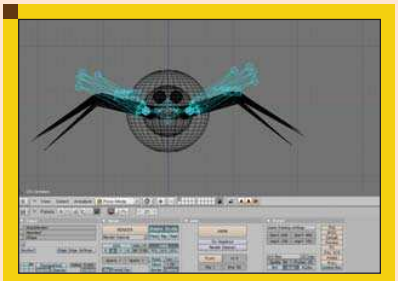
Ezzel egy láb készen is van. A többivel már könnyű dolgunk lesz: Nyomjuk meg a ***TAB***-ot, majd a 7-est és a ***Shift+D*** billentyűkombinációval másoljuk le a lábat három példányban, majd kedvünk és ízlésünk szerint mozgassuk és forgassuk át. Az *5. ábra* azt mutatja, hogy én hogy csináltam.
Ezzel kész az egyik oldalon a lábsor, ezeket összeolvaszthatjuk a ***Ctrl+J*** billentyűkombináció segítségével. Jelöljük is ki, majd készítsünk róla egy másolatot, amit aztán tükrözni fogunk. A ***TAB***-bal lépjünk szerkesztő módba és a tükrözési viszonyítási pontot állítsuk át ***3D Cursorra*** (*6. ábra*).
Nyomjuk meg az ***M*** gombot, majd válasszuk az ***Y Global***-t és már át is tükrözte.
Ugyanazzal a módszerrel, amivel a lábakat csináltuk, készítsük el egyedül a csápokat!
Következzenek a szemek: Először is adjunk hozzá egy gömböt (***Space >> Add >> Mesh >> Uvsprehe***), utána az ***S*** gombbal zsugorítsuk jó kicsire, helyezzük el a helyén, majd tükrözd a másik oldalra is!
Most pedig az egész pókot egybeolvasztjuk a ***Ctrl+J*** kombinációval, majd belépünk a ***Mesh panelra*** (***F9***) és megnyomjuk a ***Set Smoot*** gombot, ami lágyabb körvonalat ad pókunknak.
Ha mindent jól csináltunk, rendereljük a képet az ***F12*** billentyűvel. Valami olyasmit kell kapnod, mint a *7. ábra*!
Most jön az izgalmasabb része, vagyis az animálás! (A textúrázás és az ehhez hasonló dolgok most kimaradnak.)
Először is hozzá kell adnunk a jelenethez az úgynevezett ***Bone***-okat, azaz a csontokat. Először is győződjünk meg róla, hogy nem vagyunk szerkesztő módban, majd a ***3D Cursort*** mozgassuk az X tengelyen 0.65-ig és az Y-on 0-ig (***View >> View Properties >> X: 0.65 Y: 0.0***). Most adjunk hozzá egy csontot: ***Space >> Add >> Armature***. Húzzuk ki a csontot körülbelül úgy, ahogy az *8. ábra* mutatja, majd nyomjunk ***ESC***-et!
Ne lépjünk ki a szerkesztő módból, és adjunk még még csontokat a jelenethez úgy, ahogy a *9. ábra* mutatja (***SpaceKEY >> Add >> Bones***).
Most helyezzük a ***3D Cursort*** az X tengelyen -1.525-re, az Y-on pedig 0.125-ig, majd adjunk a csápjához is csontot (***Space >> Add >> Bones***)! Most tükrözzük át a csontokat, ehhez előbb a ***3D Cursort*** helyezzük az Y tengelyen 0-ra, majd jselöljük ki a lábcsontokat és a csápcsontot. Ezután nyomjuk le a ***Shift + D*** kombinációt, utána pedig egy ***ESC***-et. Az ***M***-mel tükrözzük át (a menüből az ***Y Global***-t válasszuk)!
Ezzel elkészült pókunk csontozata. Térjünk vissza objektum nézetbe a ***TAB***-bal, és jelöljük ki a pókot, majd a ***Shift*** lenyomva tartásával a csontozatot, így mindkettő kijelölt lesz, és a csontozat lesz a világosabb lila.
Nyomjuk le a ***Ctrl+P*** billentyűkombinációt, és az előugró menüből válasszuk az ***Armature*** lehetőséget, majd az újabb menüből a ***Create from closest bones***-t! Jelöljük ki most csak a csontozatot, és nyomjunk ***F9***-et, majd nyomjuk le a ***Draw Names*** gombot, hogy mindig lássuk, hogy melyik csont melyik. Jelöljük ki csak a pókot és váltsunk szerkesztő módba, ahol a lenti menüsorból ***(Vertex Groups*** menü), a bal szélen (amint azt a *10. ábra* is mutatja) lehet látni, hogy melyik csontot jelöltük ki és, ha rákattintunk a ***Select*** gombra, akkor sárgák lesznek a hozzátartozó vertexpontok.



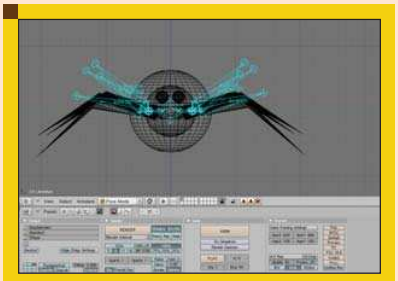
10. ábra A „Vertex Groups” menü



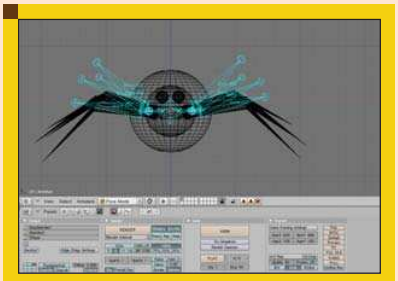
11. ábra A 11. képkocka beállításai



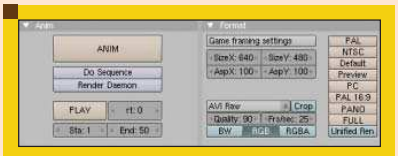
12. ábra A 21. képkocka beállításai



13. ábra A 31. képkocka beállításai



14. ábra A 41. képkocka beállításai



15. ábra Az animálás beállításai

A *8. ábrán* látható nagy csont irányítsa az egész pókot! Ehhez nyomjuk meg a ***Remove*** gombot, így most egyik vertexet sem tudja mozgatni. Most jelöljük ki az összes vertexet az ***A***-val, és nyomjuk meg az ***Assign*** gombot. Most a lábai következnek! Jelöljük valamelyik csontot, amelyik valamelyik lábához tartozik, és ugyanígy távolítsuk el tőle az összes vertexet a ***Remove*** gombbal. Most jobb egérgombbal jelöljük ki valamelyik vertexpontot, amelyik ahhoz a lábához tartozik, és nyomjuk meg az ***L*** gombot, amely az összes vele kapcsolatban álló (linked) pontot kijelöli (vagyis azt a lábat, mert az az összeolvasztás előtt külön objektum volt). Ezt csináljuk meg az összes lábával és a csápjaival is. Ezzel már van csontozata a pókunknak!
Innentől kezdve egyszerű dolgunk van. Jelöljük ki a csontozatot és váltsunk ***Pose*** módba. (Ott kell, ahol ***Edit***, vagy ***Object*** módba váltunk.) Itt a csontok kékek, a kijelöltek pedig zöldesek lesznek. Jelöljük ki az összeset az ***A***-val, majd nyomjuk le az ***I*** billentyűt. Itt válasszuk a ***LocRotSize*** lehetőséget. Ezzel az 1. képkockára eltároltuk ezt a pozíciót. Nyomjuk meg ötször a felfelé mutató nyilacskát, ezzel 50 képkockát ugrik előre, azaz 2 másodpercet. Most (még mindig legyen kijelölve az összes csont) mozgassuk előre az X tengelyen őket (***G***, majd ***X*** gomb), körülbelül úgy, hogy ahol eddig a pók feje volt, ott most a leghátsó pontja legyen. Most nyomjunk I gombot és válasszuk a ***LocRotSize***-t. Most váltsunk a 11-es képkockára, és a *11. ábrán* látható módon mozgassuk a lábait, majd nyomjuk meg az ***I***-t és válasszuk a ***LocRotSize***-t. Most válasszuk ki a 21-es képkockát, és mozgassuk úgy, ahogy a *12. ábra* mutatja. Aztán mentsük el ezt a pózt is, a 21-es képkockára (***I >> LocRotSize***).
A többi képkockával is ugyanígy kell eljárni. A 31-es képkockát a *13.*, a 41-est pedig a *14. ábra* mutatja. Ezek után, ha akarjuk ugyanígy mozgathatjuk közben a csápjait is, amit én most nem csináltam, de a pókot szándékosan úgy terveztem meg, hogy ez megvalósítható legyen.
Ezek azonban csak az én beállításaim, kis próbálkozással szebb és élethűbb animáció is készíthető. Most, hogy ez készen van, lépjünk vissza objektum módba és nézzük meg, hogy a kamera teljes egészében látja-e a pókot. Nézzük meg az 1-es képkockán, majd váltsunk át a 51-esre és itt is nézzük meg. Ha nem látná, akkor módosítsuk a kamera helyzetét úgy, hogy mindig lássa a pókot. Ha sikerült, nyomjuk meg az ***F10*** billentyűt, és állítsuk be azokat a beállításokat, amiket az *15. ábra* mutat, majd nyomjuk meg az ***Anim*** gombot. Az elkészült animáció ***avi*** formátumban a ***/tmp/blender/*** könyvtárba kerül.

Szabó Péter
(lizard@tvn.hu)
Jelenleg egy XXII. kerületi iskolában tanul. Szeret úszni, zenét hallgatni, és szeret kipróbálni mindenféle alkalmazást Linux alatt. Egyik álma, hogy egyszer majd ír egy saját ablakkezelőt!

3D ábrázolás – PoVRay (2. rész)

3D világunk fontos része a fény, s ehhez szükség van legalább egy fényforrásra, hogy lássuk is a térben elhelyezett tárgyakat. Fényforrást illetően több lehetőség között választhatunk, amelyek közül néhányat megtalálunk a valódi világban is. Természetesen léteznek olyan fényforrások is, amelyek csak a számítógép képi világában léteznek: ezekkel speciális effekteket tudunk létrehozni.

A valódi világban minden fényforrás létező tárgy is egyben, amely olyan jellegű fényt bocsát ki, ami rá jellemző: ilyen az izzólámpák széles spektrumú – pontszerű, a fénycsövek monokromatikus szórt jellegű, illetve a hidegtükrös halogén izzók erős fénykúpja és kismértékű fényudvara. A *3D* világban a fényforrásoknak vannak ilyen jellemzőik, azonban nem rendelkeznek tárgyias tulajdonságokkal: nem léteznek, csak a fényük árulja el őket. Ennek egyik mellékhatása, hogy nem okoznak megcsillanást, illetve nem vakítja el a kamerát a fényük. Ez többnyire előnyös, néha azonban a valósághoz hasonlóan előnyös lenne, ha a kamera úgy viselkedne, mint egy kamera, vagy a fényforrás lenne valóságosabb.

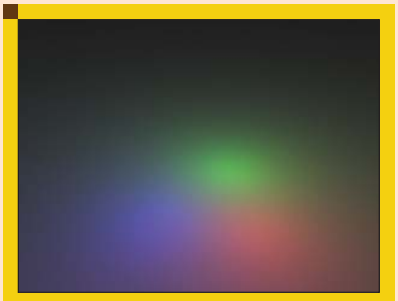
A *3D* helyszínen fényforrásoknak két alapvető tulajdonságuk van: a térben elfoglalt pozíciójuk, és a színük. A legegyszerűbb fényforrással már találkoztunk is:

```
light_source{
 <0,0,0> color White}
```

Ez a fény a koordináta rendszer origójában található és fehér színű fényt bocsát ki magából. Ha egy kamerát is hozzácsapunk az *SDL* állományhoz, akkor megtapasztalhatjuk a fényforrás és kamera mellékhatását is (*pov07.pov*). A kapott képen ugyanis tömör feketeséget látunk csupán, holott a fényforrásunk teljes erővel „világít". Ha elhelyezünk egy tetszőleges testet, amelyet a fény megvilágíthat, s onnan visszaverődhet, akkor már látni is fogunk a képen valamit (feltéve, ha nem a tárgy belsejébe tettük a fényforrást (*pov08.pov*, *1. ábra*). Nem meglepő módon a fehér fény zöld pacaként látszik, hiszen a létrehozott sík zöld színű, így a fehér fényben zöld színűnek *kell* látszania. Látható a sík megvilágításából, hogy a fényforrás felette található, viszont a kamerába nem csillan bele a fénye. Ezen a tényen jelenleg még nem változtatunk, fogadjuk el ezt a működésmódot.

1. ábra Zöld síkon fehér fényforrás

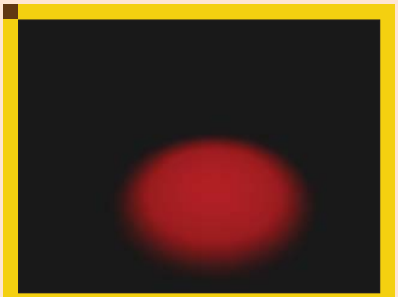
2. ábra Fekete semmi



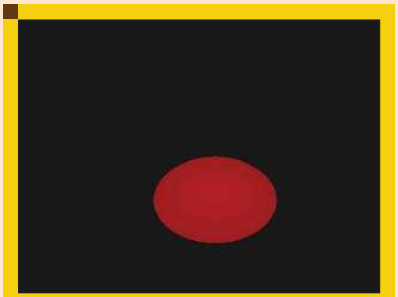
3. ábra Színátmenet

Ha a fehér fény helyett vörös színű a fényforrásunk (*pov09.pov*, *2. ábra*), akkor érdekes módon a zöld tárgyak fekete színűek lesznek. Ennek oka, hogy a tükröződésmentes tárgyak (amelyeket alapesetben a *PoVRay* készít) a fényt elnyelik, majd a saját színkomponensüknek megfelelő részét kibocsátják: a vörös fényt elnyeli a zöld színű síkunk, majd a kapott fény zöld összetevőjét bocsátja ki. A vörös fény azonban nem tartalmaz zöld komponenst, így kaptunk tömör sötétséget eredményül.

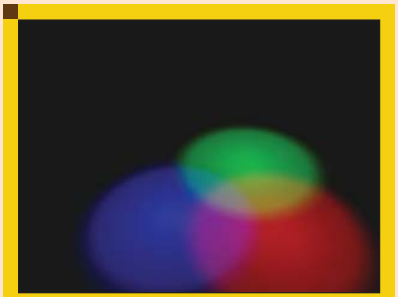
Érdekes játék a színekkel (*3 ábra*, *pov10.pov*), ha egy fehér síkot a három alapszínnel világítunk meg, ekkor ugyanis a színek találkozásánál az összes lehetséges *RGB (Red-Green-Blue)* színrendszerbeli kombináció előkerül, ha elegendően nagy felbontással dolgozunk (24 bites színmélység esetén minimum 16.7 millió pixel).



4. ábra Szpot fényforrás

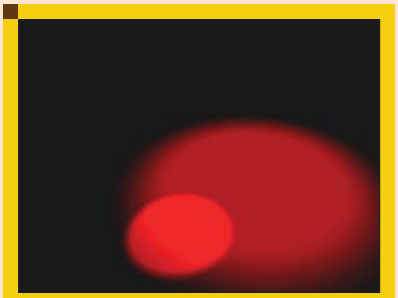


5. ábra Éles szélű szpot fényforrás



6. ábra Szpot fényátmenet

7. ábra Egy egységnyi magasan



8. ábra Két egységnyi magasan

1. lista – pov14.pov

```
light_source{
 <1,1,0> color Red
 cylinder point_at <1,0,0>}

light_source{
 <-1,1,0> color Red
 spotlight point_at <-1,0,0>}
```

2. lista – pov15.pov

```
light_source{
 <1,2,0> color Red
 cylinder point_at <1,0,0>}

light_source{
 <-1,2,0> color Red
 spotlight point_at <-1,0,0>}
```

A pontszerű fényforrás

A ***PoVRay*** alapesetben pontszerű fényforrást alkalmaz, ha elhelyezünk egyet a térben. Ez a tér minden pontja felé azt a színű fényt sugározza, amelyet meghatároztunk a számára. Ennek a típusnak nincs semmi különös beállítási lehetősége, a közös fényforrás módosítások érvényesek rá, amelyeket később részletezek. A fenti példákban ilyen pontszerű fényforrást alkalmaztunk.

A szpot fényforrás

A hidegtükrös halogénizzók mintájára beállíthatunk olyan fényforrást, amely mindössze egy meghatározott szögben sugároz fényt. Nézőpont hiányában teljesen biztos, hogy nem a megfelelő irányba fog fényt sugározni, így mindenképpen adjunk meg egyet a `point_at` kulcsszóval (***pov11.pov,*** *4. ábra*).

```
light_source{
 <0,1,0> color Red
 spotlight point_at <0,0,0>}
```

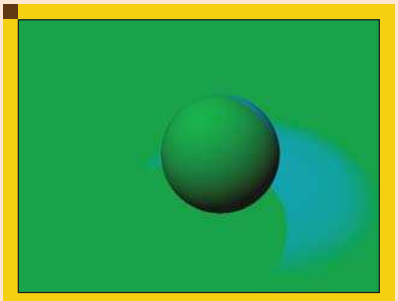
Ha nem határozunk meg egyéb értéket, akkor a szpot fényforrás 30 fokos szögben bocsát ki fénysugarakat, amelyhez még 15 foknyi fényátmenet tartozik. A fénykör paramétereit a `radius` , a `falloff` és a `tightness` kulcsszavakkal lehet beállítani. A ***radius*** adja meg közvetve a fénykör sugarát, mivel a mértékegysége fok, s ez a szög határozza meg a kibocsátott fénykúp nyílásszögét. Ha a `falloff` szöget nagyobbra vesszük, mint a `radius` értéke, akkor a kibocsátott fény a két szög által meghatározott tartományban teljesen elenyészik, így egy átmenet látszik. Ha a `falloff` értéke kisebb vagy egyenlő a `radius` értékével, akkor a fénykör éles peremmel szűnik meg, nem lesz szórt fényű udvara (***pov12.pov,*** *5. ábra*).

```
light_source{
 <0,1,0> color Red
 spotlight point_at <0,0,0>
 falloff 25}
```

Ha szpot fényforrással készítjük el a három színt sugárzó képünket (***pov13.pov,*** *6. ábra*), akkor a képen találunk olyan területeket, ahol a három szín azonos intenzitással fedi a neki szánt területet, és a találkozásoknál a szín csak a kettő (vagy három) szín tiszta találkozásából áll átmenet nélkül (zöld-vörös, zöld-kék, kék-vörös színfolt). A `falloff` által meghatározott szögön kívül már nem látunk fényt, a két szög között pedig láthatjuk a színek keveredését. Ha nem szeretnénk, hogy a szpot területen a lefedettség egyenletes legyen, akkor a `tightness` értékének (alapértéke 0, s a maximális értéke 100 lehet) megváltoztatásával, már a fénykúpon belül is csökken a fény intenzitása a szélek felé haladva.

Hengeres fényforrás

A hengeres fényforrás szinte teljesen azonos a szpot fényforrással, azonban a fénykör átmérője nem függ a fényforrás távolságától. A ***pov14.pov*** részlete (*7. ábra*) szerint a fényforrások 1 egységnyi magasan vannak a síktól, míg a ***pov15.pov*** részlete (*8. ábra*) szerint már két egységnyire.



9. ábra Árnyékvetés

10. ábra Az árnyék kikapcsolva

11. ábra Tárgyon át vetített fény

Látható, hogy a szpotfény átmérője megnövekedett, míg a hengeres fényforrás fényköre nem. Ezekkel a fényforrásokkal a lézerfényhez hasonló fényeket tudunk készíteni, amelyek nagy távolságokon át is megtartják az átmérőjüket. Fontos különbség azonban, hogy a fénysugarak nem párhuzamosak, hanem a megadott pontból indulnak, csak nem tudnak kilépni a paraméterekkel meghatározott hengerből (ez az átlátszó tárgyak fénytörésénél okozhat problémát).

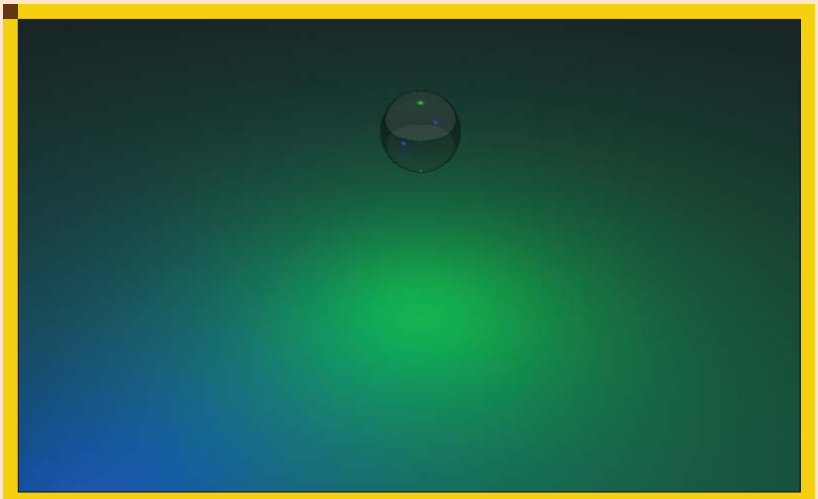
Párhuzamos fényforrás

Mind a három fényforrás fénye párhuzamossá tehető, amely akkor tesz jó szolgálatot, ha ilyen jellegű fényre van szükségünk; a napfény például gyakorlatilag párhuzamos fénysugarakból áll. Bármelyik fényforrással készíthetünk szinte párhuzamos fénysugarakat, ha nagyon messze tesszük a tárgyainktól (mint amilyen távol a Nap található a Földtől). Ezen megoldással mindössze annyi a gond, hogy a távoli fényforrás fénye gyér, meg kell sokszorozni, hogy hatása legyen. Egyszerűbb hozzáfűzni a fényforrás leírójához a ***parallel*** kulcsszót, és a fényforrás a megadott pont közelségének intenzitásával fog párhuzamos nyalábokat sugározni (***pov16.pov***).

```
light_source{
 <-1,0,0> color Green
 parallel point_at <-1,-1,0>}
```

Árnyékok

A sugárkövetés tálcán kínálja az árnyékok ábrázolását, mivel a tárgyak eltakarják a fény útját, ahol ebből következően árnyék keletkezik (***pov17.pov***, *9. ábra*). Az árnyék éles körvonallal rendelkezik, akkor is, ha a fényforrás elhalványuló része vet árnyékot. Egy kicsit természetellenesen hat, mivel a valóságban az elhalványuló fénykör a fényforrás nem pontszerű jellege miatt keletkezik, azonban ez túl sok számítási kapacitást igényel (lehet javítani ezen, de erről pár bekezdéssel később foglalkozunk majd).
Ha egy tárgy leírójában kikapcsoljuk az árnyékokat, akkor a fény keresztül tud haladni rajta mindenféle fénytörés nélkül, ugyanakkor a felülete el is nyeli azt (érdekes probléma az energiamegmaradás terén :) (***pov19.pov***, *10. ábra*).



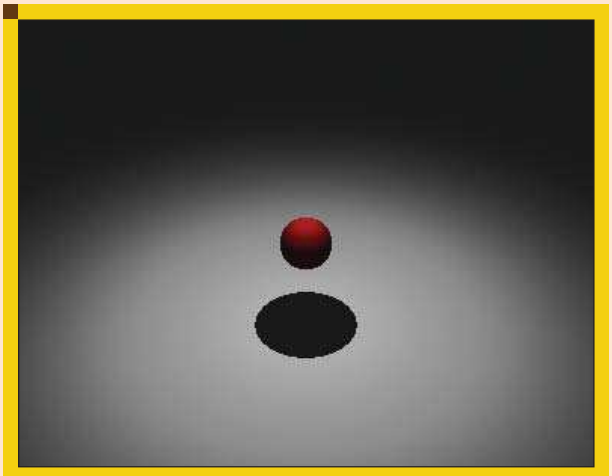
12. ábra Tárgyba épített fényforrás

```
sphere{
 <0,0,0>,1
 texture{pigment{color White}}
 ➥no_shadow}
```

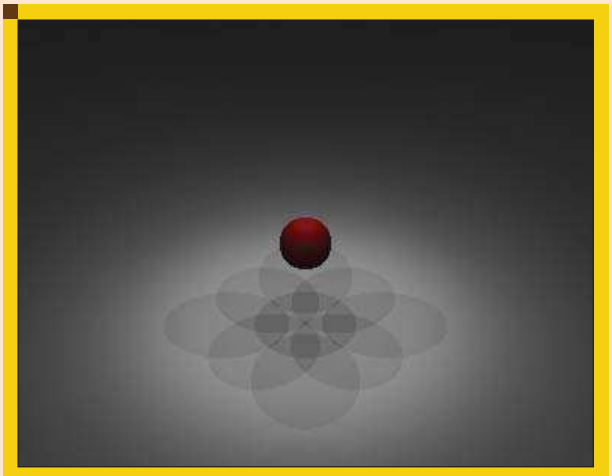
Ugyan ***nem*** az árnyékok kezelése módosítható a fényforrás leírójába írt `shadowless` kulcsszó használatával, a program készítői ezt a nevet adták ennek a kulcsszónak. Ebben az esetben a megadott fény szórt háttérfénnyé válik (***pov18.pov***), s ezzel egyfajta kellemes háttérfényt tudunk varázsolni különösebb problémák nélkül.

Alakzat vetítése

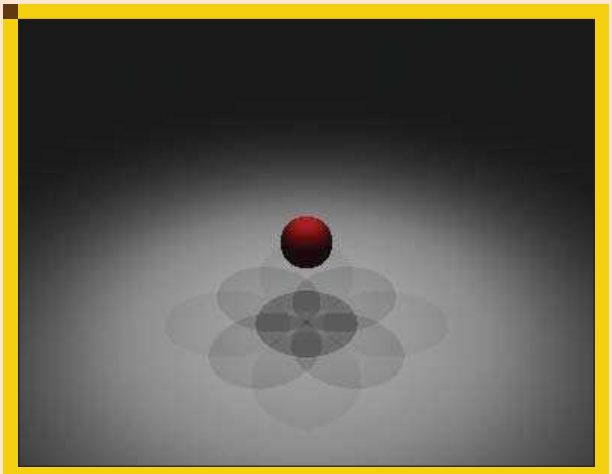
Az alakzat vetítését nevezhetjük akár negatív árnyéknak is, ugyanis a fény csak arra halad tovább, ahol az adott testen át tud menni. A megadott testnek nem kell átlátszónak lennie ahhoz, hogy a vetítés működjön. Egyszerűen csak a fényforrás leírójába kell helyezni a `projected_through` kulcsszó után azt a testet, amelyen át szeretnénk a fényt vetíteni. Ezzel a megoldással olyan összetett fényforrásokat tudunk készíteni, amelyre csak szükségünk lehet (érdekes megoldás például ködbe szöveget vetíteni :). Ha két gömböt összevonunk, majd ezen át vetítünk a szpotfényt (***pov20.pov***), akkor a (*11. ábra*) szpotfény egy része élesen ér véget, a többi része pedig a fény elhalványulásával.



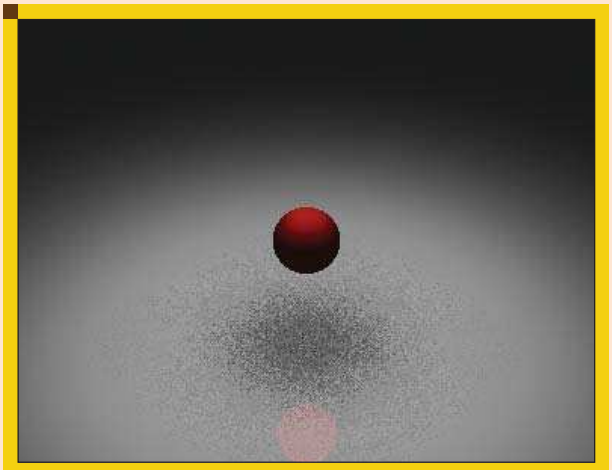
13. ábra Éles árnyék „szórt” fényben



14. ábra Éles árnyékok több fényforrással



15. ábra Éles árnyékok fényterülettel



16. ábra Fényterület szóródással

```
light_source{
 <0,2,0> color Green
 spotlight

projected_through{union{sphere{
➥<-0.4,1,0>,0.5}
➥sphere{<0.4,1,0>,0.5}}}}
```

Fényforrás építése

Készíthetünk olyan fényforrást, amely látható lesz számunkra, és a fény belőle árad majd. Ehhez egyszerűen egy testet kell készítenünk, amely villanykörteként fog funkcionálni, vagyis tartalmazni fogja a fényforrást. Ha ezt a testet hozzáadjuk a kiszemelt fényforráshoz a `looks_like` kulcsszó után, akkor a fény szabadon átjut rajta. Ha a test tömör vagy átlátszatlan, csak ekkor más fény hiányában tömör fekete lesz az elkészült képen, ezért lehetőleg üvegből készítsük el (***pov21.pov,*** *12. ábra*).

3. lista – pov24.pov

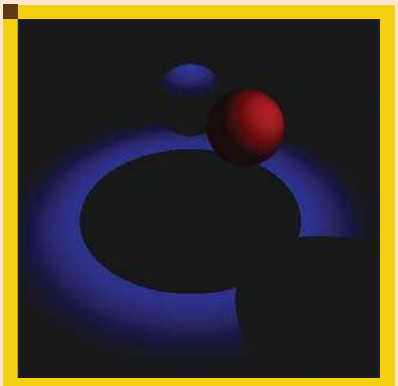
```
light_source {
 <0, 1, 0> color White
 area_light <1, 0, 0>, <0, 0,
 ➥1>, 3, 3

 spotlight point_at <0, 0, 0>
 tightness 0
 radius 40
 falloff 70}
```

```
light_source{
 <0,1,0> color Green
 looks_like{sphere{<0,1,0>,0.5
 ➥texture{T_Glass1}}}}
```

Ezzel a módszerrel sem tudjuk igazából „láthatóvá” tenni a fény forrását, hiszen nem vakítja el a kamerát, és „zavaró” becsillanások sincsenek a képen: mindössze a megadott testből árad a fény, ahogy a villanykörtéből szokott.

4. lista – pov25.pov

```
light_source {
 <0, 1, 0> color White
 area_light <1, 0, 0>, <0, 0,
 ➥1>, 3, 3
 adaptive 5
 jitter

 spotlight point_at <0, 0, 0>
 tightness 0
 radius 40
 falloff 70}
```



17. ábra Globális fények kikapcsolva

5. lista – pov26.pov

```
light_group{
 light_source {
  <0, 2, 2> color White
  spotlight point_at <0,0,0>}

sphere{
 <0,1,1>,0.5
 texture{pigment{color Red}}}

global_lights off}
```

Fényterületek

Ha igazi elmosott árnyékot akarunk kapni (tehát nem olyan éleset, mint a (***pov22.pov,*** *13. ábrán*), akkor pontszerű fényforrások helyett nagyobb kiterjedésű fényforrást kellene használnunk, ugyanis a valódi világban minden fényforrásnak van valamekkora kiterjedése. Minél nagyobb egy fényforrás, annál életlenebbek az árnyékok, ezáltal a szemnek kellemesebb képeket tudunk készíteni. A sugárkövetés működésmódja miatt egy fényforrással többet tartalmazó világ egy egységnyi idővel több ideig tartó renderelési időt is jelent. Ebből következően a nagyobb kiterjedésű fényforrásokat pontszerű fényforrások egymás mellé helyezésével tudjuk „modellezni".
Ha a fényforrások számát megtöbbszörözzük és elhelyezkedését megfelelően módosítjuk (***pov23.pov,*** *14. ábra*), akkor kicsit előbbre tudtunk jutni, viszont a kapott eredmény szerint úgy járunk, mint a focisták a stadionban: négy éles árnyékuk lesz (a képen látható gömbnek pedig kilenc).

```
light_source {
 <-0.5, 1, -0.5> color
 ➥<0.11,0.11,0.11>
 spotlight point_at <0, 0, 0>
 tightness 0
 radius 40
 falloff 70}

[...]
```

Mellékhatásként a fényforrások intenzitását annyiadrészére kell csökkentenünk, ahány fényforrást alkalmazunk, különben a kép túlvilágított lesz. Tehát ez a megoldás sem igazi, sok munkát jelent, és mégsem ad megfelelő minőséget: sokkal finomabban kellene a fényeket elhelyezni, amely sokkal több számítási kapacitást igényel a kép számolása közben.
A ***PoVRay*** képes saját hatáskörén belül megoldani, s erre az ***area_light*** kulcsszóval tudjuk utasítani. Ezzel létrehoz egy kétdimenziós tömböt fényforrásokból, amelyet használva gyakorlatilag azonos képet állítva elő (***pov24.pov,*** *15. ábra*).
Az előnye viszont az, hogy ezt egy sorban el tudjuk intézni, nem kell a fényforrások definícióját többször megismételni, amely a módosítást erősen megnehezíti. A ***PoVRay*** saját fényterületének előnye, hogy képes egy kicsit összekavarni a „fénysugarakat", és így tényleg olyan lehet az árnyék, mintha nagyobb kiterjedésű fényforrás világítana (***pov25.pov,*** *16. ábra*).
Ehhez mindössze két kulcsszót kell beleírnunk a fényforrás leírójába: ez a `jitter` és az `adaptive`. A `jitter` kell ahhoz, hogy az árnyék tényleg szórt legyen, az `adaptive` pedig a megadott szám hatványa szerint további fényforrásokat készít, ha szükséges.

Megvilágítási csoport

Ritka eset, de néha szükséges olyan megvilágítás, ahol egy-egy tárgyat világít meg egy fényforrás. Ebben az esetben egy csoportba gyűjtjük a fényforrásokat és azokat a tárgyakat, amelyeket ezek a fények fognak megvilágítani. Ezt a gyűjtést a `light_group` kulcsszó vezeti be. Ennek eredménye, hogy ***csak*** a csoportba foglalt testeket világítják meg azok a fényforrások, amelyek a csoportban szerepelnek. A többi testen nem lesz hatása ezeknek a fényforrásoknak, ellenben a globális fények hatással vannak a csoportba rendelt testekre. Ez utóbbit kikapcsolhatjuk a `global_lights off` parancs segítségével (***pov26.pov,*** *17. ábra*).

Fontosabb paraméterek

A ***PoVRay*** program használatához érdemes a parancssoros program néhány paraméterét ismerni, mivel így jelentős sebességnövekedést vagy minőségi ugrást tudunk elérni. Alapesetben a gyorsaságra van beállítva a program, tehát vannak olyan részek, ahol a kapott eredmény eléggé szegényesen néz ki.
Érdemes beállítani, hogy egy rettentő pixeles, de átlátható képet kapjunk a renderelés folyamatáról, így a durva hibákat azonnal ki tudjuk szúrni, és nem kell megvárni, amíg több perc munka után észrevesszük, hogy hiányzik egy test a képről. Ehhez a `+SPn` és a `+EPn` paramétereket kell megadni, ahol a `+SP` a első lépés pixelben való méretét kéri számszerűen, a `+EP` pedig az utolsó lépést. Érdemes 32 pixellel kezdeni és 4-8 pixel körül végezni ezzel a finomítással, mert 2-4 pixel körül már eléggé részletesen látszik minden szükséges test.
A kész kép minőségét a `+Qn` paraméterrel tudjuk befolyásolni, ahol a szükséges szám 0 és 9 között lehet közönséges sugárkövetésnél, és 10 vagy 11 lehet a speciális sugárkövetést választva.
A következő rész témája a primitív testek és a testek közötti kölcsönhatás lesz, ahol a kölcsönhatás alatt a különféle „halmazműveleteket" értjük.

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

A PovRay projekt honlapja
➲ http://www.povray.org

Az SVG világa (2. rész)
Alkotni öröm

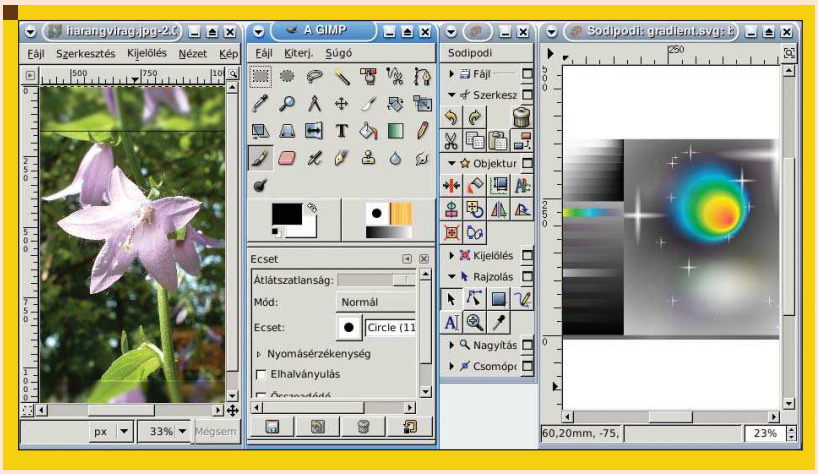
Az előző részben az SVG elméleti alapjait és a gyakorlati hasznosítási lehetőségeit ismertük meg, így hát itt az ideje, hogy a tettek mezejére lépjünk! Az induláshoz nem kell sok, mindössze egy operációs rendszer, lehetőleg valamelyik Linux-variáns. Valóban alkotni fogunk, mégpedig SVG fájlokat. Igazából maga az alkotás művészi munka, így precízen fogalmazva azokat a programokat vesszük szemügyre, melyek a művész keze ügyében jól jöhetnek, ha éppen SVG formátumban szeretné kifejezni magát. Tehát akkor nézzünk bele a szerszámosládába, hogy mik is azok az eszközök, mellyel felszerelkezve SVG művészeti műhelyt nyithatunk.

A programokról általában

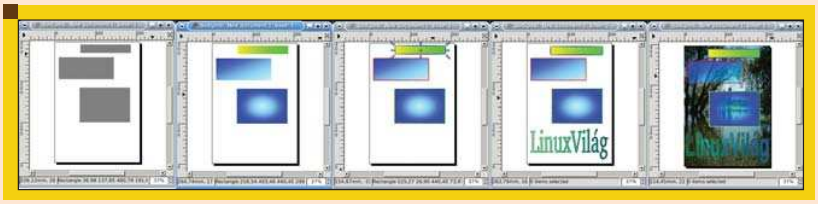
Mint azt a szabad szoftverek világában megszokhattuk, az itt bemutatott programok letölthetőek az internetről, sőt a forráskódjuk is hozzáférhető. Általában több operációs rendszerre is elérhetőek, szóval ez a cikksorozat célközönsége még véletlenül sem korlátozódik a *Linux* felhasználókra. Ezek a szoftverek teljesen jól megértik egymást az *SVG* formátumon keresztül, a teszt során egyszer sem fordult elő, hogy az egyik programban elkészített fájl ne lett volna átvihető a másik programba, de apróbb szépséghibák néha előfordultak. Ezek leginkább a programok eltérő tudásszintjéből adódtak. Tehát a választás teljesen szubjektív, győztes sem lesz, mindenki válasszon saját ízlésének megfelelően. Amint az ablakkezelők közül bőségesen válogathatunk egy disztribúcióban, úgy itt is ránk van bízva a döntés. Persze a lényeg, hogy rosszul nem is lehet dönteni.

Sodipodi

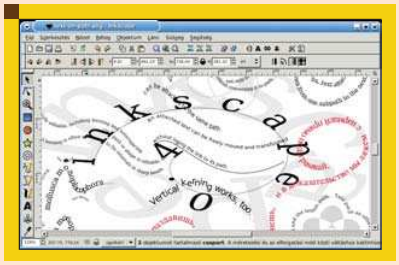
A linuxos pixelgrafika koronázatlan királya, a *Gimp* ikertestvére ez a program. Ez hallatlan előny, hiszen aki kicsit is járatos a *Gimpben*, a megszokott logikát, felépítést ebben megtalálja.

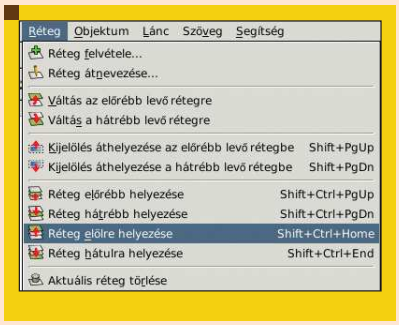


A *GTK* grafikus könyvtárat használja. Nem elhanyagolható szempont, hogy mindenféle *Linuxhoz* és *Windowshoz* is van bináris csomag a ➲ *www.sodipodi.com* címen. Azonban a legtöbb rendszeren a csomagok között megtalálható, el se kell látogatnunk a szoftver weboldalára. Elindítva a programot a már említett *Gimp*-szerűségen túl azonnal szembeötlik, hogy a magyar fordítás nem teljes, legalábbis a *Debian Sidben* lévő 0.34-es, legújabb változatban nem. Remélhetőleg hamarosan lépést tud tartani a fordítás a program fejlesztési ütemével. Mint bármely más vektorgrafikus szerkesztőben, itt is objektumokkal dolgozunk és ezeket az objektumokon lehet műveleteket végezni és a tulajdonságaikat állítani. Azok kedvéért, akiknek nem ismerős a *Gimpből* a program felépítése, erről is szót ejtünk, hiszen alapvetően eltér az egy ablak – egy program filozófiától. Minden működési egység külön ablakban van: a főmenü, a párbeszédablakok és az egyes dokumentumok. Aki ezzel a felépítéssel most találkozik először, úgy érezheti, hogy képernyőjét elárasztják az ablakok. Célszerű



például az egész programot új virtuális munkaasztalon indítani így egy egész képernyőn kényelmesen el lehet rendezni a kezelőfelületet és a dokumentumokat. Ez a fajta felépítés hatalmas rugalmasságot biztosít az egy ablakos modellhez képest.
Kezdjünk el rajzolni!
Fontos tudni, hogy mint a ***Gimpben***, a műveletek nem egy klasszikus menüből érhetők el, hanem szinte mindent lenyíló menükben találunk, amit a jobb egérgombbal csalogathatunk elő. Kezdjük egy téglalappal. Ezt fogjuk aztán tovább formázni. Tehát az üres dokumentumon hívjuk elő a helyi menüt. Ott pedig a ***Drawing Mode/Rectanglet*** válasszuk ki. Most már rajzolhatjuk a téglalapokat.
Amikor ezt meguntuk, hívjuk elő újfent a helyi menüt, de most már a téglalapon állva. Próbálgassuk a lehetőségeket, mondjuk a ***Shape/Fill Settings***-t. Itt egyetlen színnel és színátmenetekkel tölthetjük ki az alakzatunkat. Egy megadott névvel definiált színátmenetet tetszőlegesen sok elemhez hozzárendelhetünk és ha módosítjuk azt, akkor az összes elemnél módosulni fog. Ha ezzel készen vagyunk, válasszuk a ***Sodipodi*** főablakában az ***Object/Stroke settingst***. Itt a legkülönfélébb keretekkel díszíthetjük elkészült alakzatainkat.
Hiányzik a szöveg a képről?
A ***Drawing Mode/Text*** segítségével szöveget is tehetünk az oldalra és a szöveget éppen úgy bármilyen kitöltéssel és kerettel elláthatjuk. Még mindig az az érzésünk, hogy valami nagyon hiányzik ahhoz, hogy tényleg alkotni lehessen? Így igaz. Pixelgrafikus képekre az esetek döntő többségében szükség van. Azokat is beszúrhatjuk a főmenüben a ***Fájl/Import bitmap or SVG document*** alatt. Ha megjelent a kép az oldalon, már csak arra van szükség, hogy az objektumok egymáshoz viszonyított helyzetét beállítsuk, mert tegyük fel, hogy háttérnek szeretnénk használni. Ehhez a ***Selection/Lower selected objects to bottom*** opciót kell kiválasztani.





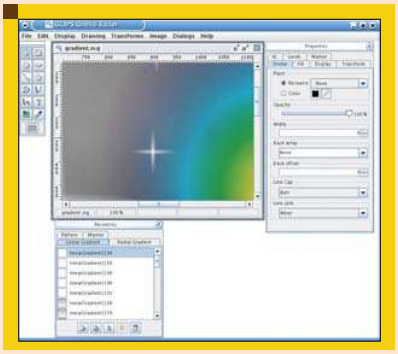
Ezután ha a lap méretére igazítjuk a képet, készen is vagyunk. Próbáljuk ki, hogy a színátmenetek egyik színének átlátszóságot adunk! Érdekes hatása lesz, ha van háttérkép.
A ***Drawing*** mód alatt számtalan lehetőség rejtőzik, természetesen nem téglalapokból kell megalkotni egy bonyolult rajzot. A rövid útmutatóban azért csak angol menünevekre hivatkoztam, mivel sok közülük még nincs lefordítva. Ha pehelysúlyú programra vágyunk, ezzel érdemes megbarátkozni, a ***Sodipodi FAQ*** (gyakran feltett kérdések) szerint akár 32 MB RAM-mal ellátott számítógépen is futtatható.

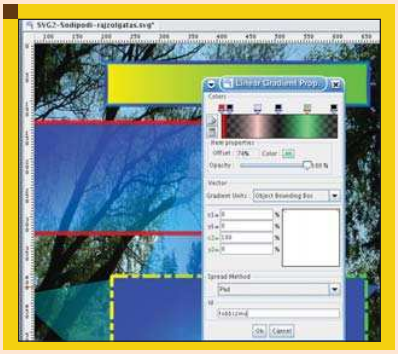
Inkscape

Akik teljesen idegennek érezték az ablakok forgatagát az előző programban, de a ***GTK***-alapú programok megfelelőek, akkor ez jó választás lehet. A technikai részletekben járatosabb olvasóknak érdekesség, hogy az ***Inkscape*** a ***Sodipodi*** „forkja” (értsd: a meglévő kódbázison új projekt indult). A fejlesztés ***C***-ben zajlott, de az ***Inkscape*** ***C++***-ra váltott. A menürendszere és egész logikai felépítése sokkal inkább hasonlít egy átlagos windowsos képszerkesztőéhez, mint a ***Gimp/Sodipodi*** pároshoz.
Ezt is éppúgy telepíthetjük a disztribúciónk csomagkezelőjéből, de a honlapról (➲ ***www.inkscape.org***) a ***Sodipodinál*** felsoroltakon túl még ***MacOs X***-re is elérhető. Tudásban sincs lemaradva, sőt, professzionálisabb benyomást kelt, az előzővel ellentétben van hozzá részletes angol, francia és spanyol dokumentáció a ➲ ***http://www.inkscape.org/doc/index.php*** címen. A felhasználói leírásban azt olvashatjuk, hogy az ***Illustratorhoz***, a ***Freehandhez*** és a ***CorelDrawhoz*** hasonló képességekkel rendelkezik.
Az itt bemutatott másik két szoftvertől eltérően nagyon sok formátumban képes menteni, ***PDF***-ben, az ***Illustrator*** saját formátumában, PostScriptben és még sok másban. Ha nem csupán a webre szánjuk az elkészült alkotásokat, ez egy fontos szempont a választásnál.
Miután a ***Sodipodiban*** rajzolgattunk, itt vegyük azt szemügyre, hogy mi az, amiben többet nyújt ez a program.
A ***Sodipodi*** utolsó munkafázisát elmentve, itt folytathatjuk a szépítgetést. Komoly grafikusnak a rétegek kezelése kulcskérdés, az ***Inkscape***-ben pedig van rétegkezelés. De mire is jók a rétegek?
Egy bonyolult, szinte átláthatatlanul sok objektumot tartalmazó rajzon nem árt, ha logikailag csoportosított rétegeket tudunk zárolni és elrejteni, így egy adott munkafázisban csak azt látjuk, amivel ténylegesen dolgozunk.
A szöveg illesztése láncra is nagyon látványos lehetőség, érdemes kipróbálni. Görbe vonalakra, spirálra és szinte bármire lehet folyatni szöveget.
Az ***inkscape.org***-on lévő példafájl jól mutatja az ebben rejlő lehetőséget, amit az első Inkscape-pel készített képen láthatunk. Aztán ha még nem ámulnánk, akkor próbáljuk csak ki a ***Lánc/Bitkép*** vektorizálását! Egy papírra felvázolt beszkennelt tollrajzot egészen jó hatásfokkal alakít át.

Glips Graffiti SVG Editor

Ha eddig a hasonlóságokat elemeztük, most valami egészen új jön.
Az ***Apache Foundation Java*** nyelven írt egy ***SVG*** szerkesztőprogramot. Ennek aztán több vonzata is van. A gépen mindenképpen kell legyen java





futtatókörnyezet, melyet a ➲ *http://java.sun.com/* címről tölthetünk le. Ez a leglomhább alkalmazás a mezőnyben. A gép, amin futott, *Athlon XP +1600* processzorral és 768 MB RAM-mal volt felszerelve és a bonyolultabb grafikáknál szinte már zavaróan sokat kellett várakozni. Cserébe szinte bármin fut a kávéfőző kivételével, amire elérhető a java virtuális gép 1.5-ös változata. Használatához le kell töltenünk a programot a ➲ *http://sourceforge.net/projects/glipssvgeditor/* címről. Miután kicsomagoltuk a letöltött *.zip* fájlt, a létrejött könyvtárban egy héjprogramot és egy *.bat* fájlt találunk. Ezzel *Unixon* és *Windowson* könnyűszerrel elindítható, ha a java benne van az elérési útban. Ez egy rendesen feltelepített java futtatókörnyezetnél nem lehet probléma. Az indítás után ez a kép fogad bennünket:
Az *Inkscape*-hez hasonlóan az egyablakos logikát követi, grafikus felületnek a *Java Swingjét* használja, így a megszokott *KDE*-s és *Gnome*-os kinézettől el fog ütni. Tudása az *Inkscape*-pel összemérve csekély, hiányzik belőle a rétegkezelés és még több dolog, azonban a webes grafikázáshoz bőségesen elegendő még így is. Azért akad, amiben felülmúlja mindkét vetélytársát: lehet vele tetszőlegesen sok színből színátmenetet készíteni, nem korlátozódik kezdő és zárószínre a beállítás lehetősége. Igaz ugyan, hogy az *Inkscapeben* is lehet ilyet csinálni, azonban sokkal inkább kézre áll a létrehozás lehetősége itt. A *Sodipodiban* megnyitva a többszínes színátmenetet, az elemre azt írja ki, hogy *„Ismeretlen elem”*. Meglepő a dokumentáció teljes hiánya: leírást sem a weboldalon, sem a letöltött *.zip* fájlban, sem magában a programban nem találunk.

Viszont a java programozók el vannak kényeztetve: a projekt *Sourceforge* oldaláról részletes fejlesztői *JavaDoc* érhető el.

Összehasonlítás

Egy objektív szempont szerint hasonlítjuk össze a három szoftvert, a memóriafoglalásuk alapján. Látható, hogy a tudásszintjük igen eltérő, így sebességi és erőforrásigények alapján beskatulyázni nem érdemes egyik alkalmazást sem. Viszont a webes grafikázáshoz mindhárom használható, így ez akár támpontot is adhat a választáshoz. A teszt során egy internetről származó bonyolult, de mindhárom szoftverrel tökéletesen megnyitható *SVG* fájlt nyitottunk meg és a `ps u` parancs *RSS* mezőjét vizsgáltuk.
A tesztkép letölthető a ➲ *http://openclipart.org/clipart/people/fata_fatina_architetto_f_01.svg* címről.

Program neve	Sodipodi	Inkscape	GLIPS Graffiti
Memóriaigény (Kbyte)	10432	30380	53560

Szerény kiépítettségű gépen mindenképpen figyelembe kell venni ezeket az adatokat. Nyilván komoly munkánál csak a tudásszint és az elvárások dönthetnek.

Az elkészített mű használata

Annak ellenére, hogy ez a cikksorozat az *SVG*-ről szól, érdemes megfontolni, hogyan is használjuk fel az elkészült grafikáinkat. Amennyiben webre szánt alkotásokról van szó és a célközönség nem speciális réteg, mindenképpen *PNG*-ben vagy *JPG*-ben is rakjuk fel a webre, hiszen így bárki meg tudja nézni. Az összes program tud menteni pixeles formátumban. Csak ha valami nyomós oka van és nem gond, ha böngészőspecifikus az oldalunk, akkor használjuk kizárólag az *SVG*-t. Ebben az esetben viszont feltétlenül *.svgz* formátumot használjuk, ami annyiból áll csak, hogy zippel betömörítjük a kész *.svg* fájlt. Töredékére csökken a fájl mérete így, mert a tisztán szöveges *XML* nagyon jó hatásfokkal tömöríthető.
Az alkotás általában nem öncélú tevékenység. Szeretnénk, ha mások is meg tudnák nézni az elkészült munkákat. Akár önálló műalkotásoknak szánjuk, akár a weboldalunk szerves részeivé tesszük az *SVG* fájlokat, a támogatás elengedhetetlen. A következő részben *SVG*-t tudó böngészőkről, böngészőmodulokról, munkaasztali ikonkészletekről lesz szó.
Azért az alkotást sem fogjuk elhanyagolni, az *SVG*-vel felokosított *MMS* kulisszatitkaiba is bepillantást nyerünk. Ez a három program a novemberi számig úgy érzem, hogy grafikai fronton gondoskodik a meglepetésekről. A cikkben alapvető funkciókat érintettünk, az *Inkscape*-re azonban különösen igaz, hogy bőven tartogat még meglepetéseket. Tehát kalandra fel, hiszen alkotni öröm!

Novák Áron
(aaron@szentimre.hu)
BME-VIK-es gólya, műkedvelő rendszergazda. Jelenleg leginkább a NetBeans-szel és mindenféle hordozható eszközzel foglalkozik, legalábbis mindazokkal amelyeket meg lehet szólaltatni Linux alatt.

KAPCSOLÓDÓ CÍMEK

A forráskódok:

- ➲ http://ovh.dl.sourceforge.net/sourceforge/sodipodi/sodipodi-0.34.tar.gz
- ➲ http://ovh.dl.sourceforge.net/sourceforge/inkscape/inkscape-0.42.tar.bz2
- ➲ http://ovh.dl.sourceforge.net/sourceforge/glipssvgeditor/GLIPSGraffitiSrc1.3.zip

A Horde keretrendszer és az IMP Webmail rendszer

Bizonyára sok rendszergazda belefutott már a feladatba, hogy egy levelezőszerver karbantartását vegye a nyakába. Ilyenkor sokszor előfordul, hogy bár a levélkiszolgáló kifogástalanul működik, sok felhasználó mégsem tudja, hogyan is állítsa be a levelezőprogramját...

Ráadásul a munkaállomások esetleges rendszerösszeomlásai és újratelepítései miatt pedig végeláthatatlan munkává válhat a felhasználók levelezéséről rendszeresen biztonsági mentést készíteni. A rendszergazda ilyenkor kezd az *IMAP* protokoll felé kacsintgatni, mellyel a teljes hálózat levelezését egy központi helyen tudja tárolni, így nagyságrendekkel könnyebben kezelhetővé válik a probléma.

A kiérdemelt nyugalom helyett azonban máris újabb felhasználó jelenik meg, aki szeretné a levelezőt otthonról is használni és de nem óhajt lemondani arról sem, hogy egy nyaralás alkalmával egy külföldi Internet kávézóból is elérje nemcsak a levelezését, hanem a levelezési partnerek címjegyzékét, vagy akár a személyes munkanaptárát is. Ezen a ponton a rendszergazda összeráncolja szemöldökét és elhatározza, hogy a levelezést egy web alapú levelezőre állítja át, mely pont ezekre a problémákra született megoldásként.

Napjainkban ha ***web alapú levelezőt (webmailt)*** szeretnénk használni több választási lehetőséggel is találkozhatunk a szabad szoftverek között. Egyszerűsége miatt az egyik legelterjedtebb levező a ***Squirrel Mail***, mely könnyen beállítható és eléggé puritán felülettel rendelkezik, így szerényebb kapacitású szerverekre is ajánlható.

Másik alternatíva lehet a manapság igencsak népes ***csoportmunka szerverek*** (***groupware***-ek) családjából választani egyet, melynek a levelezőprogramját fogjuk be a feladatra. Amennyiben viszont egy professzionális lehetőségekkel rendelkező levelezőt szeretnénk használni, mely messzemenőkig személyre szabható, érdemes közelebbről megismerkednünk a *Horde* keretrendszer által nyújtott szolgáltatásokkal.

A Horde keretrendszer

A ***Horde keretrendszer*** (***Horde framework***) egy objektum orientált, *PHP*-ben íródott alkalmazás. Valójában egy függvénykönyvtárral van dolgunk, mely a hozzá illeszkedő modulokat azok interfészein keresztül egy közös keretbe foglalja. Funkciója azokat a feladatokat megvalósítani, melyekre a *Horde* integrált alkalmazásrendszer működésekor minden modulnak szüksége lehet. Ilyen feladat például a felhasználói azonosítás és hozzáférés vezérlés, hogy a különböző szolgáltatási modulok használatához a felhasználónak elegendő legyen egyszer belépnie a rendszerbe, majd azt egységes egészként láthassa.

A *Horde* fejlesztése 1998-ra nyarán kezdődött. Az eredetileg ***Internet Messaging Program (IMP)*** néven elindult webes levelező rendszer akkoriban néhány *PHP*-ben készült állományból állt, mely alig kétezer soros volt a dokumentációval együtt. Később ebből a kezdeményezésből nőtt ki a *Horde* keretrendszer, melyben az *IMP* már egy modulként kapott helyet. A jelenlegi rendszer *PHP* forrása dokumentáció nélkül is közel kilencvenezer soros, a *Horde* keretrendszer pedig mára egy az elfogadott szabványokat messzemenőkig támogató rendszerré vált. Az általa támogatott szabványoknak és interfészeknek, valamint a objektumorientált *PHP* nyelvnek köszönhetően egy jól integrálható és könnyen továbbfejleszthető környezetté vált. Több oktatási intézmény is használja, főként web alapú levelező szolgáltatás nyújtására.

A Horde rendszer komponensei

Az ***IMP (Internet Messaging Program)*** volt az első *Horde* alkalmazás, melyből végül maga a keretrendszer is kinőtt. A *IMP* egy webmail alkalmazás, mely képes *POP3(S)* és *IMAP(S)* postafiókok kezelésére.

A ***Turba*** címjegyzék kezelő modul az *IMP* webmail egyik leghasznosabb kiegészítője. Többek között képes *LDAP* és *SQL* alapú címbejegyzéseket is kezelni.

Az *Ingo* egy levélszűrési szabályokat kezelő rendszer az *IMP*-hez. A szűrőket a szabványosított levélszűrési szabályokat leíró *Sieve* nyelven tárolja. Ezek tárolását a *Horde* keretrendszer valósítja meg, de az *Ingo* képes ezekből a szabályokból a felhasználók könyvtárába ***procmail*** szűrőszabályokat leíró állományt is készíteni.

A *MIMP* az *IMP* alkalmazás egy hordozható és korlátozott *HTML* megjelenítésre alkalmas mobil eszközökre leegyszerűsített változata. Az alapvető levelezési funkciók teljes mértékig elérhetőek benne.

A *Passwd* modul feladata, hogy a beállított interfészen keresztül lehetőséget adjon a felhasználónak a jelszóváltoztatásra. Támogatja a jelszavak tárolását és megváltoztatását többek között *LDAP*, *SMB*, *SQL* interfészeken, vagy akár saját szkripttel is. Így eldönthetjük, hogy a felhasználókat egyéb rendszereinkkel közösen, vagy azoktól külön szeretnénk-e hitelesíteni.
A *Horde* rendszer csoportmunkát támogató moduljai akár több felhasználó számára is elérhető szolgáltatásokat tudnak nyújtani. Ilyen a *Kronolith* naptárprogram, a *Mnemo* feljegyzéskészítő, a *Nag* feladatkezelő valamint a webes könyvjelzőket kezelő *Trean* modul.
A *Horde* rendszer web alapú állománykezelője, a *Gollem* képes adatbázisszerver, *FTP*, vagy valós állományrendszeren kezelni az adatállományokat.
Végül érdemes megemlíteni a fejlesztést támogató *Chora* web alapú *CVS* böngészőt, valamint a *Whups* webes hibakövető rendszert.
A felsoroltakon kívül még számos másik kiegészítő modul is elérhető, melyekről a *Horde* weblapján lehet tájékozódni.

A Horde keretrendszer beszerzése

A tesztrendszer alapját nálam egy *Debian Sarge* rendszer alkotta, melynek kiszolgáló csomagjai tökéletesen megfelelnek a webes levelező beüzemeléséhez.
A *Horde* keretrendszer működéséhez egy web kiszolgáló és a hozzá illesztett *PHP* értelmező 4-es (legalább 4.3.0), vagy 5-ös verziója szükséges. Ehhez a *Debianban* található *Apache* és *PHP* csomagokat használtam.
Szükségünk lesz továbbá egy adatbázis kiszolgálóra.
A *Horde* támogatja az elterjedtebb kiszolgálókat, egyszerűsége miatt én a *MySQL*-t választottam.
Alapvető fontosságú egy *IMAP* képes levelező kiszolgáló, ebből a választásom a *Courier-IMAP*-ra esett, mivel ez támogatja a *Maildir++* szabványt, mellyel könnyedén lehetséges a felhasználók levelezését kvótázni.
A rendszerben a felhasználók hitelesítését egy központosított *LDAP* kiszolgáló az *OpenLDAP* végezte, hogy közös adatbázisból lehessen többek között a helyi felhasználókat, valamint később a *Samba* ügyfeleket is hitelesíteni. Szintén *LDAP* adatbázisban tároltam a felhasználók teljes nevét, valamint a levelezési címét, melyek

```
Vezetéknév Keresztnév
➥<vezetéknév.keresztnév@maildomain.hu>
```

formában kerülnek bele a kimenő levelek feladó mezőjébe.
A *Linux* terjesztések általában tartalmazzák a *Horde* keretrendszert, de mivel gyorsan fejlődő alkalmazásról van szó, mindenképpen javaslom a forrásból való telepítést.
Letöltéskor több lehetőségünk van kiválasztani a megfelelő forrást. Aktív fejlesztéshez közvetlenül a *CVS* verziókezelő rendszerből tölthetjük le a legutolsó változatot. Ha fontos hibát szeretnénk orvosolni jelenlegi rendszerünkben egy újabb verzióval, akkor ajánlanám a fejlesztés *HEAD* ágát, mely egy konzisztens állapotú napi mentése a *CVS*-ben tárolt forrásnak. Egyéb esetekben pedig a legutolsó stabil verzió használata a javasolt, ebben a cikkben ennek a verziónak a telepítését mutatom be.

A telepítés menete

Miután az összes kiszolgáló működőképes, belekezdhetünk a *Horde* telepítésébe. A modulokat és keretrendszert a *Horde* honlapjáról érhetjük el.
Első lépésként telepítsük fel a *PEAR* függvénykönyvtárat (ez nagy valószínűséggel megtalálható a *Linux* disztribúciónkban), majd tegyük fel a szükséges moduljait a

```
pear install -o Log Mail Mail_Mime DB Date File
```

paranccsal.
Ezután a webkiszolgáló főkönyvtárában hozzuk létre a *horde* könyvtárat, majd ide csomagoljuk ki a *Horde* keretrendszert. A létrejövő könyvtárnevet nevezzük át úgy, hogy a verzióinformációt ne tartalmazza. Ez *Debian* alatt így néz ki:

```
cd /var/www
wget ftp://ftp.horde.org/pub/horde/
➥horde-3.0.5.tar.gz
tar zxvf ./horde-3.0.5.tar.gz
mv horde-3.0.5 horde
```

Harmadik lépésként a *Horde*-hoz mellékelt konfigurációs állományokat nevezzük át, hogy az állománynév végén ne szerepeljen a `dist` kiterjesztés:

```
cd horde/config/
for f in *.dist; do cp $f 'basename $f .dist';
➥done
```

Következőként hozzunk létre az adatbázis táblákat *Horde* főkönyvtárában levő *SQL* szkripttel. Fontos, hogy ez a szkript tartalmazza azt a jelszót, mellyel később elérhetőek lesznek a *Horde* táblái az adatbázis kiszolgálón, ezért ezt mindenképpen módosítsuk! A könnyebb követhetőség kedvéért én a titok jelszót használtam a `horde` táblához, a felhasználónév szintén `horde` volt:

```
cd ../scripts/sql
vi create.mysql.sql
mysql -u root -p < create.mysql.sql
```

Ötödik lépésként ellenőrizzük a *Horde* tesztoldalát!
Én a telepítést az aktuális gépen végeztem, így a következő címet írtam a böngészőbe:

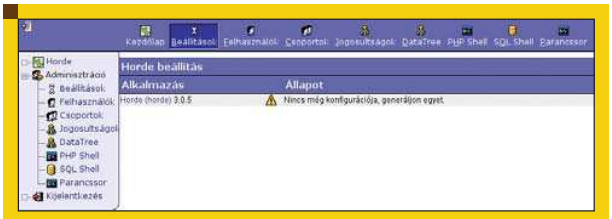
```
http://localhost/horde/test.php
```

Ezen az oldalon információkat találunk arról, hogy a működéshez milyen beállítások és kiterjesztések hiányoznak még. Érdemes átfutni a listán. A hiányzó *PHP* modulokat a *php.ini* állományban elhelyezett

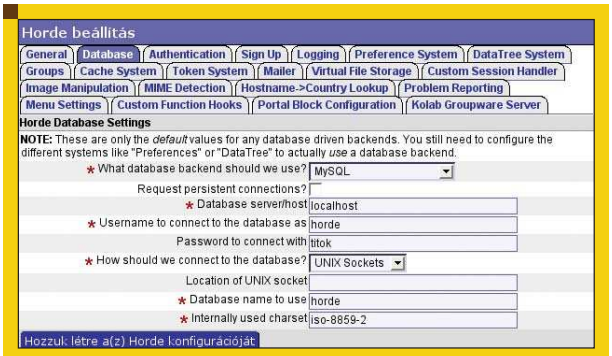
```
extension=modulnév.so
```

sorokkal tudjuk betölteni (feltéve, hogy csomagként feltelepítettük már).
A *Horde* teszt oldala javasolja a *PHP* memória korlát használatának kikapcsolását. Aki nem szeretne azzal

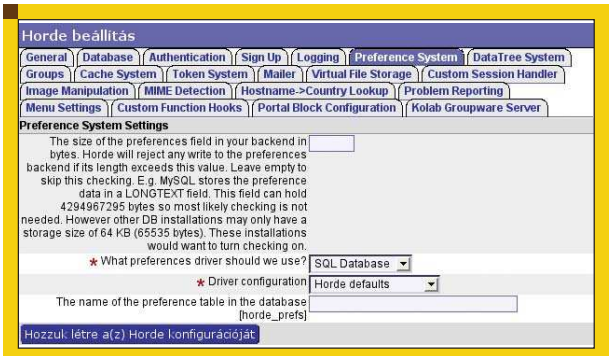
1. ábra Az első bejelentkezés



2. ábra A beállítások menü



3. ábra Az adatbázis kapcsolat paramétereinek beállítása



4. ábra Az alapbeállításokat tároló rendszer engedélyezése

kísérletezni, hogy rendszere milyen korlát mellett működik stabilan, nyugodtan kapcsolja ki. Az állomány feltöltési limitet szintén célszerű helyesen megválasztani a ***php.ini*** állományban, mert értelemszerűen az ennél nagyobb csatolásokat a rendszer nem fogja felengedni a kiszolgálóra.

1. lista A prefs.php állomány

```
$_prefs['timezone'] = array(
    'value' => '',
    'locked' => false,
    'shared' => true,
    'type' => 'select',
    'desc' => _("Your current time zone:")
);
```

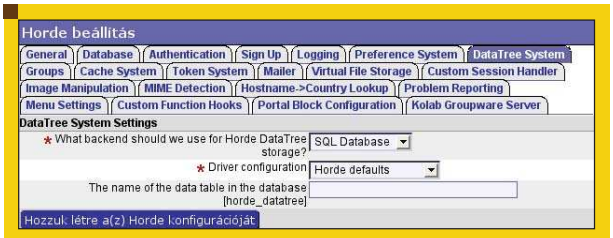
További fontos információ a teszt oldalon a ***PEAR*** modulok állapota. Amelyeket még nem telepítettük, de szükségünk lesz rá, azt a

```
pear install modulnév
```

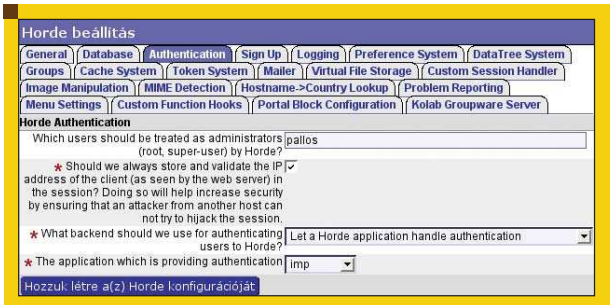
paranccsal telepíthetjük.
Miután elkészültünk, a webkiszolgáló főkönyvtárában elhelyezett ***horde*** könyvtárra nagyon sokszor lesz szükségünk. A cikk további részében erre a könyvtárra ***horde***/ néven fogok csak hivatkozni.
Aki pedig szeretné esetleg az ***Apache***-ot, vagy a ***PHP***-t saját maga forrásból telepíteni, a ***horde/docs/INSTALL*** állományban talál hozzá segítséget.
Miután elkészültünk a telepítéssel eljött az ideje, hogy végre kipróbáljuk, hogyan is néz ki mindez egy böngészőben. Töltsük be a

```
http://localhost/horde/
```

oldalt a böngészőbe. Ha mindent jól csináltunk, akkor az *1. ábrán* látható kép fogad minket, amikor először bejelentkezünk adminisztrátorként.
Ha idáig eljutottunk, akkor érdemes tudatosítani magunkban, hogy a munka java még csak most kezdődik a ***Horde*** beállításaival, ugyanis kismillió beállítási lehetőséggel és trükkel tudjuk a saját rendszerünkhöz és elképzeléseinkhez idomítani a keretrendszert.
A beállításokat a bal szélen megjelenő menü Adminisztráció/Beállítások menüpontjában tehetjük meg (*2. ábra*).
Egyelőre csak a ***Horde*** keretrendszerhez kapunk lehetőséget konfigurációt készíteni, de később a további komponenseknek is itt tudjuk elkészíteni a konfigurációs állományát.



■ ***5. ábra*** A DataTree rendszer engedélyezése



■ ***6. ábra*** A felhasználói hitelesítés (Authentication) beállítása

A beállítások végignézésére érdemes rászánni némi időt ugyanis ez az első alkalom, amikor rádöbbenünk, hogy mennyi mindent tud a rendszer.

A Horde beállítása

A beállítások elvégzése előtt érdemes egy picit elidőzni a *Horde* alapbeállításokat tartalmazó rendszerének, a *Preferences System* működésénél.

A *Preferences Sytem* feladata a felhasználók egyedi beállításait tárolni. Ilyen beállítások lehetnek például, hogy mikor lépett be utoljára az illető, milyen nyelven, milyen időzónában használja a rendszert és még számos másik hasonló paraméter.

A *Preferences System* valójában két adattárból dolgozik. Az egyik a *horde/config/prefs.php* állomány, a másik pedig az általunk beállított dinamikus adatbázis, mely lehet például *SQL*, vagy *LDAP* szerver is. A *Horde* egy beállítás értékét először az adatbázisban keresi, majd ha ezzel nem jár sikerrel, akkor a *prefs.php* állományban definiált értéket használja. A sorrendre figyeljünk, mert csúnyán meg tud viccelni ilyen módon minket a rendszer. Érdemes arra is figyelnünk, hogy a *Preferences* rendszer csak belépéskor kerül meghívásra, tehát ha valamit módosítunk benne, akkor újra be kell lépnünk a rendszerbe.

Ha vetünk egy pillantást a *prefs.php* állományra láthatjuk, hogy abban *PHP* szintaxisnak megfelelő tömbök találhatóak minden beállítási paraméterre. Ezek tanulmányozása sok időt vesz igénybe, de a rendszer viselkedésének megismeréséhez fontos átnézni őket. Az állományban elég jó magyarázatokat találunk az értékekről.

A bejegyzések legfontosabb attribútumai a `value` és `locked` paraméterek. Míg a `value` az alapértelmezett értéket tartalmazza, a `locked` változó azt definiálja, hogy a felhasználó képes-e megváltoztatni a beállítást. Amennyiben a `locked` változót igaz értékűre (`false`) állítjuk a felhasználónak az opciók menüben nem fog megjelenni a lehetősége, hogy ezt a beállítást módosítsa. Amennyiben egy értéket frissen zárunk el a felhasználók elől, gondoljunk a korábbi intelemre! Azaz ha az adatbázisban létezik a beállítást felülbíráló bejegyzés, akkor az ténylegesen is felül fogja bírálni az itt beállított értéket. Ilyenkor *SQL* adatbázis esetén például szükséges a bejegyzések kézzel való törlése a `horde_prefs` táblából!

Miután megismertük a *Preferences* rendszert a teljesség igénye nélkül kiemelnék pár beállítást, amit mindenképpen érdemes megtennünk. Legfontosabb az adatbázis kapcsolat paramétereinek beállítása (*Database*) (*3. ábra*), az alapbeállításokat tároló rendszer (*Preferences System*) (*4. ábra*) engedélyezése, a *DataTree* rendszer engedélyezése (*5. ábra*) valamint a felhasználói hitelesítés beállítása (*Authentication*) (*6. ábra*). Mivel a következő lépésben az *IMP* levelezőt fogjuk beállítani, állítsuk be, hogy a hitelesítést az *IMP* végezze.

Ha ezt nem tennénk meg, a *Horde* és az *IMP* külön-külön kérne jelszót a felhasználóktól, ami nagyon bosszantó tud lenni, mert a levelezés eléréséhez például kétszer kellene belépni. Az adminisztrátor mezőben pedig állítsuk be a belépési azonosítónkat, ugyanis ezzel tudatjuk a rendszerrel, hogy az adminisztrációs menühöz is szeretnénk hozzáférni.

Miután a beállításokkal elkészültünk el kell juttatni azokat valahogy a *horde/config/conf.php* állományba. Beállíthatjuk, hogy ilyen esetben automatikusan töltse fel a *Horde* a konfigurációs állományt, de nekem szimpatikusabb volt az alul megjelenő ablak tartalmát kézzel (*Másol – Beilleszt*) bemásolni az előbb említett helyre.

A *Horde* keretrendszer ezen a ponton már működőképes, de jelenleg még nem tud senkit sem beléptetni, mivel a hitelesítést végző *IMP* modult még nem telepítettük. Ha esetleg ezen a ponton kilépve kizárnánk magunkat, akkor az eredeti *Horde* forráscsomag *horde/config/conf.php.dist* állományát másoljuk vissza a *conf.php* helyére és kezdjük az elejéről a beállítást!

Az IMP telepítése

Miután elkészültünk a keretrendszer beállításával kezdhetjük a bővítő modulok feltöltését.

Az *IMP* modul (és általában egy új *Horde* modul) telepítése igen egyszerű: első lépésként töltsük le és bontsuk ki a *Horde* gyökérkönyvtárába a modul forrását:

```
cd /var/www
wget ftp://ftp.horde.org/pub/imp/
➥imp-h3-4.0.3.tar.gz
tar zxvf imp-h3-4.0.3.tar.gz
mv imp-h3-4.0.3 horde/imp
```

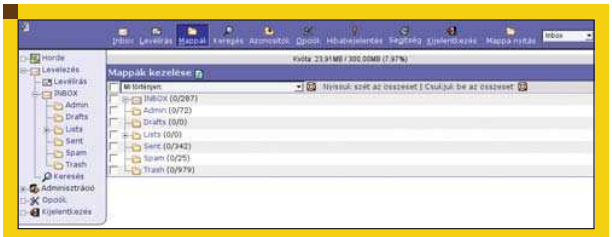
Ezután a *horde/imp/config* könyvtárban a *Horde* telepítésének harmadik lépésében ismertetett módszerrel a konfigurációs állományok végéről a `dist` kiterjesztést eltávolítjuk. Majd a *Horde* rendszer adminisztrációs felületén elkészítjük a konfigurációt, melyet a *horde/imp/config/conf.php* állományba másolunk.

Érdemes a modulok, jelen esetben az *IMP* estén végignézni a *horde/imp/config/prefs.php* állományt az *IMP* lehetőségeinek megismerésére.

2. lista A servers.php állomány

```
$servers['imap'] = array(
    'name' => 'IMAP Server',
    'server' => 'localhost',
    'hordeauth' => false,
    'protocol' => 'imap/notls',
    'port' => 143,
    'folders' => 'INBOX.',
    'namespace' => '',
    'maildomain' => 'maildomain.hu',
    'smtphost' => 'localhost',
    'smtpport' => 25,
    'realm' => '',
    'preferred' => '',
    'dotfiles' => false,
    'hierarchies' => array(),
    'quota' => array(
        'driver' => 'courier',
        'params' => array(
            'protocol' => 'imap/notls',
            'hostspec' => 'localhost',
            'port' => 143
        )
    )
);
```

■ *7. ábra* A belépésre szolgáló oldal



■ *8. ábra* Az üdvözlő oldal

Szemfüles olvasók most joggal kérdezhetik, hogy az *IMP* honnan fogja tudni, hogy hogyan is kell a *Horde* számára a felhasználók hitelesítését elvégezni. A választ a ***horde/imp/config/servers.php*** állományban kell keresni (*2. lista*). Ez adja meg ugyanis, hogy az *IMP* milyen levelező kiszolgálóhoz kapcsolódjon. A példa azt hiszem eléggé beszédes, egyedül az alsó néhány sorra hívnám fel a figyelmet, mely a *Courier-IMAP* kiszolgálótól a kvóta értékeket hivatott lekérdezni (a *Maildir++* használatához javaslom a `maildirmake` parancs `-q` kapcsolóját, illetve a súgó tanulmányozását). Mivel a többi kiszolgálóra nem lesz szükségünk, a többi részét törölhetjük az állománynak.
Ha kész vagyunk, akkor eljött az ideje, hogy megpróbálkozzunk egy valódi belépéssel. Írjuk tehát be a böngészőnkbe:

```
http://localhost/horde/
```

Ha sikerrel vettük az akadályokat akkor a belépési oldalon találjuk magunkat (*7. ábra*), ahol az IMAP kiszolgálón is működő azonosítónk és jelszavunk begépelése után az üdvözlő oldal fogad minket (*8. ábra*). Ekkor már működőképes a levelezőprogram is.

LDAP beállítások és kampók

Miután van egy működő alaprendszerünk eljött az ideje, hogy kicsit finomhangoljuk. Az *IMP* egyelőre kizárólag *IMAP* kapcsolaton keresztül kommunikál a kiszolgálóval, így nem képes lekérdezni, hogy a belépett felhasználónak mi a teljes neve és az e-mail címe. Mivel az általam készített rendszeren szükség volt arra, hogy a levélcímek `vezetéknév.keresztnév` alakúak legyenek ezért ezeket még az *LDAP* kiszolgálótól le kell kérdezni.

LDAP lekérdezések kampók (hooks) használatával

A keretrendszer az ehhez hasonló belső működésbe való beleszóláshoz speciális interfészt biztosít, melyet ***kampónak (hook)*** neveznek. Ezek olyan függvényhívási pontok, melyekhez saját rutinokat írhatunk, ha azt szeretnénk, hogy a rendszert azt meghívva a mi igényeink szerint végezzen el bizonyos feladatokat. Esetünkben két kampó használatára lesz szükség, egyre amelyik a felhasználónak teljes nevét, és egy másikra, mely a teljes e-mail címét kérdezi le az *LDAP* kiszolgálótól.
A kampók a ***horde/config/hooks.php*** állományban kaptak helyet. Itt nagyon sok példa kampót kapunk a fejlesztőktől, melyeket kis testre szabás után élesben is bevethetünk is.
Az *LDAP* séma, melyet a lekérdezésekhez használtam az `inetOrgPerson` `objectClass`. Ennek az osztálynak a `displayName` és `mail` attribútumai tartalmazzák a teljes nevet, illetve a teljes email címet (a `@` utáni rész nélkül).
A teljes név lekérdezéséhez a _prefs_hook_fullname() függvényre lesz szükségünk (a függvény nevéből látszik, hogy ez egy Preferences rendszert érintő kampó). Szedjük ki a függvény sorai elől a megjegyzésjeleket és az alábbi 3 sort állítsuk be az *LDAP* kiszolgálónknak megfelelően:

```
$ldapServer = 'localhost';
$searchBase = 'ou=Users,dc=maildomain,dc=hu';
$name = $information[0]['displayname'][0];
```

3. lista A backends.php állomány

```
horde/passwd/config/backends.php
$backends['ldap'] = array(
    'name' => 'MailDomain',
    'preferred' => '',
    'password policy' => array(
        'minLength' => 6,
        'maxLength' => 12
    ),
    'driver' => 'ldap',
    'params' => array(
        'host' => 'localhost',
        'port' => 389,
        'basedn' =>
'ou=Users,dc=maildomain,dc=hu',
        'uid' => 'uid',
        'realm' => '',
        'encryption' => 'crypt-md5',
        'tls' => false
    )
);
```

Ezután pedig már csak engedélyeznünk kell a kampó meghívását. Ehhez keressük ki a ***horde/conf/prefs.php*** állományban `prefs['fullname']` tömböt, ahova vegyük fel a

```
'hook' => true
```

bejegyzést.
Mint korábban említettem a ***Preferences*** rendszer csak belépéskor fut le, ezért, hogy kipróbálhassuk a kampó működését újra be kell lépnünk.
A teljes email cím lekérdezéséhez hasonlóan járjunk el, csak itt a `_prefs_hook_from_addr()`függvényre lesz szükségünk. Módosítsuk az alábbi 3 sort a következőképpen:

```
$ldapServer = 'localhost';
$searchBase = 'ou=Users,dc=maildomain,dc=hu';
$searchResult = @ldap_search($ds, $searchBase,
'uid=' . $user);
```

Ezután a ***horde/config/prefs.php*** állományban engedélyezzük a kampó meghívását a `$_prefs['from_addr']` tömbnek a `'hook' => true` bejegyzést felvéve. Próbáljuk ki a kampó működését egy ki- és belépéssel!

Címjegyzék használata

Miután már tudunk levelezni, itt az ideje az egyik legfontosabb bővítmény, a címjegyzék feltelepítésének. Ezt a ***Turba*** nevű modul végzi, melyet ugyanúgy kell telepítenünk, mint ahogy bármilyen plusz modult rendszerhez:

```
cd /var/www
wget ftp://ftp.horde.org/pub/turba/
➥turba-h3-2.0.3.tar.gz
tar zxvf turba-h3-2.0.3.tar.gz
mv turba-h3-2.0.3 horde/turba
```

A ***Turba*** képes többek között ***LDAP***, ***SQL*** adatbázisba dolgozni, vagy a ***Horde Preferences*** rendszerét is használhatja. Ez utóbbit az adatok fölösleges kavarodása miatt nem javaslom. Én az ***SQL*** adatbázist választottam, amihez szükséges a ***Turba*** táblájának létrehozása:

```
cd horde/turba/scripts/sql
mysql -u root -p horde < turba_objects.mysql.sql
```

Ezután a szokásos webes felületen való beállítás következik, ahol írjuk be a `localsql` nevet címjegyzék nevéhez, majd mentsük a ***horde/turba/config/conf.php*** állományt.
A ***horde/turba/config/sources.php*** állományban `localsql` a néven találjuk meg a beállításokat. Ezen nem nagyon kell módosítanunk semmit, alapértelmezésben a ***Horde SQL*** kiszolgálójának beállításait (gép, login, jelszó) veszi át.
A beállítások után már használható is a címjegyzék mely képes kezelni levelezőpartnerek dinamikus csoportjait is akár, mely automatikusan frissül akkor is, ha egy csoporttagnak megváltoztatjuk a címét.

Jelszóváltoztatás

Miután a levelezés jól működik szükségünk van még egy jelszóbeállító felület telepítésére is. Jelenleg ez a ***Passwd*** modul feladata, mely ugyan még béta állapotú, de problémát nem tapasztaltam vele. Telepítsük a szokásos módon:

```
cd /var/www
wget ftp://ftp.horde.org/pub/passwd/
➥passwd-h3-3.0-beta.tar.gz
tar zxvf passwd-h3-3.0-beta.tar.gz
mv passwd-h3-3.0-beta horde/passwd
```

Készítsük el a webes felületen a ***conf.php*** állományt!
Érdemes figyelni arra, hogy alapértelmezetten a ***Passwd*** engedélyezi bármelyik felhasználó jelszavát módosítani. Ezt érdemes kikapcsolni.
Ezután vegyük szemügyre a ***horde/passwd/config/backends.php*** állományt is (*3. lista*). Itt tudjuk ugyanis beállítani, hogy a ***Passwd*** modul miképp állítsa át a felhasználó jelszavát. Többek között lehetőségünk van ***SMB***, ***LDAP*** vagy ***SQL*** alapú jelszóváltoztatásra is. Számunkra az ***LDAP*** lesz fontos, hisz felhasználóink innen kerülnek hitelesítésre. A beállítást a *3. lista* mutatja, melyben azt hiszem csak a `crypt-md5` kódolás szorul rövid magyarázatra. A `crpyt-md5` olyan ***MD5*** kódolás, amit a ***Linux*** rendszerek a ***/etc/shadow*** kódolására alkalmaznak, ha ezt választjuk a `crypt` kódolás helyett. Azért választottam ezt a kódolást, mert ha később át kell venni más rendszerből felhasználót, akkor a jelszót a kódolt formában egyszerűen át tudjuk tölteni. Természetesen ez igaz a másik irányba is, azaz ha a rendszerünkről szeretnénk áttenni másik kiszolgálóra a felhasználót.
A ***Passwd*** modul alapértelmezetten inaktív állapotban van, a ***Horde*** számára engedélyezni kell. Ezt a horde/config/registry.php állományban az

```
applications['passwd']
```

tömbben a `'status'` attribútum `'active'` értékre való cserélésével tehetjük meg. Ezután a webes felületen a következő belépéskor elérhetőnek kell lennie a jelszóváltoztatás gombnak is.

Levélszűrési lehetőségek

Haladóbb felhasználók esetén nyújt segítséget az *INGO* modul, mely levél szűrési szabályokat képes kezelni. A szabályokat a felhasználók a webes felületen tudják összerakni maguknak. A szabályok segítségével tudják a bejövő leveleket mappákba rendezni azok fejlécének vizsgálata alapján.
Telepítése a szokásos modul telepítési módszerrel megy:

```
cd /var/www
wget ftp://ftp.horde.org/pub/ingo/
➥ingo-h3-1.0.1.tar.gz
tar zxvf ingo-h3-1.0.1.tar.gz
mv ingo-h3-1.0.1 horde/ingo
```

Ezután a webes felületen készítsük el a *conf.php* állományt. Én a *Horde Preferences* rendszert állítottam be a szabályok tárolására.
Érdekes megoldása az *Ingonak*, hogy képes *procmail* szűrőszabályt is készíteni a webes felületen összehozott szabályokból. Ehhez egy *FTP* kiszolgálóra van szükség, amin keresztül feltölti a felhasználó saját könyvtárába a *.procmailrc* állományt. Amennyiben nem ezt a módszert használjuk, akkor a szűrés kliens oldalon fog történni, ami valamivel lassabb, mintha a levél már beérkezésekor megtalálni a helyes irányt a megfelelő mappába. Sajnos *maildrop* formátumú *mailfilter* állományt jelenleg nem képes készíteni az *Ingo*.

A MIMP – Webmail WAP-on

Végezetül a *MIMP* modult érdemes még megemlíteni egy igényes web alapú levelező elkészítéséhez, mely mobil eszközök *WAP* böngészőjére teszi elérhetővé az *IMP* funkcionalitását. Tapasztalataim szerint régebbi mobiltelefonnál okozhat gondot, hogy a *MIMP* túl nagy oldalakat generál, amit a mobil nem képes fogadni. Ettől függetlenül mindképpen egy sokat ígérő komponens, melyet a fejlesztés *HEAD* ágában kell keresni ha telepíteni szeretnénk. A *Horde Wiki* oldalán *MIMPHowTo* néven találunk a beállításához részletesebb információkat.
További modulokat a *Horde* weboldalán találhatunk, a cikkben leírtak fényében egy új modul telepítése már könnyedén menni fog bárkinek.

A Horde magyarítása

A teljes *Horde* alkalmazásrendszer napról-napra fejlődik, így belefuthatunk abba, hogy a legújabb verziókban a magyar nyelv támogatása kivetnivalókat hagy maga után. Ilyenkor vesszük hasznát, hogy a *PHP* `gettext` függvénytárral lettek megvalósítva a nyelvi elemek. Amennyiben szeretnénk a szövegfüzéreket saját igények szerint átírni, a *horde/po/translation.php* állományra van szükségünk. Első alkalommal a *horde/po* könyvtárból futtassuk a következő parancsot:

```
./translation.php compendium -l hu_HU
```

Majd ezután ki kell gyűjtenünk a forrásból a szöveges elemeket, melyeket a *Passwd* modulra például így tehetünk meg:

```
./translation.php extract -m passwd
```

Majd készítsük el a *Passwd* modul *po* állományait:

```
./translation.php init -l hu_HU -m passwd
```

Ezután a *horde/passwd/po* könyvtárban a *hu_HU.po* állományban tudjuk szerkeszteni a szövegfüzéreket. Ha elkészültünk, a változtatásokat a követező paranccsal tudjuk érvényre juttatni a *horde/po* könyvtárból:

```
./translation.php make -m passwd
```

Mint láthattuk a *Horde* keretrendszer egy nagyon széleskörű és messzemenőkig testre szabható alkalmazáscsomagot biztosít számunkra. A rendszer mérete és objektumközpontú szemlélete miatt a *PHP* értelmezőre az oldallekérések folyamán elég erős terhelés jut, amit érdemes valamilyen *PHP* gyorstárazó programmal mérsékelnünk. Saját tapasztalatok szerint a *PHPAccelerator* nevű ingyenes program használatával elfogadható sebességet érhetünk el szerényebb teljesítményű kiszolgálók esetén is.
Aki úgy érzi, hogy szívesen csatasorba állítaná a rendszert, annak ajánlom a levelezőlisták forgalmának figyelemmel kísérését, ahol szívesen fogadják a jól definiált hibajelentéseket és készséggel segítenek azokon a problémákon túljutni, melyeket a dokumentációk alapos átolvasásával sem tudunk megoldani.

Pallos B. Richárd (pallos@mail.tmit.bme.hu)
Informatikus, doktorandusz hallgató a BME-n. Munkájához évek óta főként Linuxot használ. Kedveli a vízi és téli sportokat.

KAPCSOLÓDÓ CÍMEK

A Horde Projekt honlapja:
➲ http://www.horde.org/

A Horde Wiki oldala:
➲ http://wiki.horde.org/

A Horde levelezési listái:
➲ http://www.horde.org/mail/

A PHPAccelerator honlapja:
➲ http://www.php-accelerator.co.uk/