Alapok

A Qt egy sok operációs rendszeren (pl. Windows, Linux, Android, Mac OS) használható grafikus fejlesztői könyvtár. Lényegében bármely operációs rendszeren írunk vele programot az szinte azonnal lefordítható egy másik operációs rendszer alatt. Én a félév folyamán mint legkényelmesebbet a Microsoft Visual Studio 2017-et fogom használni a Qt VS Tools kiegészítővel. Aki ezt nem akarja bármely operációs rendszer alatt használhatja a QtCreator-t.

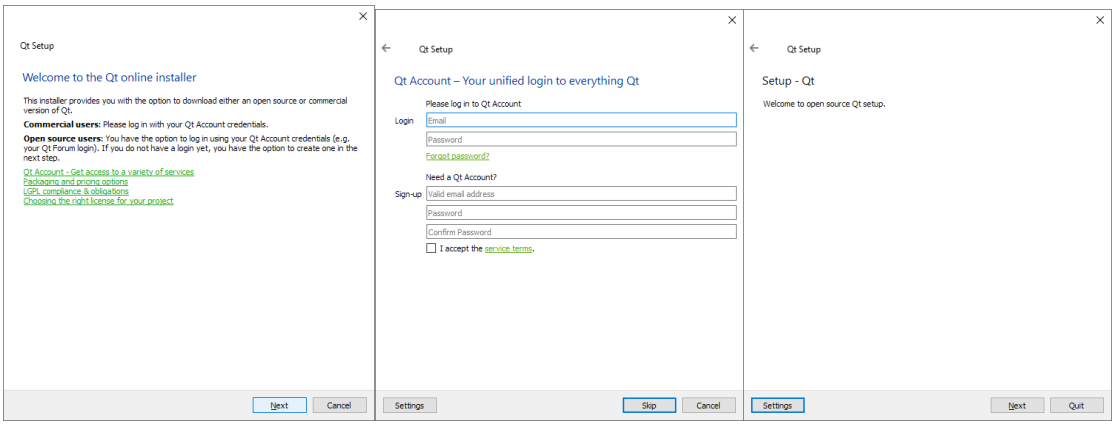
A Qt letöltése és telepítése

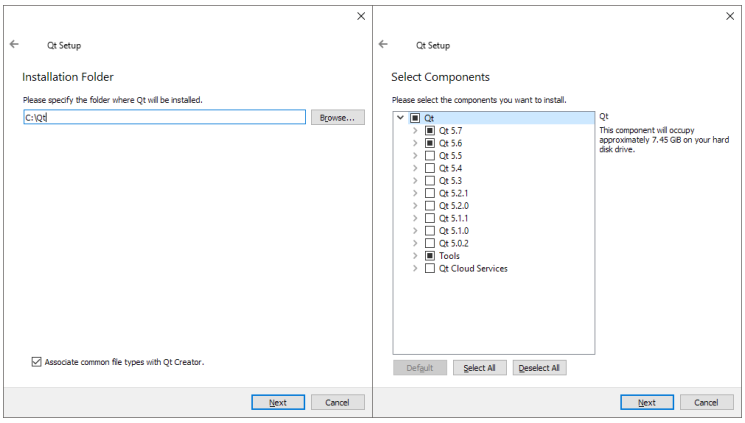
A Qt letöltéséről és telepítéséről mindenkinek magának kell gondoskodnia. A használt operációs rendszertől függően (Windows, Linux, Mac OSX) ez más és más feladatot jelent.

Windows:

Windows alatt futó verzió telepítése.

A Qt rendszer online telepítője innen tölthető le. A telepítés lépései:





Javaslom, csak az egyik verzió (pl. Qt 5.14.2) – (annak mindkét, 32 és 64 bites variációjának) telepítését, mert a telepítő sokszor nem elég okos ahhoz, hogy több

verzió telepítése esetén az eszközök ne akadjanak össze[1]. A telepítésnél választanunk kell milyen fordítóprogramot használunk majd, de a telepítő fel is ajánl olyanokat, amik már vannak a gépen. A félév folyamán mi a Microsoft Visual Studio 2017-t (az ingyenes „community" verzióval is) használjuk, működik együtt. A két program össze is kapcsolható, ha a VS2017-hez letöltjük ezt a Visual Studio-ba beépülő programot is, amelyikkel magából a Visual Studioból hozhatunk létre QTs programokat.

Ha nincs Visual Studio-nk, vagy csak nem akarjuk telepíteni, akkor válasszuk a megfelelő 32, ill. 64 bites MINGW opciót a telepítőben.

Ha valaki nem akar Visual Studio-t használni, annak kényelmetlenebb dolga lesz.

A Qt-hez a Qt Creator az alapértelmezett fejlesztő rendsze , azt mindenképpen érdemes feltelepíteni.

Linux:

Linux-os rendszerből sokféle van, ezeket *disztribucióknak* hívják. Legismertebbek az Ubuntu, a Mint, az Open SuSE, a Red Hat, vagy a Debian, de szinte naponta jelennek meg új kedvencek. Az egyes disztribúciók a mi szempontunkból egyenrangúak.

Linuxok telepítése során választhatunk különböző ún. *csomag*ok közül. Egy ilyen csomag tartalmazza a fejlesztőeszközöket (fordító programokat, integrált fejlesztői környezeteket – IDE-ket, stb) és a Qt-t is. Csak annyit kell tennünk, hogy a C++ fordítót és a Qt5-öt is kiválasztjuk telepítésre.

Akinek eleve Linux van a gépén, az valószínűleg tudja, miként kell ezt utólag is megcsinálni. Aki e tárgy keretein belül akar a Linux-szal megismerkedni, annak magának kell a rendszert felrakni, mert arra nincs idő az órán.

A Linux használata

T0bb oka is van, hogy még annak is érdemes a Linux alapjaival megismerkedni, aki a hétköznapokban (ma még) nem használja:

- Qt-vel Linuxban is fejleszthetünk,
- minden új Windows 10-es rendszernek van egy Linux-os alrendszere
- sok (persze messze nem mind) feladat a parancssor segítségével kényelmesebben és gyorsabban végezhető el, mint a grafikus programokkal

Linux alatt lényegében ugyanúgy lehet dolgozni, mint Windows, vagy Mac OS X alatt egy grafikus képernyőn, ablakokkal menükkel egérrel és billentyűzettel. A Linux esetében azonban érdemes a parancssorral is megismerkedni, egyrészt, mert távolról ez még lassú internet kapcsolat esetén is jól

[1] Az 5.15-ös verzió csak a Visual Studio 2019-et támogatja. Aki a Qt Creatort akarja használni, annak a legújabb verzió a legjobb választás.

használható, másrészt sok feladat egyszerűbben oldható meg a karakteres képernyőn, mint a grafikuson.

A Linux grafikus képernyőjéről karakteres üzemmódba kétféleképpen léphetünk át:

1. A *Ctrl + Alt + 1, Ctrl + Alt + 2, … Ctrl + Alt + 6* gombkombinációkkal. Ezek mindegyike egy karakteres képernyőre kapcsol át, ahol bejelentkezhetünk a gépbe egy karakteres terminálra.. Visszalépés a grafikus ablakra a *Ctrl + Alt + 7* kombinációval.

2. A grafikus képernyőn egy terminál ablak megnyitásával. A terminál program neve minden disztribúcióban más lehet. Akárhány terminál ablakot nyithatunk.

Terminál ablak megnyitása a menüből és annak maximalizálása.

A terminál ablakban egy ***parancsértelmező program*** (más néven '*shell*'), ami egy nem grafikus interfész, fut – azaz gépelni kell. A shell a mi esetünkben és általában a *bash*, de vannak mások is. Az ablakban egy parancs váró szöveg (command prompt) látható utána villog a kurzor. Ez a prompt többnyire a következő szerkezetű[2]:

felhasználó@gép:/aktuális/könyvtár/elérési/útja$

Pl ha a felhasználónevünk *tigris*, a gépünk neve *dzsungel* és az aktuális könyvtár a saját könyvtárunk, (aminek az elérési útvonala ekkor */home/tigris,* aminek a rövidítése ~) ezt látjuk:

tigris@dzsungel:~ $

Mögötte ott villog a kurzor várva, hogy begépeljünk egy vagy több parancsot. Az ENTER/RETURN megnyomásával hajtatjuk végre a parancsokat. A parancsokban a *kis- és nagybetű különböző*! A rendszerparancsok kisbetűsek.

Trükkök:

Korábbi parancsok visszahozása: *Felfele nyíl.* Ha már túlmentünk a parancson, akkor *Lefele nyíl*, parancs keresése: *Ctrl + P* és utána *parancs részlete,* ha nem az jön elő, amit vártunk, akkor ismét *Ctrl + P.*

Parancssorban szó törlése visszafelé: *Ctrl+W, vagy Ctrl+U.* Ha egy fájl nevét elkezdjük begépelni és csak egy olyan fájl van, akkor a TAB kiírja a teljes nevet, ha több fájl név is van ami ugyanúgy kezdődik, akkor egy második TAB-ra kiírja az összeset, hogy tudjuk folytatni a gépelést.

Fájlrendszerek

Linuxban a könyvtárak (vagy mappák) és fájlok – a Windowsban megszokotthoz hasonlóan - egy hierarchiába vannak rendezve. Azonban Linuxban nincsenek meghajtó betűk. Minden meghajtón

[2] ez, mint szinte minden Linux alatt átállítható

levő minden *fájlrendszer*[3] ugyanabba a hierarchiához kerül be. A hierarchia legfelső szintje (gyökere) a / könyvtár. Egy fájl, vagy könyvtár *teljes elérési útja* a legfelső szinten kezdődik, (példa: */srv/www/htdocs/index.html)*[4].

Ha pl. van három merevlemezünk amiknek a neve Windows alatt 'C','D' és 'E' lenne, akkor ezek egyikének tartalmát a '/' könyvtárban találjuk, a másik kettőt pl a */mnt/zenék* és */mnt/filmek* könyvtárban.

Fájlok és könyvtárak neve nem tartalmazhatja a '/' karaktert és nem célszerű a következőket sem használni: ?:!|*\. A \ (*backlash*, vagy *rep)* karakter egy olyan speciális karakter, mint a C nyelvben. Pl ha egy fájl neve betűközt ás '?' –t tartalmaz, akkor a tulajdonságait így listázhatjuk ki:

ls –l alma\ a\ fa\ alatt\?,

vagy így[5]:

ls –l 'alma a fa alatt?'.

A pont '.' egy rendes karakter. Linuxban nincsenek „kiterjesztések", a fájl nevében az utolsó pont utáni karaktereknek csak egyes programok számára van jelentése. Ha a fájl neve „.exe"-re végződik, attól még az a fájl nem feltétlen végrehajtható. Azt a fájlt *próbálja meg* a rendszer futtatni, aminek van megfelelő *jogosultsági bitje*. Az, hogy egy fájl típusa mi azt a *file* paranccsal kérdezhetjük le, pl

```
$ file /usr/bin/ls

/usr/bin/ls: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked (uses
shared libs), for GNU/Linux 3.0.0,
BuildID[sha1]=9fb6916841caa41f75f932e66e8c6fe8e9242dc6, stripped

$
```

A Linuxok többsége hasonló könyvtárszerkezetet és neveket használ:

/ - a fájlrendszer gyökere
├── bin
├── boot
├── dev - itt vannak az eszközök. Pl az első HD neve /dev/sda, az első particiójáé /dev/sda1
├── etc - ez tartalmazza a konfigurációs fájlokat. Itt van pl. a *passwd* és a *group* fájl is.
├── home – minden felhasználónak van egy saját könyvtára itt, pl /home/jozsi
├── lib
├── lib64
├── mnt - ide csatolhatunk fel külső eszközöket a *mount* parancssal
├── opt
├── proc - az itteni „fájlok" a rendszerről adnak információt
├── root - a root felhasználó home könyvtára. El van különítve a meghajtó többi fájljától

[3] A fájlrendszer egy olyan logikai fizikai struktúra, amiben fájlokat lehet tárolni. Windows alatt pl az NTFS, CD-ken és flash driveokon más fájlrendszerek vannak. Linux alatt a különböző eszközökön többféle fájlrendszert is használhatunk egyszerre.

[4] A Windows parancsértelmezője a / helyett a \ karaktert használja a fájl elérési útjában, de maga a Windows megérti a / karaktert is. Ezért, ha olyan programot akarunk írni ami Windowson és Linuxon egyaránt fut, akkor használjunk / karaktereket!

[5] Ha nem lenne '?' benne akkor használhatnánk dupla idézőjelet is. Azzal most az "*alma a fa alatt\?*"-t kellene használjuk.

```
├── run
├── sbin
├── selinux
├── srv  - itt szoktak a szerverek (pl WEB szerver) könyvtárakat létrehozni
├── sys
├── tmp – ideiglenes fájlokat ide célszerű rakni
├── usr
│   ├── bin - rendszerparancsok
│   ├── games
│   ├── include – C/C++ include fájlok
│   ├── lib         - pl Cs lib fájlok
│   ├── lib64
│   ├── local
│   ├── sbin
│   ├── share
│   ├── src
│   ├── tmp -> ../var/tmp – link a tmp könyvtárra
│   ├── X11R6    - a grafikus képernyő programja
│   └── x86_64-suse-linux
└── var  - mindenféle változó hosszúságú adat, pl naplók
```

/home

Minden felhasználónak van egy saját könyvtára. Ebben létrehozhat alkönyvtárakat és fájlokat. Bejelentkezés után ide kerülünk, ez lesz az *aktuális könyvtár.* Ha a saját könyvtárunk az aktuális könyvtár akkor a *prompt*ban egy ~ karakter van:

tigris@dzsungel:~$

Néhány fontos Linux parancs (nem ABC-sorrendben)

Konvenció: parancsok leírásánál a következő konvenciót fogom használni:
<szöveg> - kötelező paraméter. Ide a szövegnek megfelelő nevet, stb kell beírni. Pl *ls <könyvtár>* azt jelenti, hogy a *könyvtár* szöveg helyére egy könyvtárnevet kell írni: *ls /dev*
[opcionális] – nem szükséges, de megadható paraméterek
| (függőleges vonal, vagy „pipe") vagy a baloldalán, vagy a jobboldalán levő paramétert kell használni
Minden szabadon álló szöveg szó szerint veendő. Pl. *ls [-l|c] alma.txt* lehet ezek bármelyike: *ls alma.txt, ls –l alma.txt, ls –c alma.txt*. Ha egy opcionális paraméteren belül is megjelenhet szó szerinti szöveg és paraméter is, akkor mindkét zárójelezést használom, pl *alma [-p<paraméter>],* vagy *alma [-p<paraméter1> […]]*

A Linuxos parancsok többsége sokféle opciót ismer. Az opciókat általában egy, vagy két mínusz jel előzi meg. Sokszor több opció összevonható, pl *–o –d* ből lehet *–od*. Sok parancsnak van súgója, amit a *–help* opcióval kaphatunk meg. Ezenkívül ha telepítve vannak a dokumentációk is, akkor a *man* <parancsnév> ezt a súgót kiírja. Kilépni belőle a *q* gombbal lehet.

man – parancsok súgója. PL. *man grep* Ha egy parancsról többféle dokumentáció is van, akkor választhatunk melyiket akarjuk megnézni. Ebben a példában két opció van: 1 és 1p

pwd - Nézzük meg melyik könyvtárban vagyunk: pw*d* (print working directory)

cd - Menjünk vissza a saját könyvtárunkba bárhonnan: *cd*

ls[6] - Listázzuk ki az aktuális könyvtár tartalmát: *ls.* Vagy mondjuk az összes *cpp* fájlt az adott könyvtárban: *ls *.cpp.*

Ez a lista túl tömör, nézzük meg így is: *ls –l* *.cpp
A sorok szerkezete:

<jogosultságok> <hard linkek száma[7]> <felhasználó> <csoport> <méret> <datum> <név>

Pl.

-rw-r----- 1 felhaszn csoport 1234 Okt 19 2019 main.cpp

A josultságok első karaktere utal a listaelem típusára, (*'d'* – könyvtár (directory), *'l'* – (soft) link egy fájlra, vagy könyvtárra, *'-'* közönséges fájl). Ezt követi 3 jel 3 csoportja, ami vagy betű, vagy a '-' jel. Ezek adják meg a hozzáférési jogokat. Az első három a tulajdonos, a második három a tulajdonos csoportjának[8], a 3. három mindenki másnak a jogai. Az első jel ('r' - read) az olvasási jog, a második ('w' –Write) az írási jog, a harmadik (x - eXecute) a végrehajtási/keresési jog. A '-' jel azt jelenti, hogy nincs az adott jog.

Pl. a fenti rw-r----- azt jelenti, hogy a '*felhaszn*' nevű tulajdonosnak (és a *root* -, azaz az adminisztrátor felhasználónak!) joga van írni és olvasni a fájlt, a *'csoport*' csoport többi tagja csak olvashatja, mások egyiket sem tehetik meg.

Linuxban a futtatható fájlokat az x jog jelzi. Mivel bármilyen fájlnak bármilyen jogosultságokat megadhatunk, ha egy fájlnak megadjuk ezt a jogosultságot attól még nem feltétlenül lesz futtatható.

A jogok jelentése könyvtárak esetén:

r - listázhatjuk milyen fájlok vannak a könyvtárban

w – bármilyen fájlt létrehozhatunk, ill. kitörölhetünk a könyvtárból, *még akkor is*, ha nem nézhetjük meg milyen fájlok vannak ott.

x – beléphetünk a könyvtárba, illetve megnézhetjük a fájl adatait is a listázásnál.

---

[6] Az ***echo** ** parancs ugyanazt adja ki, mint az *ls*, de nincs hosszú lista módja

[7] 'hard link' – egy fájlnak több neve is lehet és ezek más és más könyvtárakban jelenhetnek meg. Ezeket hívjuk 'hard link'-nek. Egy fájl mindaddig létezik, amíg legalább egy hard link-je van. Léteznek 'soft link'-ek is, amik maguk is fájlok és egy másik fájlra mutatnak. Szemben a hard linkekkel nem mindegy melyik fájlt töröljük le. Ha a fájlt, amire a soft link mutat, akkor a fájl letörlődik és a link onnantól a levegőben lóg. Ha viszont a linket, akkor a fájl megmarad.

[8] Egy felhasználó több csoport tagja is lehet, ezek közül van egy elsődleges csoportja, a listában ez látszik.

chmod – Saját fájljaink és könyvtáraink hozzáférési jogainak módosítása:

chmod <kinek>+|-<jog>, ahol a <kinek> egy betű (u – felhasználó, g – csoport, o vagy a - mindenki más), a + a jog megadását, a – az elvételét jelenti és a <jog> az r,w,x betűk valamelyike lehet[9]. Példa: *chmod g-w alma.txt* csoporttagok sem írhatják a fájlt. (De letörölhetik, ha a fájl tartalmazó könyvtárhoz van írási joguk!)

cat – szöveges fájlok tartalmának listázása az ablakba: *cat ./ alma.txt*. Az összes fájl kilistázása: *cat *.*

cp –fájl másolása pl *cp ./innen/alma ./ide/korte* Ha valaki más fájlját átmásoljuk magunkoz az a mi fájlunkká válik és a fájl dátuma is a másolás dátuma lesz. Ha meg akarjuk tartani az előző dátumot, de a tulajdonost nem akkor a *cp --preserve=mode, timestamps alma korte* parancsot, ha a tulajdonost is meg akarjuk tartani akkor pedig a *cp -preserve=mode,ownership,timestamps alma korte,* vagy *cp –p alma korte* parancsot

mv – fájl átnevezés átmozgatás máshova, pl *mv alma korte*

less – fájl tartalmának listázása, úgy, hogy a nyíl /lapozás és betűköz gombokkal mozoghatunk a fájlban és kereshetünk benne karaktereket is a / megnyomása után. Ez a keresés nem érzékeny a kis és nagybetűre. Még két hasznos gomb: *G* a fájl végére, *g* az elejére ugrik.

grep – *szövegminták* keresése szövegfájlokban soronként. A *szövegminták* lehetnek egyszerűen szövegek, de más ún. *reguláris kifejezések* is. Ha nem adunk meg speciális opciókat, akkor a *grep* kiírja azokat a sorokat amelyekben megtalálta a megadott reguláris kifejezést. Célszerű a reguláris kifejezést *egyes idézőjelekbe* (') rakni.
A *grep* szintaxisa: *grep [opciók] <reguláris kifejezés> <fájlok listája>.* A fájlok listája lehet egy * karakter, amit a shell lecserél az összes fájl listájával, vagy pl. *a*b*.txt* ami azon szövegfájlok listája, amelyek neve a-val kezdődik és van benne valahol egy b is. A következőket kereshetjük:

normál szöveg – pl *grep alma szoveg.txt* keresi az *alma* szöveget a szoveg.txt fájlban,
A reguláris kifejezésekben a következő speciális karaktereket használhatjuk:

. – egy darab tetszőleges karakter,
Pl. *grep a.lma szoveg.txt* megtalálja az *alma, a1ma, aama*, stb szövegeket
\t – tabulátor (TAB) karakter
\d - decimális számjegy,
\w – teljes szó
\s – betűköz, vagy bármely más *whitespace* szóelválasztó karakter (pl. TAB, föggőleges TAB)
[] karakter csoport. Pl. [a-z,1] nek megfelel bármely betű a és z között a-t és z-t is beleértve vagy az 1-es számjegy,
+ az előtte levő karakternek minimum egyszer szerepelnie kell, de szerepelhet akárhányszor,
pl. *grep "\s+alma"* akkor találja meg az *alma* karaktersorozatot, ha előtte minimum 1, de bármennyi whitespace karakter van.
* az előtte levő karakter szerepelhet akárhányszor (akár egyszer sem),
pl. *grep "\s*alma"* azt jelenti, hogy az *alma* szöveg elött vagy nincs, vagy akármennyi whitespace karakter van.
? – a legrövidebb előtte levő konstrukció (pl. []) ,
^ - sor eleje, de ha az első karakter egy []-ben akkor negálja a kifejezést,
$ sor vége, () –ami ebben van az egy külön kifejezés, ami bármi lehet.
Pl. *grep ^alma* csak a sor elején levő *alma*-t találja meg.

[9] Mi csak ezeket fogjuk használni, bár van még más is.

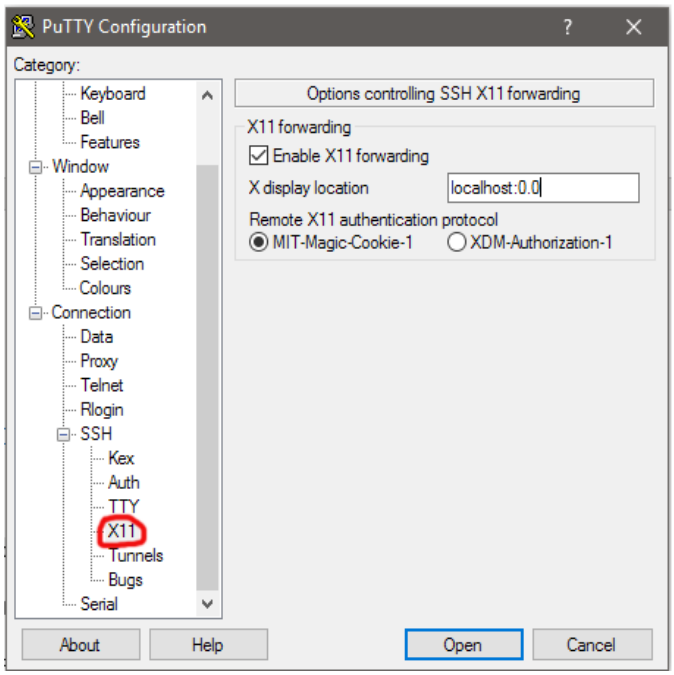
$ - sor vége

Pl. *grep alma$* csak a sor végén levő *alma*-t találja meg.

\ segítségével lehet pl a ()[]{}-jelekre keresni.

Grafikus programok készítése távoli Linux-on Windows alatt

Ha a Qt fejlesztő rendszer egy távoli Linux-os gépen fut és ezt Windows alól szeretnénk használni, akkor ehhez két programra van szükségünk. Az első a PUTTY, a második pedig egy ún. *X server*[10] pl. az az ingyenes Xming. Feltelepítése után először a PUTTY-ot konfiguráljuk fel[11]. Ha egyébként be tudtunk lépni a távoli gépre PUTTY-al, akkor töltsük be a távoli gép profilját, amelyben az alábbi beállításokat kell megadnunk:



Ezután mentsük el újra a profilt!

Az *Xlaunch* program indítása után konfigurálhatjuk fel az *Xming*-et.

A távoli gépről bármilyen grafikus programot (X kliens) elindíthatunk pont úgy, mint parancssori társát és a grafika a mi gépünk képernyőjén fog megjelenni. Célszerű a grafikus programokat *a háttérben indítani*, amihez a program nevéhez az & jelet kell hozzáírnunk. Pl. *./programom* helyett *./programom&* -et használunk. Az előbbi lefoglalja a terminált és, amíg be nem fejeződik azt másra nem tudjuk használni, az utóbbi esetben a konzolt használhatjuk tovább, miközben a programunk a hátérben fut.

[10] Az X Window egy grafikus felhasználói felületet (GUI) ad egy távoli gépen. Eredetileg arra találták ki, hogy a grafikus kimenetű programok (*X kliensek*) számára csak egy központi szerver gépet kelljen venni, és a grafika egy sokkal kisebb teljesítményű *X Terminálon* (*X szerver*) jelenik meg, amit hálózaton keresztül kapcsoltak a szerver géphez. Linux alatt a kliensek és a szerver sokszor ugyanazon a gépen fut.

[11] Az Xming maga is tartalmaz egy PUTTY verziót.