Linux alkalmazása

Horváth Árpád

2014. február 18.

1. Előtörténet

A UNIX-ot 1969-ban fejlesztették ki AT&T vállalatnál. 1972-ben ugyanitt fejlesztették ki a C programozási nyelvet, és a korábban assemblerben írt UNIX-ot újraírták C-ben. Kezdetben szabadon terjeszthető volt forráskóddal együtt, majd üzletet látva benne zárt kóddal pénzért árulták. Több változata lett System V (V=öt), BSD (Berkley Software Distribution, a Berkley egyetemről), HP-UNIX, SUN Solaris, Mac OS X. Ezek egy része utóbbi később szabad szoftverré vált: *BSD (pl. FreeBSD), OpenSolaris.

A Szabad Szoftver Alapítvány (FSF) által 1984-ben indított GNU projekt célja egy teljes Unix-kompatibilis szoftverrendszer kifejlesztése volt. Ebből származik a Linuxra épülő terjesztések fordítóinak, szövegszerkesztőinek, parancsértelmezőinek (shell-jeinek) jelentős része. És az olyan UNIX-os parancsok nyílt forráskódú változata, mint például az ls, cp parancsok. Az alapítvány alapítója és a mozgalom „élharcosa" Richard Stallman (rms).

1987-ben egy egyetemi tanár, Andy Tanenbaum, kifejlesztett oktatási célokra egy minimális operációs rendszert, a MINIX-et. Ennek teljes forráskódja megtalálható Tanenbaum: *Operációs rendszerek* című könyvében.

Ezt használta fel Linus Torvalds, svéd anyanyelvű finn egyetemi hallgató egy operációs rendszer kernelének a kifejlesztéséhez, melyet később Linuxnak neveztek el és azóta is Linus fejleszti. A munkát 1991-ben kezdte.

Precíz szóhasználatban a Linux a kernelt jelenti. Teljes operációs rendszerré a Linuxot a GNU-s programok teszik, ezt az operációs rendszert GNU/Linuxnak szokták hívni. A különböző Linux terjesztések ezeket a programokat szoftvercsomagokba szervezik és csomagkezelő programokat használnak a szoftvercsomagok telepítésének egyszerűbbé tételéhez.

A Linuxok (talán kizárólag) az X Window rendszert használják a grafikus felületek alapjaként, ez biztosítja az alapvető keretrendszert GUI környezetek építéséhez: ablakok kirajzolása és mozgatása a képernyőn, együttműködés az egérrel és billentyűzettel.

Erre épülnek rá az ablakozó rendszerek (pl. IceWM, (Open/Flux)Box) és a összetettebb grafikus felhasználói felületek (GNOME, KDE, Xfce). A kettő közötti pontos különbséget megnézhetné valaki, az előbbiekben nincs kölcsönhatás az ablakok között (drag-and-drop, vágólap).

A GNU/Linux rendszerek és a UNIX-ok között jelentős hasonlóság van, amit több szabvány, például a POSIX nagymértékű betartása tesz lehetővé. Aki egy UNIX-rendszert ismer, az nem fog elveszni a Linuxban és fordítva.

2. A szabad szoftver – nyílt forrású szoftver

A Szabad Szoftver Alapítvány a szabad szoftvert nem egyszerűen a szabad (jogtisztán ingyenes) felhasználást érti, tehát nem a shareware programokat, hanem a következő jogok meglétét:

- 0. szabadság: A program tetszőleges célra *szabadon felhasználható.*
- 1. szabadság: A program működése *szabadon tanulmányozható, és módosítható* a saját célunknak megfelelően.
- 2. szabadság: Szabadon *terjeszthető és továbbadható.*
- 3. szabadság: Lehetőség van a szoftver *továbbfejlesztésére* és a fejlesztés *közreadására.*

Az 1-3. szabadág feltételezi a *forráskód elérhetőségét.*

A szabad szoftver nem jelent feltétlenül ingyenességet. Ha például valaki hajlandó azért fizetni, hogy egy szabad szoftvert az ő igényeire alakítson valaki, akkor nyugodtan kérhet pénzt a fejlesztő. Mindenesetre nem lehet visszaélni annyira vele, mint egy zárt forráskódú program esetén, hiszen kidobhatom a fejlesztőt, ha túl sokat kér, és másikat vehetek fel.

A szoftverek nagy csoportja a *nyílt forrású szoftverek*, amelyek nem feltétlenül szabad szoftverek, mert elképzelhető, hogy a fenti feltételeket nem teljesítik. Tehát a nyílt forrású szoftverek részhalmaza a szabad szoftverek.

Bővebben itt: `http://www.fsf.hu/index.php/Whatis_fs`.

3. Linux-terjesztések (distribution)

Grafikus felületek: KDE, talán a leghasználhatóbb kezdőknek, sokan jobban szeretik a GNOME-ot, mert nincs annyi felesleges csicsa, az előbbi a Qt grafikus könyvtárra (C++), az utóbbi a Gtk grafikus könyvtárra (C) épül. Vannak kisebb erőforrásúak is mint például az IceWM, LXDE, az XFCE és az FVWM. Nem Windows-szerű, de állítólag nagyon használható: *box (Fluxbox, Openbox...), Enlightenment.

Linux-terjesztések (distributions):

- Debian: szervernek szeretik (meg a FreeBSD-t), deb csomagok
- Ubuntu: egyszerű telepítés és használat (sima: GNOME-os, Xubuntu: gyors Xfce-vel, kubuntu: KDE-vel, edubuntu: oktatási programok) deb csomagok
- SuSE, RedHat → Fedora, Mandriva (rpm csomag)
- Slackware, gentoo, arch: a csomagok a forráskódot vagy a forráskód letöltési helyét tartalmazzák a fordításhoz szükséges információkkal

Érdemes otthon telepíteni Linuxot. Az Ubuntut vagy kitartóbbaknak a Debiant ajánlom. http://ubuntu.hu http://debian.com Szívesen veszem, ha valaki másikakkal kísérletezik (Gentoo, Arch, FreeBSD, PC-BSD tapasztalatok érdekelnek).

Ha valakinek extra eszköze van, vagy soros egere, akkor Ubuntuból az alternatives változatot, amúgy a live változatot töltse le. (Szervernek természetesen a server változatot, az alapból grafikus felület nélküli.) A live változat telepítés nélkül is elindul CD-ről, így meg lehet nézni, szeretne-e ilyet az ember, milyen hardwert ismer fel. Lassú gépekre lubuntut telepítsünk, az jóval gyorsabb. Mivel a live CD-re nem fér rá minden nyelv csomagja ezért érdemes a magyar változatot letölteni, de a magyar nyelvi csomagokat felrakhatjuk hálózatról is.

Ugyan nem Linuxok, de a BSD változatai (FreeBSD, NetBSD, OpenBSD, PC-BSD,...) is szabad szoftverek. Amíg a Linuxot több összetevőből rakják össze (GNU projekt, Linux-kernel), addig a BSD-k teljes alaprendszere (kernel+operációs rendszer) együtt kerül fejlesztésre így jobban összecsiszolt. A programoknál lehetőség van ugyan a binárisból történő telepítésre, de a forráskódból történő telepítés is nagyon egyszerű, a gentoo-hoz hasonló. A hardvergyártók egyelőre még kevésbé veszik figyelembe a BSD-seket, ezért a harvertámogatottság kisebb mértékű, mint a Linux esetén. Végül három érdekesség: 1) a Mac OS X operációs rendszer, az Apple Macintoshának 2002-ben bevezetett operációs rendszere szintén BSD alapú; 2) hogy a Debian 2011 februárjában kiadott 6.0-ás verziójában választható olyan telepítő is, amely Linux kernel helyett a FreeBSD kernelt használja; 3) az Android a Linux leszármazottja.

4. Vim

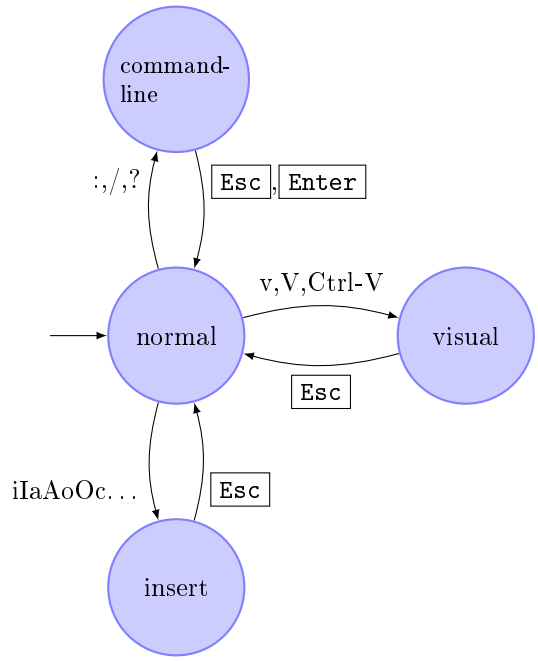
A Vim egy nagyon hatékony szövegszerkesztő (text editor), amely karakteres felületen is működik, de van grafikus felületű változata is (gvim). Ez utóbbi használható Windowson is, és van változata Mac OS X-re is.

A Vim képes több puffert is kezelni, és azokat összehasonlítani (vimdiff), azok között másolni. A Vim kezelni tudja az egeret is, de fő ereje a billentyűzet használatában van, és Unix/Linux rendszereken abban, hogy a pufferek tartalmát külső Unixos szűrőutasításokkal (lásd 5.2 alfejezetben) könnyedén lehet szűrni.

A Vi egyik utódja, melytől azonban megkülönbözteti egy halom hasznos lehetőség, például több szövegpuffer kezelése, szintaxiskiemelés, többszörös visszavonás (undo).

Több fejlesztőkörnyezet és shell, mint például az Eclipse és a bash, nem véletlenül tartalmaz olyan lehetőséget, hogy Vim-szerűen lehet velük szerkeszteni a szöveget.

A legfontosabb, amit a Vimnél meg kell érteni, a módok. A módok egy vázlatos állapotgáfja található alább.



Általában a Vim normál módban indul. Command-line (parancssori) módba kerülünk például, ha :-ot nyomunk, ekkor vim parancsokat írhatunk be, amelyet egy Enter után végrehajthatunk (vagy Escape-pel megszakíthatunk). A parancs megjelenik az alsó sorban. A vim módokat megkereshetjük például a help paranccsal `:help vim-modes`

Szintén a Parancssori módhoz tartozik a keresés is. Előrefelé a keresést normál módból /-el indíthatjuk. Ezután egy reguláris kifejezést írhatunk be és ezt Enter után megkeresi.

Ha be szeretnénk írni valamit, azt Insert módban tehetjük. Ebbe a módba leggyakrabban az `i` paranccsal jutunk. Ha beírtuk a szöveget Escape-el juthatunk vissza Normál módba.

A visual módban tudjuk kijelölni a szöveg egy részét (karaktereket, sorokat vagy téglalap alakú blokkot), hogy azt például töröljük, átírjuk, módosítsuk, szűrjük egy szűrőutasítással, cseréljünk benne szövegrészeket vagy köré írjunk valami környezetet.

4.1. A Vim beállítása, sorok kezelése, parancsismétlés, kilépés

Vegyük végig a vimtutorból az első 3 leckét. Lépjünk be a Linux alá, futtassuk a vimtutor parancsot.

Tanuljuk meg a blokkok kezelését a vimtutorból. Lépjünk be a Linux alá, futtassuk a vimtutor parancsot, keressünk rá a téglalap szóra (`/téglalap`) és hajtsuk végre az ott található feladatokat (9. lecke).

Hozzunk létre egy másolatot a `.bashrc` fájlról `tmp.sh` néven és gyakoroljunk rajta a vim-mel.

```
cp .bashrc tmp.sh
vim tmp.sh
```

Állítsuk be, hogy színezzen (syntax highlighting): `:syntax on`

Még jobb követni a `http://github.com/horvatha/vimrc` oldal alján levő leírást a tároló klónozásához, a .vimrc létrehozásához és a hasznos kiegészítők telepítéséhez.

Sorok törlése (delete): `dd`, parancs ismétlése `.` (pont), törölt sor (szó) beillesztése aktuális sor (karakter) után `p` (paste), elé `P`. Hogyan lehet leggyorsabban két sort felcserélni?

Szöveg beírása: `i` (insert) után lehet, utána `Esc` a normál módba visszatéréshez. Ekkor szerkeszthetjük, ahogy más szerkesztőkben szoktuk. (Az `i` vált át normál módból beszúrás (insert) módba, amit megfelelő beállítások esetén alul láthatunk (`--INSERT--`), az `Esc` tér vissza.)

Parancs n-szeri végrehajtásához előtte beírni a számot. Próbáljuk ki:

`3dd5↓2p`

`22i+-` `Esc`

A visszavonásokat az `u` -val tehetjük meg. Nyomjuk le többször az `u`-t. Mi történik?

Mentsük a fájlt `:w`, más néven is: `:w fajlnev`! Próbáljuk meg mégegyszer ugyanolyan néven! De csak azért is mentsük `:w! fajlnev`!

Fontos: Kilépési lehetőségek: mentéssel `:wq`, mentés nélkül `:q!`. Ha nem vagyunk biztosak, milyen módban vagyunk, nyomjunk előtte két `Esc`-et

A `d` helyett `y`-t (yank=ránt) használva nem törlünk, hanem csak másolunk. Próbáljuk ki több sor másolását több példányban!

Ez csak a kezdet. Apránként ismerkedünk majd meg a Vim finomságaival. Az eddigiek közül a szintaxiskiemelés nem működik sima vi-jal és az `i` parancs után kissé nehezebben lehet szerkeszteni a fájlt.

Egyedül tanuláshoz:

Írjuk be a `vimtutor` parancsot :-)

magyarul: http://people.inf.elte.hu/birki/ab/gyak/02.txt.html

angolul: `vimtutor` parancs végigvezet sok hasznos tudnivalón.

angolul: A vim helpjében sok hasznos dolgot találunk:

```
:help user-manual
```

angolul: fontosabb parancsok http://worldtimzone.com/res/vi.html
A Vim letölthető Windowsra is (gVim): vim.org

5. Parancsuralom a shell felett

Töltsük le a Linuxos anyagot a bazaar verziókezelő rendszerrel:

```
# apt-get install bzr    # Ha még nincs bazaar
$ cd ahova/le/akarom/tölteni
$ bzr branch http://mail.roik.bmf.hu/linux
```

A bazaar részletesebb használatát lásd a Bazaar szakaszban (9.2. szakasz).

Fordítsuk le és nyissuk meg a gyakorlatok leírását (F11 teljes képernyő):

```
cd segedlet
make
evince linalk.pdf&
```

Az & háttérbe küldi a folyamatot, és visszakapjuk a promptot, további utasításokat írhatunk be az előző folyamat megszakítása nélkül.

5.1. BASH: fájl- és könyvtárműveletek, jogok, beállításai

A `bash` a Linux/UNIX egyik héja (shellje, parencsértelmezője), amely parancsok végrehajtását teszi lehetővé. Van még sh, csh, zsh, ksh, dash...

Készítsünk részletes listát a fájlokról, könyvtárakról, a rejtetteket is beleértve: `ls -la`

A rejtett fájlok/könyvtárak ponttal kezdődnek, alapból nem jelennek meg listázáskor, általában beállításokat tartalmaznak.

A fájllista elején szerepelnek a jogosultságok pl. `drwxr-x--` első betűje könyvtárat jelöl, a következő három betű a felhasználó jogait (olvasás/listázás, írás és végrehajtás/belépés) a következő kétszer három a csoport ill. a többi felhasználó jogait.

A következő oszlop még nem érdekes, a hardlinkek számát mutatja. Utána áll a fájl/könyvtár tulajdonosa és csoportja, amire a (jogok vonatkoznak), majd a méret, módosítási dátum és a fájlnév.

Keressük meg az ls leírásában a `-l -A -a -t` és `-r` kapcsolókat:

```
man ls
/-l<Enter>
```

A man-ból q-val léphetünk ki.

Mit jelent az `ls -trl`?

Váltsunk könyvtárat `cd /`, listázzuk, térjünk vissza `cd` (call directory)

Ha a `manpages-hu` csomag telepítve van, akkor a leggyakoribb utasításokról magyarul kapunk leírást.

Nézzük meg a `mv` parancs leírását. Mit csinál, milyen formában használhatjuk? Mit jelent a forrás... (a három pont) az áttekintésben?

Hozzunk létre és tegyünk futtathatóvá a bin könyvtárban a `cx` parancsot, melynek tartalma:

```
chmod 755 $*
```

Állítsuk be, hogy a `bash` a gép minden indulásakor megtalálja a homekönyvtárunkban található `bin` könyvtárat!

Állítsuk be a hasznos `ll, la` parancsokat, hogy mindig rendelkezésre álljanak (alias)!

Indítsunk új terminált (vagy újból jelentkezzünk be) és próbáljuk ki az `ll` és `cx` parancsainkat! Állapítsuk meg a típusukat! `type ll cx`

Fontos parancsok: cp (copy), ls (list), mv (move), rmdir, mkdir, pwd, chmod, chown, alias, type

Próbáljuk kitalálni a még ismeretlenek jelentését! Hozzunk létre egy temp nevű könyvtárat, majd töröljük! Hozzuk létre megint (felfelé gomb=előző utasítás), másoljuk bele a tmp fájlunkat! Most töröljük! (`rm -r`, rekurzív)

Otthoni tanuláshoz magyarul:
`http://www.cab.u-szeged.hu/local/doc/UNIX/orlando/bev.html` Orlando Unix-iskola

Bevezetés a bashbe angolul:
`http://www.tldp.net/LDP/abs/html/index.html` Advanced bash-scripting guide

5.2. Csővonal (pipe) és átirányítás

A UNIX-os utasítások egy része szűrő jellegű, ami háromféleképp veheti a bemenetét.

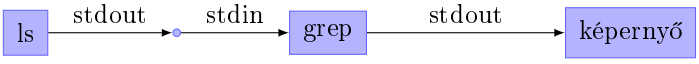
1. egy csővonalból
2. a billentyűzetről
3. egy fájlból

Igazából az első kettő a UNIX/Linux szempontjából ugyanaz. Ő ha nincs megadva fájl, az úgynevezett standard bemenetről (stdin) várja az adatot, ami vagy az úgynevezett csővonalból jön, vagy a billentyűzetről. A hasznos kimenetet minden szűrő a standard kimenetre (stdout) küldi, ez csővonallal továbbküldhető egy másik utasításnak, vagy megjelenik a képernyőn, esetleg beleirányíthatjuk egy fájlba. Ha egy utasítás végrehajtása során hiba történik, az nem megy a csővonalba, hanem az úgynevezett standard hibakimenetre (stderr).

Hogy értsük is, pár példát mutatunk: az egyik szűrőutasítás a `grep`. Ez egy szövegrészletet keres meg a bemenetében, pontosabban egy reguláris kifejezésre keres rá.

1a. példa. `ls /usr/bin | grep im$`

Itt az ls parancs kilistázza a felhasználók által végrehajtható utasításokat és kiírná a képernyőre (stdout-ra) a kimenetet, de a | (függőleges vonal) azt mondja, hogy a kimenetet adja át a grep utasításnak, ez lesz az ő stdin-je. A grep pedig kiszűri az im végű fájlneveket. A `$` a sor végét jelenti a reguláris kifejezésekben.



1b. példa. `ls /usr/bin | grep im$ >ezegyfajl`

Az előzőhöz hasonló, csak a kimenetet egy fájlba küldjük, nem a képernyőre.

2. példa. `grep im$`

Ilyenkor a billentyűzetről vár bemenetet, és minden soremeléskor, ha a sor illeszkedik a mintára, kiírja a sort.

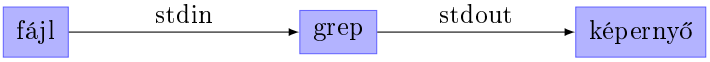
A futtatása így néz ki:

```
$ grep im$
sünim
sünim
image
Palim
Palim
FELEIM
```

A végén Ctrl-D vel jelezzük, hogy végeztünk (ezzel fájlvége jelet, EOF-et küldünk).

3. példa. `grep ^alias ~/.bashrc`

Ha egy fájlnév adott a szűrőutasításban, akkor a fájlból veszi az adatokat. Itt a alias-szal kezdődő sorokat. A `^` (kalap jel) a sor elejére illeszkedik.

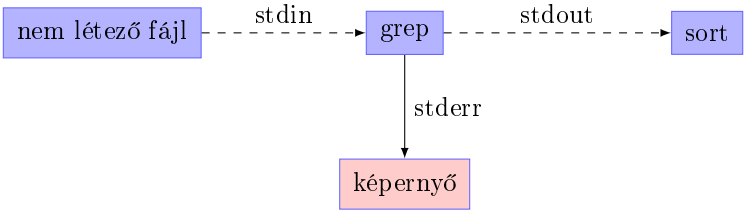


A futtatása így nézhet ki:

```
$ grep ^alias ~/.bashrc
alias lt='ls -ltr'
alias ll='ls -l'
alias la='ls -A'
alias cx="chmod a+x"
alias cr="chmod a+r"
alias ipythonlab='ipython -pylab'
```

4. példa. `grep ^alias nincsilyenfajl | sort`

Ha a grep-ben valamilyen hiba van, például nem létezik a megadott nevű fájl, akkor a kimenete (a hiba jelzése) nem megy tovább a csővonalon, hanem a stderr-re megy, és mindenképpen kiíródik. (Na jó, van arra is mód hogy átirányítsam a hibakimenetet is egy fájlba.) Így a sort szűrő, ami rendezni szokott nem nyeli el a hibautasítást (ekkor azt hinném, minden rendben volt), és nem hibautasítást fog rendezni.



További szűrőutasítások. Milyen héjaink (shelljeink) vannak?

```
ls /bin |grep sh
```

A `cat fn` utasítással kiírhatunk egy `fn` fájlt, vagy többet. Keressük meg ezzel a diakok sorait a `/etc/passwd` fájlban! Használjuk a grep-et, hogy csak a diak tartalmú sorokat írja ki.

Számoljuk meg, hány ilyen sor van! wc (word count, sor-szó-betűszám) -l kapcsolóval csak sorok száma.

```
cat /etc/passwd |grep diak|wc
```

Mivel a grep fájlból is olvashat ezt így is végrehajthatnánk:

```
grep diak /etc/passwd |wc
```

Ha viszont több fájlból kell szűrnünk, akkor a cat nélkül nem megy:

```
cat /etc/passwd /etc/group |grep diak
```

Vajon hogyan számolhatjuk meg a sorok számát a passwd fájlban? És `/home` fájljainak számát (azaz általában a felhasználók számát)?

Szűrők és fontosabb kapcsolóik (minuszt (-) tegyünk a kapcsolók elé):

neve	**leírás**	**kapcsoló**
grep	szűri a sorokat egy minta szerint	v
sort	rendezi a sorokat (alapból abc-rendbe)	n u
wc	szószámláló (word count)	l c w
head	csak a fájl elejét írja ki (=fej)	n
tail	csak a fájl végét írja ki (=farok)	n
sed	pl. szócseréket végez a szövegben (stream editor)	
awk	összetett feladatok végzésére (a szerzők nevéből)	
tr	karaktercsere	

Az alábbi példákat a homekönyvtárban próbáljuk ki:

```
grep alias .*
cat /etc/passwd | sort
ls /bin | wc -l
head .bashrc
tail -5 .bashrc
ls -l ~ | sed "s/diak/****/g"
cat .bashrc | tr eat EAT
cat /etc/fstab | awk '/dev/ {print NR, $1, $3}'
```

Keressük meg a man-ban a fenti utasítások táblázatban szereplő kapcsolóit. (pl. -l keresése: /-l)

A sed csereutasítása a Vimben is hasznos. A head és tail nagy fájloknál is gyorsan működik. Az awk egy önálló programnyelvvel rendelkező szűrő. Jó leírása van a Kernighan–Pike könyvben, de a man-ban is vannak hasznos példák.

A legvégső kimenet általában a képernyőre kerül. Bárminek a kimenete cső vagy a képernyő helyett fájlba is helyezhető > segítségével. Pl:

```
cat .bashrc .bash_profile > egyesitett
(df; uname -a) >gep_adatai
```

5.3. ClusterSSH

Ha egyszerre több gépen – például egy cluster gépein – kell ugyanazokat a parancsokat végrehajtanom, akkor hasznos program a ClusterSSH. Futtatásához, grafikus felület kell. A `cssh` parancsnak paraméterként megadva a szerverek IP-címének listáját mindegyik szerver terminálja külön ablakban jelenik meg, és egy kis beviteli mező is megjelenik, amelyben egyszerre az összes szervernek adhatunk parancsot. Az egyes terminálokba belépve külön is foglalkozhatunk egyetlen géppel is.

Telepítése Debian és Ubuntu rendszereken:

```
# apt-get install clustterssh
```

6. `.deb` csomagok telepítése és kezelése

A telepítési parancsokat root-ként adhatjuk ki, vagy megfelelő beállítások esetén `sudo`-t írva eléjük (Ubuntuban ez az alap, `man sudo`).

A Linuxokon a programok, könyvtárak csomagokba vannak telepítve. Egy könyvtárat több program is használhat. A `dpkg` egyedi csomagok telepítésére (`-i`), telepített csomagok listázására (`-l`), csomagok újrakonfigurálására (`dpkg-reconfigure`). Gyakran hasznos a `dpkg -l|grep kulcsszó` alak csomagok keresésére.

Az apt a dpkg-re épül, függőségeket, csomagok letöltését kezeli. Azt, hogy honnan tölthet le, a `/etc/apt/sources.list` szöveges fájl tartalmazza. Az alábbi első sor frissíti a csomaglistát, a második letöltögeti az `mc`, `vim-full` és `gimp` csomagokat függőségekkel együtt és telepíti.

```
apt-get update
apt-get install mc gimp vim-full
```

Bővebben: `http://321.hu/Debian/apt-guide.html`

Az alábbi sorokkal kereshetünk kulcsszónak megfelelő csomagokat, illetve megnézhetjük a csomag leírását:

```
apt-cache search postgresql  |grep python
apt-cache show vim
```

Az utóbbi kimenetének részlete:

Package: vim
Priority: optional
Provides: editor
Depends: libc6 (>= 2.7-1), libgpmg1 (>= 1.19.6-1), libncurses5 (>= 5.6+20071006-3), python2.5 (>= 2.5), vim-common (= 1:7.1-138+1ubuntu3), vim-runtime (= 1:7.1-138+1ubuntu3)
Filename: pool/main/v/vim/vim_7.1-138+1ubuntu3_i386.deb
Description: Vi IMproved - enhanced vi editor
Origin: Ubuntu

Ebből láthatjuk, hogy milyen más csomagok melyik verziójától függ a `vim` csomag. A csomagok prioritása lehet required, important, standard, optional and extra; ebben az esetben optional. A csomagok pl. az Ubuntu alatt három fő tárolóban vannak, ezek nevei: main, universe és multiverse. A Filename kezdetű sorból láthatjuk, hogy a `vim` a main tárolóban található és az Intel 368-as (i386) architektúrára fordított csomagunk van.

Igazából a fenti kimenetet a `apt-cache show vim` sorral csináltuk. Az apt-get és apt-cache parancsok helyett gyakran írhatunk aptitude-ot is (az alábbi összefoglalóan az upgrade-eket kivéve ugyanúgy), hasonló információkat tudhatunk meg, kicsit más kinézettel. Az aptitude talán többmindent magyarít, ha a magyar nyelv van beállítva.

Az apt grafikus felülete a Synaptic csomagkezelő (Rendszer > Adminisztráció > Synaptic).

Hasznos parancsok röviden összefoglalva:

```
dpkg -i <deb-fájl>        # Egy deb-csomagot telepít.
dpkg -l                   # (esetleg greppel szűrve) Telepített csomagok listája
dpkg-reconfigure <csomag> # (xserver-xorg, tzdata) Csomagok újból-beállításai
```

```
apt-get update             # A tárolóra vonatkozó információk frissítése.
apt-get  install <csomag>  # Csomag telepítése (függőségeket feloldja)


apt-get upgrade       # A tárolóban frissült csomagok telepítése
aptitude safe-upgrade # Az előző megfelelője.

vim /etc/apt/sources-list  # A tárolók helyének megadása.

apt-get dist-upgrade  # Frissítés új verzióra.
aptitude full-upgrade # Az előző megfelelője.

apt-cache search <keresett_szó>  # Keresés a csomagnévben
                                 # illetve a rövid és a hosszú leírásban
apt-cache show <csomag>          # Adott csomag fontosabb adatai
```

7. Telepítés

```
==================================================
  Debian GNU/Linux telepítése
==================================================

Kísérletezés céljából érdemes a Linuxot virtuális gépre telepíteni.
Erre a Virtualbox vagy a VMWare megfelelő. Ezek beszerzéséről és
beállításáról a fájl végén van szó.

Telepítés
===============

A ‘debian.org <http://debian.org>‘_-on keressük meg, hol lehet letölteni
képfájlokat, és hol találunk leírásokat. Ez otthoni házi.

Gyakorláshoz a helyi szerverről is leszedhetjük::

 wget http://arek.uni-obuda.hu/repo/linux/extra/debian-6.0.2.1-i386-netinst.iso

Virtuális gépben:
CD-ROM-nál megfelelő meghatót ki kellett jelölni vagy szedjük le
linux/telepitesbol a iso filet, és azt állítsuk be.
Indítsuk el a virtuális gépet. Ha van telepítő-CD bent legyen.

Miután megjelent az Debian felirat+csiga lépjünk be az ablakba (katt
oda) hogy tudjunk dolgozni ott.

(Virtualboxban Jobb Ctrl, WMWare alatt Ctrl-Alt a kilépés,
bal-alul mindkettőben ki van írva.)

Válasszunk szakértő módot. (Advanced options/Expert install)
```

Ha CD-ről telepítünk,
ellenőrizzük a CD integritását. Lehet, hogy a telepítés
vége felé derül ki, hogy nem ép és akkor kezdhetjük előről.

Legyen hu_HU.utf-8 az alapértelmezett nyelvi beállítás.
Adjunk meg pár más helyi beállítást is: pl. en_GB.utf-8 de_DE.utf-8 a
későbbi gyakorláshoz.

A telepítendő összetevőket átnézhetjük, most nem kell semmi.

A hálózat beállítását DHCP-vel végezzük.

gépnévnek mindeni debian<gépsorszám>-ot adjon
tartomány arek.uni-obuda.hu

Particionálás

Kézzel particionáljunk, az alábbi partíciókat hozzuk létre:

1. Fat32 terület (5% méret /windowsra csatolva)
   (képzeletbeli Windows, vagy egy olyan terület, amit
   közösen használ a Win és Linux adatoknak).
   (Ha akarjuk kihagyhatjuk, de jegyezzük meg, hogy ilyet is lehet.)

2. ext3 4 GB, hely /   Boot "zászlóval"

3. cserehely (swap) 256 MB (amekkora a RAM)

4. ext3 maradék /home

   jellemző_használat beállítása úgy: hogy várhatóan nagy fájlokat - pl.
   nagyfelbontású képeket és filmeket - tárolnak a felhasználók (csak példa)

Rendszeróra nem UTC szerint jár, ha van Windows, ha csak Linux akkor
igen.

Felhasználók:

- root (jelszó:most legyen root, de éles rendszeren számok-kis-nagybetűk legyenek benne)

- diak (diak jelszóval; amivel majd beléphetnek a gépünkre másik gépről)

Szoftverválasztás: csak szabvány rendszer

GRUB-ot a master boot record-ba

Feladatok a kész Linuxon
==========

Töltsük le a feladatokat a frissen telepített gépre::

```
 apt-get install git vim

 git clone git://github.com/horvatha/linux.git

 cd linux/telepites

 vim debian-telepites.txt
```

ha ez nem megy, akkor::

```
 wget http://arek.uni-obuda.hu/repo/linux/telepites/debian-telepites.txt
```

Telepítés befejezése után indítsuk újra. Ilyenkor megtalálnánk minden
Windowsos indítható partíciót és korábbi linuxainkat a GRUB menüjében.
De most nincs ilyen.

Rendszergazdává válás és visszatérés

Lépjünk be root-ként (SuperUserként)::

```
   su
   <jelszó megadása>
```

Ekkor # alakú promptot kapunk a $ helyett, a továbbiakban ahol #-tel kezdek
sort rendszergazdaként kell futtatni.

Ha vissza akarunk térni, akkor az alábbi billentyűkombinációt üssük le
(Most még ne.)::

```
  <Ctrl+d>
```

Használhatunk külön ablakot is a normál és a su felhasználónak.
Lásd lejjebb.

Programtelepítés

Telepítsünk egy postgresql szervert:

1. Parancssorban:

  a. Keressük meg, milyen csomag::

```
apt-cache search postgresql
```

   Így sok, szűrjünk::

```
apt-cache search postgresql |grep postgresql
```

nézzük meg egy kiválasztott csomag adatait::

```
apt-cache show postgresql
```

b. Telepítsük::

```
apt-get install postgresql
```

Mielőtt igent mondunk, nézzük meg hány csomag, mennyit kell leszedni, mekkora területet használ.

2. Aptitude felületén (ezt nem fogom kérdezni)

aptitude indítása:

```
aptitude
```

Keressünk rá a csomagra::

```
/postgre
```

Ez így túl sok, reguláris kifejezéssel keressünk::

```
/^postgre<Enter>
```

Következő előfordulás::

```
n
```

3. Grafikus felület alatt synaptic is használható.

Nézzük meg mekkora szabad hely van a rendszeren, hány % foglalt (jegyezzük meg)::

```
df
df -h
```

Nézzük meg milyen csomagok töltődtek le eddig (telepítés során és után)::

```
ls /var/cache/apt/archives
```

Ezek valószínűleg nem kellenek már, ha csak újra nem kell telepíteni ezeket, töröljük::

```
# apt-get clean (a # jelzi, hogy ezt rootként kell)
```

Most mekkora szabad hely van a rendszeren::

```
  df

Vim beállítása
----------------
Másik ablakon jelentkezzünk be diákként::

  <Alt><F2> ...

Nézzük meg a .bashrc-t::

  vim .bashrc

Nem színez::
  :syntax on

Alapból a Debian és Ubuntu csak egy minimális csomagot telepít a Vimből
(vim-tiny) ami nem tud szintaxiskiemelést, de szerencsére már
telepítettük a vim csomagot nemrég.

Állítsuk be, hogy alapból színezzen. De hol vannak a Vim rendszerszintű
beállításai? ...

Keressünk gyorsan fájlokat locate-tel.

a) Ehhez először telepítsük a locate csomagot::

     # apt-get install locate

b) Ehhez frissítsük a locate adatbázisát (rootként)::

     # updatedb

c) Mostmár kereshetünk::

     locate vimrc


Állítsuk be, hogy alapból színezzen a Vim!

a) Váltsunk rendszergazdára::

      su
      <jelszó megadása>

b) szerkesszük a beállítási fájlt::

     # vim /etc/vim/vimrc
     :syntax on  # Így jobban átlátható
     /syntax on    # rákeres a megfelelő sorra
```

```
     <syntax on sor elől " kivétele>
     <a set mouse=a sor elől is>
     :wq      # Kilépés mentéssel

c) Lépjünk vissza normál felhasználóra::

     <Ctrl+d>

Próbáljuk ki újra diákként::

     vim .bashrc

Otthon kipróbálhatjuk (vagy ha sikerül megtalálni,
hogy lehet karakteres felületen bekapcsolni az
egeret vmware alatt)::

   :sp .bash_profile
   (a két ablak határát egérrel húzkodhatjuk)

ssh és futási szintek
-------------------------

Nézzük meg melyik szintek milyen be-(ki-)lépéshez tartoznak és melyik
alapértelmezett::

  vim /etc/inittab
  (úgy tudom újabb Ubuntukon az inittab helyét más vette át)

Mik indulnak el az alapértelmezett szinten::

  ls /etc/rc2.d

Mentsük fájlba későbbre (gondoljuk át, hová)!

::
  ls /etc/rc2.d > lsrc2

Nézzük meg a gépünk hálózati adatait::

  # ifconfig    (# a root promtot jelöli, nem kell beírni)

Telepítsünk ssh szervert (keresés, telepítés)!

Ismételten listázzuk az rc2.d könyvtárat, mi változott?

::

  ls /etc/rc2.d > lsrc2_
  vimdiff lsrc*
```

```
Milyen típusú fájlok ezek?

::
  ls -l /etc/rc2.d

Milyen program indul el az indításakor? Nézzünk bele az S20ssh
tartalmába!

::
  vim /etc/rc2.d/S20ssh

..
  Ez valószínű, hogy nem fog menni:
  | Próbáljunk bejelentkezni egy másik friss Debianra:
  |
  |   ssh diak@192.168.3.2xx  (pl: xx=08)
  |
  | Ki (és mit) dolgozik ott?
  |
  |   w

A legvégén állítsuk le a gépet (root)::

  # halt

Újraindítás::

  # reboot

Virtualbox beszerzése és beállítása
===================================

Telepítsük Debian/Ubuntu alá a Virtualboxot:
virtualbox csomag és virtualbox-ose-modules megfelelő változata.

Rakjuk bele a felhasználót a vboxusers csoportba rootként vagy sudoval::

  # adduser diak vboxusers

diak felhasználóként ellenőrizhetük, hogy benne van::

  $ groups

Rootként betöltjük a kernelbe a vboxdrv modult::

  # modprobe vboxdrv

Ellenőrizzük, hogy megvan::

  # lsmod |grep vbox
```

Ez elveszik újabb indításkor, ezért rakjuk be a
/etc/modules fájlba a vboxdrv sort.

Linux 2.6 kernel kijelölése

Memóriaméret: 256MB elég karakteres szerverhez, később állítható

6GB merevlemez-méret legyen a változó méretű jó nekünk.

Ellenőrizhetjük, hogy mekkora helyünk van a gépünkön::

```
$ df
```

VMWare beszerzése és beállítása
==========

VMware Workstation-t be tudunk szerezni, azzal is telepíthetünk. A kész
telepítést a szabadon letölthető VMware Player is elfuttaja.

Indítsuk a VMWare-t!

Hozzunk létre új virtuális gépet!

Configuration: Custom
V. machine format: New - Workstation 5
Networking: Bridged

Linux/Other Linux 2.6 kernel kijelölése

Memóriaméret: 256MB elég karakteres szerverhez

6GB merevlemez-méret legyen
és ugyanabban az ablakban foglaljuk le a diszkterületet előre
(Allocate all disk space)

8. Munka több gép között

Az alábbiakban a távoli gépekkel történő kapcsolatokat vizsgáljuk. Ha távoli gépen akarunk ugyanúgy dolgozni, mint a sajátunk karakteres felületén, akkor azt általában ssh-val tesszük, ami titkosított kapcsolatot létesít. Fájlok másolására az ftp, az sftp vagy az scp szolgál. Leírás ezekről (Windowsos lehetőségek is):
http://progkor.inf.elte.hu/pandora.htm

8.1. ssh

Jelentkezzünk be a távoli gépen.
`ssh -p x diak`*n*`@mail.roik.bmf.hu`
ahol $n = 1, 2, \ldots, 15$, a port x értékét nem írom ide.

Mostmár az alábbi kódolási eltéres ritkább, a szerverek és asztali (desktop) gépek Linuxai is általában utf-8-as kódolást használnak, de hátha mégis hasznos lesz valakinek.

Ha utf-8-as kódolású gépről jelentkezünk be latin2-esre (iso-8859-2), akkor az ékezetes betűk nem mennek át. Ekkor hasznos az alábbi alak.

```
luit -encoding iso-8859-2 -- ssh -p x diak1@mail.roik.bmf.hu
```

Windows alól a putty.exe programmal léphetünk be. Ott is beállítható a karakterkódolás a `Translation` részben.

A fejezet többi része nem tartozik szorosan az ssh-hoz. Váltsunk a gyökérkönyvtárra `cd /`, listázzuk, térjünk vissza `cd` (call directory) A diak15 felhasználó (home)könyvtárának rövidítése: `~diak15`, a sajátunké `~`, a pillanatnyié `.`, a pillanatnyi szülőkönyvtáráé `..` Váltsunk diak15 könyvtárába, majd vissza!

Mennyi szabad hely van a lemezen: `df` (disc free). Mennyit használ a pillanatnyi könyvtárunk: `du -h` (disk usage).

Nézzük meg ki mit csinál a gépen: `w` (who). Ha elfelejtem hol vagyok: `pwd` (print working directory, egyébként azt többnyire látjuk a prompt ($) előtt). Milyen felhasználói könyvtárak vannak: `ls /home` vagy (ha a sajátomban vagyok) `ls ..`

Indítsuk el a Vim oktatóját: `vimtutor hu` Ennek a végigtanulmányozása házi feladat. Kilépés q! Ha valaki otthon akarja kipróbálni, és nem akar bejelentkezni a mail-re, a http://mail.roik.bmf.hu/linux/vim oldalról letöltheti a tutor.hu fájlt, és `vim tutor.hu` utasítással megnyithatja. (vim.org-ról letölthető a windowsos vim telepítője, azzal is megfelel.)

8.2. (s)ftp, scp

A sima ftp-t használhatjuk, ha nem titkosított fájlokat kell letöltenünk, pl. valamelyik Linux-disztribúció iso-ját. Ellenkező esetben sftp-t használhatunk ($\in$ lftp), nyilván a titkosítás és visszafejtés időt és erőforrást igényel.

Az ftp leggyakoribb parancsaival ismerkedjünk meg. Jelentkezzünk be:

```
ftp mail.roik.bmf.hu
```

Ez majd rákérdez a felhasználóra és a jelszóra. cd-vel válthatunk távoli könyvtárat ls-sel listázhatunk, lcd-vel helyi könyvtárat válthatunk. (A `!cd` parancs jogosan nem vált könyvtárat, hiszen minden parancs egy saját héjat (shellet)

nyit meg, és abban vált könyvtárat, majd visszalép az eredeti héjba, ahol ez nem érvényes.) A ! elejű sorok helyi parancsként értelmeződnek. help segítséget ad.

Az (m)get és (m)put paranccsal lehet fájlokat lehívni és feltölteni. Az m esetén több (multiple) fájlt, pl. `mget *.jpg` Nagyon sok fájl esetén érdemes a `prompt` utasítást kiadni, akkor nem kérdez rá minden egyes fájlra letöltés előtt.

Töltsük le a mail szerverről a diak1 felhasználó kepek/kepek könyvtárában levő képeket!

sftp szintaktikája:

```
sftp diak1@mail.roik.bmf.hu
```

Amúgy a Firefox-szal is ftp-zhetünk. Beírhatjuk URL-nek:

```
ftp://mail.roik.bmf.hu
```

Midnight Commanderben (mc) a menüből kiválasztható a Jobb/FTP-kapcsolat menüpont. A Total Commander is képes ftp-re.

Az scp utasítást akkor használhatjuk, ha pontosan tudjuk a fájl nevét és helyét. Használata a cp utasításához hasonló. Ilyenkor csak a jelszóra kérdez rá, nem ad promptot. Ez szkriptben hasznos. Pl:

```
scp diak1@mail.roik.bmf.hu:.bashrc temp
```

9. Elosztott verziókezelő rendszerek

A központosított verziókezelő rendszerek (SVN, CVS) esetén minden fejlesztésben résztvevő személy ugyanarra a szerverre küldi fel a változásait, és a gépén csak a fájlokról van másolata.

Az elosztott verziókezelő rendszerek (git, bazaar, mercury) esetén a fejlesztőknek a saját gépükön is megtalálhatóak azok az információk, amelyek a tárolón (repository) vannak, így akár internetkapcsolat nélkül is tud feltölteni (commit) a saját gépén lévő tárolójába változásokat új változatot (revision) hozva létre, amit aztán – megfelelő jogosultságok esetén – felküldheti (push) egy interneten elérhető szerverre. Leszedheti (pull) és összefésülheti (merge) mások változataival.

Mindegyik fejlesztésben alapvető egység az ág (branch), melyben fejlesztési modelltől függően a különböző verziókat tárolhatjuk, vagy esetleg egy új megvalósítandó is új ágat nyithatunk: http://nvie.com/posts/a-successful-git-branching-model/ .

A bazaar alapegysége az ág, a git-é az akár több ágat is tartalmazó tároló. A bazaar-ban egy új ág létrehozásához új könyvtárat kell létrehozni, míg a git-ben az ágak váltásakor ugyanabban a könyvtárban változik meg a fájlok tartalma.

A bazaar esetén tehát ágat (branch) másolok le, git esetén a tároló klónját (clone) hozom létre.

9.1. git

A git elosztott verziókezelő rendszert Linus Torvalds fejlesztette ki a Linux kernel fejlesztéséhez. Jól illeszkedik a https://github.com oldalhoz.

Egy tárolóban több ág található, amelyek között gyorsan válthatunk.

Letölthető illetve frissíthető vele például a linuxos tananyag legfrisebb változata:

```
# aptitude install git gitk
$ cd ahova/le/akarom/tölteni
$ git clone https://github.com/horvatha/linux.git
$ cd linux
```

Ha közben változik a githubon a tároló, ugyanebben a könyvtárban a klónunkat frissíthetünk így:

```
$ git pull
```

A # a root-jogosultságot jelöli, helyette sudo is lehet.
Grafikus felületen áttekinthetőbb képet kapunk a változásokról így:

```
$ gitk
```

9.2. Bazaar

A bazaar elosztott verziókövető rendszert, az a Canonical feljeszti, aki az Ubuntut is fejleszti. Az Ubuntu http://launchpad.net oldalával jól használható.
Letölthető vele például a linuxos tananyag legfrisebb változata:

```
# aptitude install bzr
$ cd ahova/le/akarom/tölteni
$ bzr branch http://arek.uni-obuda.hu/repo/cxnet
```

Frissítéskor csak ennyi kell

```
$ bzr pull
```

Ha változtattunk, és a változtatásainkat is meg szeretnénk őrizni, akkor a következőket tegyük.

```
$ bzr commit  # Ha változtattunk valamit
$ bzr merge http://arek.uni-obuda.hu/repo/cxnet
```

Grafikus felületen áttakinthetőbb képet kapunk a változásokról így:

```
$ bzr visual
```

Feltölthetünk saját verziókövetett rendszerünket egy tetszőleges helyre, amit pl. sftp-vel elérünk:

```
bzr push sftp://diak1@mail.roik.bmf.hu:4522/home/diak1/public_html/linux
```

A 4522 a port száma. A szerver azóta nem létezik, de a példának jó. A linux könyvtár, ha nem létezik, létrejön.

10. PostgreSQL

Az alábbi leírás szövegfájlban: linux/postgresql/README.txt

```
=======================
A PostgreSQL kezelése
=======================

Bevezető
====================

Az alábbiakban azt feltételezzük, hogy létrehoztunk egy felhasználót
diak felhasználónévvel pl. a

   # adduser diak
   (vagy sudo adduser diak)

paranccsal.
Ha diak helyett más felhasználónevünk van, akkor a diak-ot mindenhol
helyettesítsük azzal.

Feladatok
====================

0. Telepítés beállítás:
---------------------------------------


Telepítés (root-ként vagy sudo-val)::

   # aptitude install postgresql
   vagy
   sudo aptitude install postgresql

diak adatbázisfelhasználó létrehozása::

   # su postgres
   vagy
   sudo su postgres

   $ createuser diak

(y után n, ha jogot akarunk adni a felhasználónak
adatbázisok létrehozására)


1. Adatbázis létrehozása:
---------------------------------------

(postgres-ként, vagy ha a felhasználónak adtunk
```

adatbázis létrehozására jogokat, felhasználóként)::

```
 $ createdb diak
```

Alapból mindenkinek egy saját nevével azonos
nevű adatbázist érdemes létrehozni.

2. SQL-fájl tanulmányozása és beolvasása:

Normál felhasználóként::

```
 $ vim bank.sql
 $ psql -f bank.sql
```

Milyen függvényeket és táblázatokat hoztunk létre, és azok mit
csinálnak?

3. Interaktív munka:

Indítsuk el a PostgreSQL parancssorát! ::

```
  $ psql
```

Használhatjuk a bash-hez hasonlóan felfele gombot és a <Tab>-ot.
Minden SQL-parancsot pontosvesszővel kell lezárni. Ha elfelejtettük
a következő sorba csak a pontosvesszűt írjuk, ne az egész parancssort.
A psql saját parancsai \-vel kezdődnek, utánuk nem kell pontosvessző.

Listázzuk a táblákat! ::

```
 \d
```

Irassuk ki a táblázatok és a "betetesek" nézetmód mezőit! ::

```
 \d <táblanév>
```

Listázzuk a függvényeinket! ::

```
 \df public.*
```

Hozzunk létre új betétest!

(Találjuk ki a bank.sql és az előző feladatban szereplő függvények
segítségével, hogyan kell.)

Próbáljunk létrehozni nem létező névvel befizetést!

Próbáljunk törölni olyan sorokat, amelynek valamelyik mezőjére
REFERENCES hivatkozik.

Hozzunk létre új táblázatot! ::

```
 \h CREATE TABLE
```

Például iranyitoszamok néven (iranyitoszam, település) mezőkkel.
Segíthet az ebben a fájlban található puska (keressünk rá a Cheat Sheet
szövegre).

Hozzunk létre sorokat benne!

Kihagyható:
Keressünk benne reguláris kifejezésekkel (Cheat Sheet)

Lépjünk ki a psql-ből!
(Vajon melyik billentyűkombinációval lehet?)

4. Adatbázis mentése és visszaállítása

Adatbázis mentése::

```
 $ pg_dump diak >mentes.sql
```

Érdemes belenézni a mentett fájlba, tanulságos.

Adatbázis törlése
(postgres-ként, vagy ha a felhasználónak adtunk
adatbázis létrehozására jogokat, felhasználóként) ::

```
 $ dropdb diak
```

Adatbázis létrehozása
(postgres-ként, vagy ha a felhasználónak adtunk
adatbázis létrehozására jogokat, felhasználóként)::

```
 $ createdb diak
```

Adatbázis visszatöltése::

```
 $ psql -f mentes.sql
```

Ellenőrizzük, hogy tényleg megvan-e minden.

5. Törlések

```
---------------------------------------

Próbáljuk törölni a szemelyes_adatok táblázatot! ::

    DROP TABLE szemelyes_adatok;

Milyen sorrendben törölhetem a táblázatokat? Tegyük meg!

Listázzuk a függvényeinket::

    \df public.*

Töröljünk közülük::

    DROP FUNCTION ...;


PostgreSQL Cheat Sheet
==========================

from http://www.petefreitag.com/cheatsheets/postgresql/

Create database::

    CREATE DATABASE dbName;

Create table (with auto numbering integer id)::

    CREATE TABLE tableName (
     id serial PRIMARY KEY,
     name varchar(50) UNIQUE NOT NULL,
     dateCreated timestamp DEFAULT current_timestamp
    );

Add a primary key::

    ALTER TABLE tableName ADD PRIMARY KEY (id);

Create an INDEX::

    CREATE UNIQUE INDEX indexName ON tableName (columnNames);

Backup a database (command line)::

    pg_dump dbName > dbName.sql

Backup all databases (command line)::

    pg_dumpall > pgbackup.sql
```

Run a SQL script (command line)::

```
psql -f script.sql databaseName
```

Search using a regular expression::

```
SELECT column FROM table WHERE column ~ 'foo.*';
```

The first N records::

```
SELECT columns FROM table LIMIT 10;
```

Pagination::

```
SELECT cols FROM table LIMIT 10 OFFSET 30;
```

Prepared Statements::

```
PREPARE preparedInsert (int, varchar) AS
  INSERT INTO tableName (intColumn, charColumn) VALUES ($1, $2);
EXECUTE preparedInsert (1,'a');
EXECUTE preparedInsert (2,'b');
DEALLOCATE preparedInsert;
```

Create a Function::

```
CREATE OR REPLACE FUNCTION month (timestamp) RETURNS integer
 AS 'SELECT date_part(''month'', $1)::integer;'
LANGUAGE 'sql';
```

Table Maintenance

```
VACUUM ANALYZE table;
```

Reindex a database, table or index::

```
REINDEX DATABASE dbName;
```

Show query plan::

```
EXPLAIN SELECT * FROM table;
```

Import from a file::

```
COPY destTable FROM '/tmp/somefile';
```

Show all runtime parameters::

```
SHOW ALL;
```

```
Grant all permissions to a user::

    GRANT ALL PRIVILEGES ON table TO username;

Perform a transaction::

    BEGIN TRANSACTION
     UPDATE accounts SET balance += 50 WHERE id = 1;
    COMMIT;

Basic SQL
--------------------

Get all columns and rows from a table::

    SELECT * FROM table;

Add a new row::

    INSERT INTO table (column1,column2)
    VALUES (1, 'one');

Update a row::

    UPDATE table SET foo = 'bar' WHERE id = 1;

Delete a row::

    DELETE FROM table WHERE id = 1;

SQL-kezelése Pythonból
============================

Lásd::

    hp/alapsql.cgi
    hp/htmltabla.py
```

10.1. Elérése Pythonból

Az alábbi programfálban nyomon követhetjük, hogyan érhetjük el Python alól az adatbázisunkat:

```
#! /usr/bin/python
# coding: utf-8
from __future__ import division

"""Egyszerű program az adatbázis elérésére Pythonból."""
```

```
# python-pygresql csomag kell hozzá
# (apt-get install python-pygresql)

# http://www.pygresql.org/pgdb.html

import pgdb
# Ha más adatbázis van, akkor csak ezt a sort kell változtatni. Pl:
# import oracledb

connection = pgdb.connect(database='diak', user='diak',
                    password='diak', dsn='mail.roik.bmf.hu')

cursor = connection.cursor()
cursor.execute("SELECT * FROM betétesek;")
connection.commit() # Csak SELECT parancs esetén nem fontos

lekert_adatok = cursor.fetchall()

for nev, cim, egyenleg in lekert_adatok:
    print nev, cim, egyenleg
    #print nev.left(20), cim.left(20), egyenleg.left(5)
    #print "{0:20} | {1:20} | {2:5}".format(nev, cim, egyenleg))
```

Az utolsó megjegyzésbe helyzett sorokkal szebb formátumba tudjuk írni. Próbáljuk megérteni az adott sorokat!

11. LaTeX

Töltsök le githubról a horvatha/latex következő tárolót:

```
git clone https://github.com/horvatha/latex.git
```

A `latex/peldak/keret.tex` fájl jó kiindulófájlnak.

A `latex/latex-roviden` könyvtárban a `make` paranccsal fordítsuk le a rövid összefoglalót. A benne leírtakat kell tudni használni.

Ennek a fájlnak a forrása a linux tárolóban van, ha magunk akarjuk lefordítani, töltsük le a tárolót:

```
git clone https://github.com/horvatha/linux.git
```

A tikzpicture ábrákhoz szükség van a pgf csomag telepítésére:

```
apt-get install pgf
```

Nyissuk meg ennek a fájlnak a forrását és fordítsuk le:

```
cd linux/segedlet
vim linalk.tex
:!pdflatex linalk.tex
:!evince linalk.pdf &
```

A :! a vim-ben lefuttaja az utána lévő parancsot, mintha shellbe írnánk.

A pdflatex-et Vimben egyszerűbben is futtahatjuk, mivel a `%` az aktuális fájlt jelöli, sőt – mivel a fenti fájl a Makefile-ban is benne van – make-kel is fordíthatom, de ilyenkor a célfájlt kell megadni:

```
:!pdflatex %
:make linalk.pdf
```

Mint látjuk a make-hez nem kell felkiáltójel Vimben.

Újraforsításkor a régebbi Evincében a `<Ctrl>+R` frissít. Újabbakban automatikusan frissül.

Kis kedvcsinálónak nézzük meg a latex tárolóból a `latex/latex-izelito/sziv.pdf` fájlt. Ezek inkább érdekességek, hogy mit lehet csinálni a LaTeX-hel. A forrás is ott van `sziv.tex` néven. Szintén lefordíthatjuk

`pdflatex sziv.tex`

utasítással.

Menjünk át a latex tároló latex-roviden könyvtárába, és fordítsuk le a latex-roviden.tex fájlt pdflatex-hel!

Másoljuk át a latex-roviden.tex-et más névre, és a benne leírtak és a forrás segítségével:

- Írjunk még pár bekezdést!
- Hozzunk létre új fejezetcímeket!
- Helyezzük át a tartalomjegyzéket máshová!
- Töröljük a felesleges részeket!
- Állítsuk be magunkat szerzőnek és írjuk át a címet!
- Szúrjuk be a Pitagorasz-tétel képletét `$`-ral és `equation` környezettel, valamint a trigonometrikus Pitagorasz-tétel!
- Hozzunk létre táblázatot!
- Szúrjunk be ábrát! Állítsuk be a szélességét fél `\textwidth`-re, 6 cm-re! Esetleg forgassuk el!
- Hozzunk létre verbatim szövegrészt környezettel és `\verb` paranccsal!
- Későbbre: Hozzunk létre egy LaTeX-táblázatot psql paranccsal, és hozzuk be `\input` utasítással!

  ```
  psql -q -P format=latex -P border=3 -c "SELECT * FROM betetesek;"
  ```

  (Lásd még `linux/postgresql/bank/kimutatas.sh` fájlt.)

12. Megjegyzések, nem kötelező anyag

- `pygmentize -O full -o bashrc.tex -f tex -l bash ~/.bashrc`
- latex2html: Esetleg fordítsuk le a `linux/latex/peldak/komplex/komplex.tex` fájlt weboldalakká. Segít a `Makefile`.

13. A make használata

A `make paranccsal` lehet több egymástól függő fordítást irányítani. Eredetileg a C-forrásfájlok fordítására készült. Képes arra, hogy csak a frissült C-forrásfájlokat fordítsa újra. Hasznos lehet például LaTeX-forrásfájlok fordításánál is.

A `make` parancs a `Makefile` vagy `makefile` fájlt keresi az aktuális könyvtárban, és azalapján dolgozik. A `latex/peldak/komplex` könyvtárban található egy példa:

`Makefile`

```
komplex.pdf: komplex.tex
        pdflatex komplex.tex
        pdflatex komplex.tex

html: komplex.tex
        latex2html komplex.tex

clean:
        rm *.log *.aux
```

Ebben az esetben, ha sima `make` parancsot adok ki, akkor a legelső szakasz hajtódik végre, kétszer lefordítja a LaTeX-fájlt PDF-be, de csak akkor, ha a `komplex.tex` fájl újabb, mint a `komplex.pdf`, vagy egyáltalán nincs `komplex.pdf`. Ugyanez a helyzet, ha az alábbi első sort írtuk volna be:

```
$ make komplex.pdf
$ make html
```

A második sor HTML-be fordítja a LaTeX-fájlt tartalmát. A html nem egy létező fájlnév (nem is jön létre), tehát bármikor hívva végrehajtódik az utána található parancs.

Vimben kettőspont után írhatunk make parancsokat, mint shellben. Kettőspont sem kell make elé.

A szintaktákánál fontos, hogy a parancsokat tartalmazó rész tabulátorral legyen behúzva. Végrehajtáskor minden egyes parancssor külön shellben kerül megnyitásra, tehát nincs mód a burokváltozók felülírására.

A make parancs lehetőséget ad arra is, hogy több gépen fordítsunk le egy nagyobb – sok C- vagy C++-kódot tartalmazó – forrást. Ehhez a `distcc` parancs szolgál. Bővebb leírás erről és általában a make-ről az alábbi oldalon található prezentációban:

`http://cern.ch/ahorvath/prez/make_rpm`

14. Emberi nyelvek, karakterkódolások

Házi feladat köv alkalomra: Ismerkedés a Pythonnal `http://pythontutorial.pergamen.hu` honlap alapján. Átnézni az alábbi fejezeteket: 3. Kötetlen bevezető... 5. Adatstruktúrák

14.1. Emberi nyelvek

Otthon beállíthatjuk, hogy ne kelljen mindig külön konzolt nyitni a rootnak:

- Ezegyszer utoljára jelentkezzünk be rootként.
- Állítsuk be a sudo-t: `visudo` parancs
- Másoljuk át a root-os sort (`yyp`)
- Az új sorban cseréljük a `root` szót `diak`-ra (`cw`)
- Lépjünk ki (`<Esc>:wq`)

Ekkor mindenre hatalmunk lesz diak felhasználóként is, ha a parancs elé biggyesztem a sudo parancsot. Szerveren nem illik ez a mindent megengedő beállítás, ott a rendszergazda csak egy-egy részterületet (webszerver kezelése, adatbázis mentése) szokott másra bízni.

Visszatérve normál felhasználói módba – állítsuk be, milyen nyelveket ismerjen a gép, legyenek ezek (magyar latin2-es és UTF-8: hu_HU, hu_HU.UTF-8 német de_DE, amerikai: en_US):

```
sudo dpkg-reconfigure locales
```

Miután végeztünk, nézzük meg a /etc/locale.gen tartalmát!

Amíg tovább dolgozunk, futtassuk le egy másik ablakban a `sudo updatedb` parancsot.

Vannak olyan programok, amelyek a LANG változót nézik, mások a LANGUAGE változót. Nézzük meg a tartalmukat: `export | grep LANG`. Próbáljuk meg működnek-e (nem kell mindet kipróbálni):

```
export LANG=de_DE
vimtutor (kilépés :q)
abiword&        (ha van ilyen)
export LANGUAGE=de_DE
vim   (majd kilépés)
gimp &
```

Miután kipróbáltuk, állítsuk mindkettőt vissza hu_HU-ra.

A magyarítási fájlok egy .mo kiterjesztésű fájlban találhatóak. Amely a szerkeszthető .po fájlokból készülnek. Töltsük le a http://mail.roik.bmf.hu/linux/vim oldalról a vim.po fájlt (és a vimrc-t mentsük el /.vimrc néven, ha még nem tettük). Később kell majd ennek a szülőkönyvtárából a galaktos_utf-8 nevű fájl is.

Ha megnézzük a tartalmát, akkor msgid sorokban angol, msgstr sorokban magyar szövegeket találunk. Ilyen – de magyar szövegek nélküli – fájl hozható létre a Vim és más programok forráskódjából, amelyet mindenféle nyelvre le lehet fordítani. Az updatedb bizonyára lefutott, ez egy adatbázist hozott létre a

fájlok/könyvtárak neveiből, ami alapján gyorsan kereshetünk fájlokat, keressük meg a `vim.mo` fájlok helyét:

```
locate vim.mo
```

Látszik, hogy még hu-s változat nincs. Fordítsuk le a gép által ismert .mo fájllá a .po fájlunkat. (Ehhez a gettext csomag kell.) Másoljuk be a megfelelő helyre. És indítsuk el magyar Vimünket.

```
msgfmt -o vim.mo vim.po
sudo mkdir -p /usr/share/vim/vim63/lang/hu/LC_MESSAGES/
sudo cp vim.mo /usr/share/vim/vim63/lang/hu/LC_MESSAGES/
vim
```

(A mkdir a -p opcióval úgy csinál könyvtárat, hogy a még nem létező szülőkönyvtárakat is létrehozza.)

Szerkeszthetjük is a fájlt szövegszerkesztővel, de hosszabb fájlok esetén a kbabel, gtranslator vagy poedit használata könnyíti a dolgunkat. Ezeken a gépeken a kbabel és a gtranslator van fent, csak ízelítésképpen megpróbálkozhatunk a használatával. (`kbabel vim.po`)

Csináljunk mi fordítható programot! Töltsük le a hello.py fájlt a szokásos linalk könyvtárból, és kövessük az elején található utasításokat. (Egy Python-programot magyarítunk, bővítjük a programot, és ismét magyarítjuk.)

Miért érdemes ismerni ezeket a dogokat? Előfordulhat, hogy a cégnek, ahol dolgozunk szüksége lenne egy olyan szabad szoftverként meglevő programra, amely még nincs lefordítva, és a felhasználók nem tudnak (kellő mértékben) angolul. Megkereshető általában az interneten egy CVS vagy SVN archívumban a .pot fájl, amiből megcsinálhatjuk a mi .po fájlunkat, lefordíthatjuk és már van működő magyar alkalmazásunk. Természetesen örülnek a fejlesztők, ha visszaküldjük. A fordításnak van egy webes változata, a launchpad.net, ahol elsősorban az Ubuntuban található csomagok fordítása folyik, melyek közül a legtöbb nem csak az Ubuntuban szerepel (pl. a Vim, a GIMP, és a Firefox M$ Windows alatt is megy).

Előfordulhat az is, hogy mi szeretnénk fordítható programot összeütni.

15. A fájlrendszer

15.1. A fájlrendszer

Közösen átnézni:
/etc: fstab group profile crontab apt/sources.list
/bin zgrep (`file /bin/*`)
/usr: local share share/doc
/var: log (syslog, dmesg) www cache/apt/archives cache/locate
/tmp /home /root /sbin /dev /mnt (cdrom, floppy...) /lib

16. Képek kezelése

Képtípusok raszter/vektoros, tömörítés típusai.

16.1. Képek átalakítása az Imagemagick programokkal

Töltsük le a bash.tar.gz fájlt. Ennek kicsomagolásakor a bash/kepek könyvtárban vannak képek.

Alakítsunk át egy képet másik formába:

```
convert valami.png valami.jpg
```

Kérjünk adatokat a képeinkről:

```
identify *
```

Tanulmányozzuk a `bash/kepek/kepatalakito` nevű szkripttel a `convert` lehetőségeit.

Ehhez ismerni kell a következőket. A `basename` utasítás használata. Az argumentumlista kezelését tanulmányozzuk a `bash/argumentum_lista` fájljain.

Sok kép Exif dátumainak módosítása (ha nem jól járt a fényképezőgép órája): `exiv2` parancs/csomag `-a` opció.

16.2. Képek kezelése a gthumb programmal

Nyissuk meg a gthumb programmal a képek könyvtárat. Adjunk megjegyzést a képekhez (c billentyű). Listázzuk a rejtett állományokat a képek könyvtárában (hova kerülnek a megjegyzések?).

Jelöljük ki a képek egy részét. (`Ctrl`- ill. Shift-katt) és csináljuk egy honlapra kitehető galériát ezekből: Eszközök > Webalbum létrehozása

A. Az AREK könyvtárában fellelhető irodalom

A darabszámok az éppen bent találhatóak, azaz minimumadatok.

A.1. UNIX általános/parancssor

Bartók: UNIX felhasználói ismeretek 3db
Harley: UNIX bevezetés/haladó 3db
Kernighan – Pike: A UNIX operációs rendszer, 1987, 6db, régi de alapos
Bagoly: UNIX alapismeretek 1995 5db
Büki: Héjprogramozás

A.2. Linux

Petersen: Linux, 1997, 2db, Parancssor mellett, (felhasználó/hálózat)adminisztráció, TeX

Gagné: Válts Linuxra! Búcsú a kékhaláltól, 2004, Hangsúlyosabb a grafikus felület, KDE

A.3. Speciális területek

Flickenger: Linux bevetés közben, 2003, Tippek
Butzen: Linux hálózatok
Kirch: Linux hálózati adminisztrátorok kézikönyve
Friedl: Reguláris kifejezések
Pere László: Linux felhasználói ismeretek I.
Pere László: Linux felhasználói ismeretek II, Adatkezelés (awk, bc, adatbázis)
Pere László: GNU/Linux rendszerek üzemeltetése I. és II.
Gagné: Linux rendszerfelügyelet

B. Hasznos fogások

Levél küldése bash-ből mail (∈ mailx).

B.1. Érdemes lehet a .basrc-be tenni

/.bashrc

```
alias sshmaildiak1="ssh -p xx22 diak1@mail.roik.bmf.hu"
alias cx="chmod a+x"
alias ipythonlab='ipython -pylab'

export EDITOR="vim"  # A vim lesz az alapértelmezett szerkesztő
export LESSEDIT="%E ?lt+%lt. %f" # lessből a vim az adott sortól kezd
                                 #szerkeszteni
export CDPATH=.:~   # cd parancsnál a homekönyvtárban is keres
```

B.2. mc (Midnight Commander $\in$ mc)

Beállítások > alapbeállítások... > Lynx-hez hasonló navigálást érdemes beikszelni, akkor a jobb nyíllal be lehet menni a könyvtárakba, bal nyilakkal szülőkönyvtárba $\Rightarrow$ nagyon gyors navigálás.

Ha a belső szövegszerkesztő elől kivesszük az x-et, akkor az alapértelmezett szövegszerkesztőnket kapjuk `<F4>`-re. Fentebb van, hogyan állíthatjuk ezt vim-re a .basrc-ben.

Grafikus felületen gyakran az F10 gomb másra foglalt. F9-cel kell menübe menni Fájl > Kilépés-sel lehet kilépni. Fájlokból kilépés két `Esc`-el.

Másik mód a GNOME-terminál esetén annak beállításával: Szerkesztés > Gyorsbillentyűk: Menüelérés billentyűjének (F10) tiltása

B.3. Pár dolog, amit együtt érdemes megjegyezni

/	keresés	Firefox, Vim, man, less, info
n és N	adott irányban illetve ellenkező irányban keresi a következő találatot	Vim, man, less
q	kilépés	man, less, mutt, info esetén
:q	kilépés	Vimből, ha nem volt módosítás
:q!	kilépés	Vimből, ha nem mentem a módosításokat
Ctrl+D	kilépés	shellből (ssh-t ill. terminál-ablakot bezárja); szövegbeíráskor egy szűrő (pl. cat, mail, grep) után, ipython, python shell
<F11>	teljes képernyő	GNOME terminál; Firefox; evince
Ctrl+ <PageUp> Ctrl+ <PageDown>	váltás a lapok között egyik ill. másik irányban	GNOME terminál; Firefox
Alt+1 ... Alt+9	adott sorszámú lapra ugrik	GNOME terminál; Firefox

B.4. GNOME, KDE és LXDE együtt Ubuntun

Akár melyik változatot telepítettük az `ubuntu-desktop`, `kubuntu-desktop`, `lubuntu-desktop` csomag telepítésével megkapjuk az adott desktop környezetet.

B.5. `sudo` Debianon is

Otthon beállíthatjuk Debian alatt is, hogy ne kelljen mindig külön konzolt nyitni a rootnak, hanem sudo-val dolgozhassunk:

- Ez egyszer utoljára jelentkezzünk be rootként.
- Állítsuk be a sudo-t: `visudo` parancs
- Másoljuk át a root-os sort (`yyp`)
- Az új sorban cseréljük a `root` szót `diak`-ra (`cw`)
- Lépjünk ki (`Esc` `:wq`)

Ekkor mindenre hatalmunk lesz diak felhasználóként is, ha a parancs elé biggyesztem a sudo parancsot. Szerveren nem illik ez a mindent megengedő beállítás, ott a rendszergazda csak egy-egy részterületet (webszerver kezelése, adatbázis mentése) szokott másra bízni.

Tartalomjegyzék