Kóródi Mihály

GTK+ programozás Free Pascalban

Kóródi Mihály

GTK+ programozás Free Pascalban

BBS-INFO, 2009.

Szerző: Kóródi Mihály

A könyv nagyobb mennyiségben megrendelhető a kiadónál:
BBS-INFO Kft., 1630 Bp. Pf. 21. Tel.: 407-17-07

ISBN 978-963-9425-36-1

Kiadja a BBS-INFO Kft.
1630 Budapest, Pf. 21.
Felelős kiadó: a BBS-INFO Kft. ügyvezetője
Nyomdai munkák: Bíró family nyomda
Felelős vezető: Bíró Krisztián

Tartalomjegyzék

Előszó

Ez a könyv a GTK+ grafikus eszköztár Free Pascal alatti felhasználásával foglalkozik. Bár a GTK+ tulajdonképpen csak egy alkalmazásprogramozási felület (API), a Free Pascal pedig egy teljes körű fejlesztési platform, a két projekt nagyon jól összekapcsolható egymással. Mindegyikük olyan ingyenes (nyílt forráskódú) eszközt ad a programozó kezébe, amely egyrészt jól átlátható és könnyen továbbfejleszthető forráskódot eredményez, másrészt biztosítja a legelterjedtebb PC-s operációs rendszerek alatti felhasználhatóságot.

A programozás alapjait még ma is gyakran Pascal nyelven oktatják, illetve tanulják, elsősorban annak nagyon jó áttekinthetősége miatt. Szerencsére a nyelv fennmaradását és további fejlődését a Free Pascal jelenléte is segíti, amely egy jelenleg is folyamatos fejlesztés alatt álló, modern fejlesztői környezet. Miután az eredetileg C nyelvhez készült GTK+ eszköztár már a Free Pascalos implementáción keresztül is elérhetővé vált, jelentős segítséghez jutottak mindazok, akik ezen a nyelven szeretnének grafikus felhasználói felülettel rendelkező alkalmazásokat készíteni.

A GTK+ is egy folyamatosan karbantartott és bővített projekt. Legutóbbi nagy verzióváltása (1.2-ről 2.0-ra) jelentős változásokat hozott az eszköztár tartalmában. Néhány elavultnak megítélt utasítást teljesen újak váltottak fel, valamint további szolgáltatások jelentek meg. A régi és az új széria közötti kisebb inkompatibilitás miatt az utóbbit gyakran GTK2-ként is említik, ám mivel a hivatalos megnevezés továbbra is a GTK+ maradt, ragaszkodjunk mi is ehhez.

A könyv elsősorban a grafikus felhasználói felületen futó ablakok munkaterületének kialakítására és az ott elhelyezett objektumok gyakorlati alkalmazására fekteti a hangsúlyt. Az anyagot ne tekintsük teljes GTK+ ismertetőnek, mivel annak megvalósításához talán a dupla ekkora terjedelem sem lenne elegendő, ennek ellenére nem csupán az

elindulás során, hanem akár összetettebb alkalmazások készítésekor is hasznosnak bizonyulhat ez a könyv.

Az anyag az általános Pascal programozási ismereteket nem tartalmazza, mivel ebben a témában már tetemes szakirodalom érhető el így is. A Pascal-szintaktika ismerete, a változótípusok, az alapvető vezérlési szerkezetek (elágazások és ciklusok), a tömbök és a rekordok kezelése, a mutatók használata stb. mind-mind annak a tudásanyagnak a része, amely a leírtak megértéséhez javallott. Szükséges továbbá az operációs rendszer (Windows és/vagy Linux) magabiztos használatának képessége (telepítés, fájl- és könyvtárkezelés, szövegfájlok szerkesztése stb.). Ezeket az alapismereteket kiegészítve, a bevezetés után rögtön a programozási környezetünk, a munkamenet kialakítására térünk rá, amely segíthet átlendülni a kezdeti nehézségeken.

A könyvben található forráskódok esetében – ráirányítva a figyelmet az új ismereteket tartalmazó részekre – legtöbbször csak program-részletekkel fogunk találkozni, így a teljes programlista helyett a már tárgyalt részek sorait kipontozva láthatjuk. A platformfüggetlenség jegyében a programok mindegyike működőképes kódot fog eredményezni Linux és Windows esetében egyaránt. Ahol mégis bármiféle különbség lenne, az átjárhatóság megvalósítása is megemlítésre kerül. A könyvben a GTK+ alkalmazások képernyőképei Linux alatt készültek, de a Windowsos látvány is szinte teljesen megegyezik.

1. Bevezetés

A Pascal programozási nyelvet és az első Pascal nyelvű fordítóprogramot a svájci *Niklaus Wirth* alkotta meg 1970-ben. Wirth a nyelvet *Blaise Pascal*ról, a XII. század neves francia matematikus-filozófus tudósáról nevezte el. A Pascal nyelvet 1973-ban szabványosították, amit 1982-ben átdolgoztak a ma is érvényben lévő Pascal szabvánnyá (ISO 7185). Az ezt a nyelvet megvalósító legnépszerűbb fejlesztői környezet a Borland cég DOS alatt használható *Turbo Pascal*ja volt. Nagy fejlődésen ment át és egészen a 7.0-s verzióig jutott el (1983-1991). A Turbo Pascal termékvonalának megszűnésével jelent meg a Borland Pascal 1992-ben, mely még néhány újítást tartalmazott. Már a Turbo Pascal 5.5-ös verziójában megjelent az objektumok használatának lehetősége, mely alapjául szolgált az *Object Pascal* nyelv megjelenésének, amit azóta már önálló programozási nyelvként tartanak számon. A Windows széleskörű elterjedésével a Borland Pascalt is fokozatosan leváltotta az 1995-ben megjelenő *Delphi*, mely egy Object Pascal nyelvre épülő, Windows alatt futó vizuális fejlesztőeszköz.

A *Free Pascal* kidolgozását a német *Florian Klämpfl* kezdte meg, hogy egy olyan Pascal fordítót készítsen, ami működőképes Linux operációs rendszer alatt is, teljesen 32 bites kódot generálva (azóta – természetesen – már 64 bites verzió is létezik). A projekt az FPK (*Free Pascal Kompiler*) nevet kapta. A fordító 1996-ban vált az internetes közösség számára is elérhetővé és hamarosan nagy népszerűségre tett szert. További programozók csatlakoztak a projekthez és az új cél már egy ingyenes és platformfüggetlen Pascal fordító megalkotása lett, ami szinte teljesen kompatibilis a Turbo/Borland Pascallal, illetve a Delphivel. Ma már többféle processzorarchitektúrára (x86, x86_64, PowerPC stb.) és több operációs rendszerre (Linux, FreeBSD, Mac OS, DOS, Windows stb.) egyaránt elérhető. Forráskódja is bárki által hozzáférhető, terjeszthető, a GNU licence (*GNU General Public Licence – GNU GPL*) alá tartozik. A

nyílt forráskód egyik óriási előnye, hogy szabad betekintést kapunk a program bármely részébe, ezáltal megtudhatjuk, hogyan készítették el azt a fejlesztők. Ez a Free Pascal esetében elsősorban a modulok (vagy *unitok*) forráskódjának böngészésekor van nagy hasznunkra, hiszen bármiféle dokumentáció igénybevétele nélkül is megismerhetjük az implementált eljárásokat, függvényeket és adattípusokat.

Az utóbbi években rengeteg kiegészítővel bővült a Free Pascal csomag, ilyen például a *GTK+* grafikus eszköztárak (*gtk1*, *gtk2*) felhasználását megvalósító unitok kidolgozása.

A GTK+ alapvetően egy alkalmazásprogramozási felület (*Application Programming Interface*, röviden *API*). Úgy használható, mintha egy építőkockákat tartalmazó ládika lenne, amely különböző méretű, alakú és rendeltetésű építőelemeket tartalmaz. Ezt az általános megvilágítást konkrétabbá alakítva: a GTK+ a grafikus operációs rendszereknél megszokott elemek (ablak, felirat, nyomógomb stb.) gyűjteménye, amelybe az ezen elemek megfelelő működtetésének biztosítása is beletartozik.

A GTK+-t eredetileg a GIMP (*GNU Image Manipulation Program*) képszerkesztő program fejlesztésére készítették és a nevét is innen kapta (*GIMP Toolkit*). A plusz jelet később adták hozzá a névhez, amikor az eszköztár objektumorientált jellemzőket és bővíthetőséget kapott. A projektet a kaliforniai Berkeley egyetem két hallgatója, *Spencer Kimball* és *Peter Mattis* indította el 1995-ben, melyhez később egy csoport önkéntes is csatlakozott – továbbra is a *GNU Project* pártfogása alatt –, ezzel is növelve a fejlesztés hatékonyságát. A GTK+ rövidesen már egy nyílt forrású ablakkezelő rendszer, a GNOME alapjául is szolgált, amely egy könnyen kezelhető, letisztult és barátságos grafikus felhasználói felületet (*Graphical User Interface – GUI*) nyújtó ablakkezelő, elsősorban a Linux felhasználóinak.

A GTK+ nagy népszerűségnek örvend a nyílt forráskódban fejlesztők körében és ma már nem csak a Unix-alapú operációs rendszereken, hanem Windows alá is elérhető. A GNU függvénykönyvtárakra vonatkozó licence alá tartozik (*GNU Library General Public License – GNU LGPL*), ami rugalmas licencelést tesz lehetővé a kliens alkalmazások számára. Bármit közzé tehetünk tehát, csak meg kell említenünk, hogy a program a GNU LGPL alá tartozó függvénytárral készült, de akár azt is lehetővé teszi, hogy a mű, vagy annak bármely része rögtön GPL licenc alatt kerüljön kiadásra.

Maga az eszköztár C nyelven íródott és az általa, mint grafikus környezet által nyújtott szolgáltatásokat – a *gtk.h* fejállományon keresztül – is csak C programozási nyelven érhettük el kezdetben. Népszerűsége miatt azonban hamar más, a C-hez közel álló nyelvek (C++, C#, Python, PHP stb.) mellett már Pascal alatt is felhasználhatóvá vált. Az átiratnak köszönhetően így már Pascal nyelven is készíthetünk grafikus felhasználói felülettel rendelkező, többplatformos alkalmazásokat.

2. A munkamenet

A programozásnak csak akkor ugorhatunk neki, ha előkészítettük a munkamenetet és gondosan kialakítottuk a számunkra kényelmes felhasználói környezetet. Elsőként töltsük le a *http://freepascal.org* címről (vagy az *ftp.freepascal.org/pub*-ról) a bármelyik platformra elérhető Free Pascal csomag legújabb stabil vagy akár snapshot (a fejlesztés pillanatnyi állapota) verzióját.

Linux alá a „minden az egyben" *tar.gz* tömörített formátumon kívül letölthetünk *rpm* vagy *deb* telepítőcsomagot is, melyek a telepítés során minden állományt a szabványos könyvtárakba (*bin*, *lib*, *share*) helyeznek el. A Linuxos változat esetében a GNU Assembler (*as*) és a GNU Linker (*ld*) jelenlétére is szükségünk lesz, ám ezeket minden mai disztribúció tartalmazza, így ellenőrzésük szükségtelen.

A Windowsos változat esetében az aktuális stabil verziók csak telepítő formájában tölthetőek le. Tömörített, *zip* csomagot jellemzően vagy csak a régebbi, vagy a jelenlegi stabil változatnál is naprakészebb, fejlesztői változatok esetében érhetünk el. A telepítő előnye, hogy szükség esetén beállíthatjuk az automatikus fájltársításokat a *pas* és *pp* állományokhoz, valamint elkészíttethetjük vele az alapértelmezett konfigurációs fájlokat. A telepítés során ügyeljünk arra, hogy a Free Pascal egy szóközök és ékezetek nélküli könyvtárba kerüljön (például *C:\FPC*).

2.1 Az IDE használata

A Free Pascalhoz hivatalos fejlesztői környezet (*Integrated Development Environment – IDE*) is tartozik, mely tökéletesen hasonlít a régi idők népszerű, a DOS karakteres felületén futó, egérrel is vezérelhető Turbo Pascal IDE-jéhez. Ahogy annak idején a Borland is lehetővé tette a TP IDE felületéhez hasonló programok elkészítését a Turbo Vision keretrendszer kiadásával, úgy a Free Pascal IDE-jét fejlesztő csapat is elkészítette saját Free Vision keretrendszerét, mely elődjével szinte tökéletesen kompatibilis. A fejlesztői környezet DOS (Go32v2), Windows

és Linux alatt is használható, az ezekre a platformokra letöltött Free Pascal alapcsomag tartozékát képezi.

Az IDE a megnyitott forráskód fordítása során külső programként hívja meg a *ppc386* parancssori fordítót. A fejlesztői környezet teljes körű szolgáltatást nyújt a programozónak: menüvezérelt, Pascal szintaxis-kiemeléssel ellátott szerkesztője van, akár több forráskód egyidejű megnyitását is lehetővé teszi több szerkesztő ablakban, a forráskódot lefordítja, illetve futtatja is, segíti a hibakeresést stb. A kezelését viszont szokni kell, és eleinte igen kényelmetlen lehet a használata, mivel mégis csak egy karakteres módú munkakörnyezetet kell használnunk.

Az IDE Windows alatti futtatásához (*fp.exe*) szükség van a *cygwin1.dll* fájl jelenlétére is. Abban az esetben, ha az alapcsomag *bin* könyvtára nem tartalmazná ezt a fájlt, külön kell letöltenünk azt (http://cygwin.com).

A program első indításakor az *Options* menü ⌾ *Directories* menüpontban be kell állítani a munkakönyvtárakat. A legfontosabb ilyen könyvtárak a *Units* fülön a unitok könyvtárai, valamint megadhatunk még a *Misc*. fülön az *EXE* és *PPU output directory*-nak egyaránt egy külön könyvtárat a lefordított bináris állományok számára. Fontos tudnivaló, hogy az IDE bármelyik platformú változatában a „/" helyett a „\" jel is használható a könyvtárak elválasztására. A beállítások megőrzésére a program automatikusan egy *fp.cfg* fájlt generál, mely ezután bármilyen szövegszerkesztővel (*plain text editor*) szerkeszthető.

Mivel az IDE-t a lehető legalaposabb Turbo Pascal fejlesztői környezetre való hasonlatosság jegyében tervezték, a használata is megegyezik azzal: a mentés itt is az F2-vel (*File* menü ⌁ *Save*), a fordítás és futtatás a Ctrl+F9-cel (*Run* menü ⌁ *Run*) történik stb.

2.2 Egyedi munkakörnyezet kialakítása

Mivel a teljes Free Pascal csomagra biztosan nem lesz szükségünk, a legszükségesebb összetevők kimásolásával létrehozhatunk egy teljesen egyedi könyvtárstruktúrát is egy saját könyvtárba. A csak legszükségesebb fájlok megtartásával fejlesztői környezetünket sokkal hordozhatóbbá és áttekinthetőbbé tehetjük. Ez Linux és Windows alatt egyaránt megvalósítható. Egy ilyen egyedi összeállítás elkészítésének menetét fogjuk most megismerni.

2.2.1 A könyvtárszerkezet

A Free Pascal csomag legfontosabb állománya a *ppc386 (*ennek 64 bites változata a *ppcx64)*, ami a tulajdonképpeni Free Pascal fordító (*compiler*), vagyis az a program, ami a Pascal forráskódunk bináris, gépi kóddá való átalakítását végzi. A sikeresen lefordítható Pascal kódból először egy assembly kódot generál, amiből egy külső assembler és linker segítségével készíti el az object-állományt, végül a kész, futtatható programot.

A fordításhoz a fordítóprogram számára a forráskód által igényelt modulokat (unitokat) is elérhetővé kell tennünk. Hozzuk létre tehát az alábbi listának megfelelő, egyszerű könyvtárszerkezetet (az itt látható *2.3.1* egy éppen aktuális *snapshot* verzió száma):

```
fpc-2.3.1
  ├──bin
  ├──lib
  │   └──ppu
  │       ├──gtk2
  │       ├──rtl
  │       ...
  └──src
```

Az itt létrehozott *bin* könyvtárban minimálisan a *ppc386* állományt kell elhelyeznünk. A fordító önmagában is teljesen alkalmas bármilyen forráskód bináris állománnyá fordítására. Hátránya viszont, hogy ebben az esetben minden fordítást a forráskód teljes elérési útjának és további kötelező paraméterek (leginkább a felhasznált unitok) megadásával kell elvégeznünk. Ennek elkerülésére a Free Pascal csomag kétféle megoldást kínál: az IDE (*fp* illetve *fp.exe*) és a parancssori fordító közvetlen (*ppc386* illetve *ppc386.exe*) vagy közvetett (*fpc* illetve *fpc.exe*) használatát. Jobb, ha mindegyiket bemásoljuk a *bin*-be, mert bármelyik mellett döntünk is, előfordulhat, hogy szükségünk lesz a másik megoldásra. Ne feledkezzünk meg arról sem, hogy az *fp* Windowsos változata a *cygwin1.dll* jelenlétét is igényli. A *bin* mappa ajánlott minimális tartalma tehát Linux és Windows (zárójelben) alatt a következő:

```
bin
  ├──(cygwin1.dll)
  ├──fp (.exe)
  ├──fpc (.exe)
  └──ppc386 (.exe)
```

A *ppu* könyvtárba mindenképpen másoljuk át az RTL és a GTK2 unitcsomagokat (egyelőre többre nem is lesz szükség). Az RTL (*Run-Time Library – Futásidejű Könyvtár*) látja el a legalapvetőbb I/O feladatokat, azaz a memória- és fájlkezelést, a konzolon történő kapcsolattartást a felhasználóval, a dátum és idő kezelését stb. Fontos, hogy a unitok másolásakor ne csak ömlesztve kerüljenek át a fájlok, hanem az őket tartalmazó *rtl* és *gtk2* nevű szülőkönyvtárakkal együtt, így az összetartozó unitok a későbbiek során is elkülöníthetők lesznek egymástól.

2.2.2 A fordítás

A Pascal forráskód futtatható állománnyá való fordítását a *ppc386* (vagy *ppcx64*) *program végzi*. Mivel a *ppc386* fordító nem tudhatja, milyen könyvtárstruktúrát alakított ki a felhasználó, a forrásállományon kívül minden, a forráskód által kért unit elérési útját is meg kell adnunk a *ppc386* paraméterezésével. Ennek módját és a többi használható fordítási lehetőséget a *ppc386* paraméterek nélküli futtatásával nézhetjük meg. A hosszú listában többek között látható, hogy a *–Fu* beírása után adhatunk meg unit elérési utat, a *–FE* paraméter adja meg az elkészült bináris helyét és így tovább. Ez alapján megállapíthatjuk, hogy Linux alatt, a *ppc386* könyvtárában (*bin*) kiadott *./ppc386 ../src/proba.pas –Fu../lib/ppu/rtl* parancs sikeresen lefordítja az *src* könyvtárban elhelyezkedő *proba.pas*-t, amennyiben az csak az RTL csomag unitjait használja. Ugyanennek az utasítássornak Windows alatti változata: *ppc386 ..\src\proba.pas –Fu..\lib\ppu\rtl*.

A Pascal kód fordítása során a fordító először egy átmeneti assembly forráskódot, majd ebből egy *.o* kiterjesztésű *object* (tárgy) állományt készít. Ezeket összefűzve hozza létre a futtatható állományt vagy modult. Futtatható állomány esetében ezután az object állomány törölhető, unit esetében viszont meg kell tartanunk, ugyanis a további, e modult használó programjaink fordításához nélkülözhetetlen.

Fordításhoz a *ppc386* helyett az *fpc* nevű programot is igénybe vehetjük. Hogy használható legyen, készítenünk kell mellé egy *fpc.cfg* nevű konfigurációs fájlt is. Az *fpc* rendkívül egyszerűen működik, mindössze egyetlen paramétert vár a felhasználótól: a Pascal forrásállomány nevét, illetve elérési útját. Akár létezik ez a fájl, akár nem,

összeállít egy paraméterezett utasítássort, melyben a *ppc386* fordítót hívja meg a forrásállomány nevével és az *fpc.cfg* szövegfájl teljes tartalmával egybefűzve.

Az *fpc.cfg* tartalmára láthatunk egy egyszerű példát is, amely feltételezi, hogy betartottuk az előzőekben vázolt könyvtárszerkezet felépítését:

```
# fpc fordítási paraméterek

-l
-viwn

-Fu../lib/ppu/gtk2
-Fu../lib/ppu/rtl

-FE../out
```

A `-Fu` és `-FE` kezdetű sorok szerepét már ismerjük. A `-l` csak tartalmasabbá teszi a fordítás menetét, ugyanis ennek bekapcsolásával a Free Pascal logó is kiíródik a fordítás megkezdésekor. A `-viwn` ennél már fontosabb szerepet tölt be, ugyanis bekapcsolásakor minden fontos fordítási információt láthatunk.

Miután elkészítettük a konfigurációs állományunkat, már csak a forrásállomány nevét és elérési útját kell megadnunk az *fpc*-nek. Még így is sokat kell gépelni, ezért készíthetünk egy szkriptet a forrásállomány könyvtárába (*src*), hogy annak futtatása indítsa el a fordítási folyamatot a benne megadott fájlnévvel. Ez a szkript Linux alatt a Bash parancsértelmező által kínált bőséges szolgáltatáslistának köszönhetően egészen összetett is lehet.

Egy kis Linux héjprogramozás következik. Hozzuk létre az alábbi fájlt, melynek az egyszerűség kedvéért adjuk a *run* nevet, majd állítsuk át az attribútumát futtathatóvá:

```
#!/bin/sh

SOURCE="forrasfilenev"
cd ../bin
if [ -f ../out/$SOURCE ];
  then rm -f ../out/$SOURCE;
fi
if [ -f ../src/$SOURCE.pas ];
  then ./fpc ../src/$SOURCE.pas;
  else echo "Nincs meg a $SOURCE.pas forrasfile!";
fi
if [ -f ../out/$SOURCE ];
  then ../out/$SOURCE;
  else echo "Nem keszult el a ../out/$SOURCE file!";
fi
```

Az első részben kell megadni a forrásállomány nevét az idézőjelek között ("`forrasfilenev`"), ami például egy *proba.pas* nevű fájl esetében "`proba`". Ezután a héjprogram átlép a *../bin* könyvtárba, ahol az *fpc* található. Ehhez a könyvtárhoz képest figyeli, nem létezik-e már egy korábban lefordított példány, amit ki lehet törölni. Miután ezen is túllépett megvizsgálja, hogy a forrásfájl létezik-e. Ha igen, lefordítja, ha nem, közli annak hiányát. Végül, ha sikeres volt a fordítás, az elkészült binárisnak ott kell lennie az *out* könyvtárban, amit el is indít.

Láthatjuk, hogy mindenképpen érdemes időt szentelni a fejlesztési környezetünk kialakítására – még ha ezt nem is az eddigiekben részletezett módon tesszük meg –, mivel ez nagyban megkönnyítheti későbbi munkánkat. Az IDE helyett használhatunk külső szövegszerkesztőt is, ugyanis rengeteg ingyenes, Pascal szintaxiskiemeléssel rendelkező editor létezik, melyek között akadnak nagyon jó képességűek is, például a *gedit*, a *notepad++* vagy a *geany*. Ezek közül a *gedit* és a *geany* is GTK+-ban fejlesztett, Windowsra és Linuxra egyaránt letölthető alkalmazások.

2.2.3 Segédprogramok

A Free Pascal csomag a fordításban közvetlenül résztvevő programokon és az alapvető futásidejű könyvtáron (*Run-Time Library*) kívül még néhány segédprogramot és rengeteg további unitot is tartalmaz. A unitok felhasználását több hasznos demo és példaprogram mutatja be, így azok szerepe egyértelmű.

A segédprogramok ezzel szemben a fordítástól teljesen függetlenek, hiányuk még nem teszi lehetetlenné a fejlesztést. A csomag *bin* könyvtárában számos segédprogramot találunk, ezek közül vegyük sorra a fontosabbakat:

fpcmake Beolvassa a neki megadott helyen található *Makefile.fpc* fájlt és egy *Makefile* nevű állományt próbál generálni a *make* nevű segédprogram számára. A *make* jelentősen felgyorsítja a fordítást, ha egy sokszorosan összetett, sok állományból álló forráskódról van szó. A használható kapcsolókat és opciókat az *fpcmake --help* parancs kiadásával írathatjuk ki.

ppudump Megmutatja a paraméterként megadott unit tartalmát. Egyszerű, kapcsolók nélküli kiadásakor minden információt láthatunk, amit ajánlott megszűrni a felesleges részletektől. A *ppudump -Vh unitnév.ppu* csak az összesített fejléc információt, a *ppudump -Vi unitnév.ppu* csak az interface információkat jeleníti meg.
Az összes használható kapcsoló listáját a *ppudump --help* parancs kiadásával írathatjuk ki.

ppumove Osztott vagy statikus függvénykönyvtárakat készíthetünk unitok felhasználásával. A függvénykönyvtárak kiterjesztése *.a* lesz statikus állományoknál, osztottaknál pedig *.so* Linux alatt illetve *.dll* Windows és OS/2 alatt. Bővebb információkat a programról a Free Pascal dokumentációjában találhatunk (*doc/user.pdf*).

ptop Forráskód-szépítő program. Elegendő két fájlnevet megadnunk neki, ahol az első a bemeneti, a második a kimeneti forrásállomány neve. A kimeneti, új fájl egy jobban áttekinthető forráskódot fog tartalmazni. Kivéve persze, ha már az eredeti forráskódunk is a saját ízlésünknek megfelelő. A programhoz konfigurációs állományt (*ptop.cfg*) is készíthetünk, melynek használatát a *-c* kapcsolóval kényszeríthetjük ki.

strip A kész futtatható program méretének csökkentését végzi. Erre azért van lehetőségünk, mert a fordító rengeteg olyan adattal is megtölti programunkat, melyekre annak soha nem is lesz szüksége. A *strip* használata nagyon egyszerű: *strip fájlnév*. A fájl ezután akár több mint 30%-al kisebb méretű lesz. Tömör állományt már fordításkor is készíthetünk a *-Xs* kapcsoló használatával.

Egyedi munkakörnyezetünk *bin* könyvtárát igény szerint kibővíthetjük a segédprogramokkal, teljesebb értékűvé téve azt. A fentiek mellett a teljes Free Pascal csomag még számos további programot kínál fel.

2.3 A GTK+ telepítése

Ahhoz, hogy a GTK+ függvénytárat használó programjaink egyáltalán működőképesek legyenek, szükségünk van a függvénytárat megvalósító fájlok jelenlétére. Ezeket a fájlokat (Windows alatt *.dll*-eket, Linux alatt *.so*-kat) a GTK+ alkalmazás számára megtalálható helyre kell elhelyeznünk, amely vagy a lefordított alkalmazásunkat is tartalmazó könyvtár, vagy az adott operációs rendszer egy rendszerkönyvtára (vagyis, benne van a *„path"* környezeti változóban).

Linux esetében többnyire már a gyári telepítés is tartalmazza a GTK+-t, így a fájlok oda kerülnek, ahová valók (*/usr/lib*), Windows alatt viszont nekünk kell gondoskodnunk a telepítésről. Linux alatt – feltéve, hogy a függvénytár és annak minden szükséges kiegészítője problémamentesen feltelepült – szükségtelen a következőkben leírt mélységben beásnunk magunkat a GTK+ könyvtáraiba, alkalmazásaink azonnal indíthatóak.

2.3.1 Tennivalók Windows alatt

A mindenkori legújabb stabil verzió szabadon letölthető a *http://gtk.org* címről, ahol ajánlott a teljes csomagot tartalmazó *„bundle"* állományt választani (jelenleg ez a *gtk+-bundle_2.16.2-20090601_win32.zip*). Ennek az archívumnak a *bin* könyvtárában találhatjuk meg a nekünk szükséges, alapvető *dll*-eket, valamint néhány egyéb állományt. Ahhoz, hogy egyáltalán elinduljon majdani GTK+ alkalmazásunk, minimálisan a csomag *bin* könyvtárából az összes *dll* fájl kimásolására szükségünk van. Két lehetőségünk van arra, hogy ezeket a fájlokat hol helyezzük el:

1. Átmásoljuk a *dll*-eket az alkalmazást is tartalmazó könyvtárba vagy
2. használhatjuk a PATH környezeti változóban bent lévő egyik könyvtárat is, amilyen például a *Windows\System.*

A másolás után bizonyosodjunk meg róla, hogy működőképesek lesznek-e programjaink: a csomag *bin* könyvtárából a *gtk-demo.exe* fájlt bontsuk ki és próbáljuk elindítani. Az alapvető *dll*-ek megfelelő helyre másolását követően már képes elindulni egy GTK+ alkalmazás, de néhány további, a *bundle* csomagban megtalálható könyvtár és állomány átmásolása is ajánlott:

- A *lib*/*gtk-2.0*/*2.10.0* könyvtárszerkezet létrehozása és annak teljes tartalma. Itt a képformátum-betöltő és a különböző nyelvű beviteli módokat megvalósító függvénytárak találhatóak. Ezek mellett az *engines*-ben található *libwimp.dll* a Windows XP stílusú megjelenítést végzi el.
- Az *etc* könyvtárra és annak teljes tartalmára is szükség lehet. Az ezen belüli *gtk-2.0* könyvárnak a tartalmát ki kell egészítenünk a csomag *share*/*themes*/*MS-Windows*/*gtk-2.0* könyvtárának *gtkrc* fájljával, mely az előbb említett *libwimp.dll* használatát kapcsolja be, beállítva ezzel a megjelenítési témát „XP stílusúra". Az alapértelmezett beállítás szerint a nyomógombok nem tartalmazhatnak ikonokat. Ezt a *gtkrc* fájl szerkesztésével, a `gtk-button-images` paraméter `1` értékűre állításával tudjuk visszakapcsolni.
- A csomag *share* könyvtárának *locale* könyvtára a beépített angolon kívül további 98 nyelvet tartalmaz, így az itt lévő *hu* könyvtár megfelelő helyre történő átmásolása a GTK+ beépített szöveggel ellátott elemeinek (például „Save", „Print" gombok stb.) magyarosítását eredményezi.

Az elkészült GTK+ alkalmazásainkat bizonyára terjeszteni is szeretnénk majd. Egy másik, Windowst futtató számítógépre való telepítéskor a legjobb megoldás, ha alkalmazásunk mellé minden szükséges könyvtárat és állományt egyetlen, közös könyvtárba helyezünk. Ezen a könyvtáron belül legalább 4 mappa megléte ajánlott: *bin*, *etc*, *lib*, *share*. Az alkalmazásnak és a GTK+ elsődleges *dll* állományainak a *bin* mappában kell lennie. Bizonyosodjunk meg róla, hogy az alábbi könyvtárak és az egyes alkönyvtárak fájljai a helyükön vannak:

```
projektnév
  ├──bin
  ├──etc
  │   └──gtk-2.0
  ├──lib
  │   └──gtk-2.0
  │       ├──2.10.0
  │       │   ├──engines
  │       │   └──loaders
  │       └──modules
  └──share
      └──locale
          └──hu
              └──LC_MESSAGES
```

Az új gépre másoláskor ezek után a felhasználónak már semmi dolga nincs: mivel alkalmazásunkat a GTK+-val együtt mellékeltük, az minden bizonnyal el fog indulni.

A *GIMP 2* képszerkesztő program (1. ábra) korai verzióinak Windowsos változata esetében a GTK+ fájljainak rendszerkönyvtárba helyezéséről egy különálló telepítő gondoskodott, tehát teljesen független volt a programtól. A GIMP újabb verziói már a program fájljait tartalmazó könyvtáron belül tartalmazzák a GTK+-t. Az 1. ábrán látható képernyőképen egyébként szinte valamennyi, az elkövetkezőkben bemutatandó látványelem-típus megtalálható.

1. ábra

2.4. Programírási tippek

A Pascal már önmagában is egy jól (sőt, az egyik legjobban) áttekinthető programozási nyelv. Ennek ellenére egy hanyagul összedobott forráskódban könnyen eltévedhetünk, nem beszélve arról, hogy ami már nekünk is nehezen átlátható, az más számára akár teljes

káoszt is jelenthet. A másik érzékeny pont a készülő alkalmazás felhasználói felületének megjelenése, amelynek esetében az egyik legfontosabb szempont a felhasználó eligazítása, komfortérzésének segítése.

A fejezetnek ez a szakasza igyekszik némi útmutatást adni, hogy melyek azok a tényezők, amelyeket érdemes betartani a Free Pascalban írt alkalmazások készítésénél. Ezeknek a tippeknek a betartása természetesen nem kötelező, mégis megeshet, hogy számunkra új ötleteket meríthetünk belőlük, amelyek esetenként még jobb programok megszületéséhez vezethet.

2.4.1 A forráskód

Mindig, minden körülmények között ügyeljünk a forráskód megfelelő tagoltságára! A programunkat célszerű úgy írni, hogy a későbbi fejlesztések során is könnyen átlátható legyen, akár más programozó számára is. A `begin` és `end` közötti programblokkok például két karakterrel beljebb kezdődhetnek, a ciklusmagok és elágazások újabb két karakterrel beljebb és így tovább.

A kis- és nagybetűs írásmódot az esetek többségében figyelmen kívül is hagyhatnánk, de itt is tanácsos egy egységes szabályrendszert kialakítani magunkban. A könyvben látható programrészekben is egy ilyen szabályrendszert fogunk viszont látni, amelynek megfelelően csak bizonyos kategóriákba tartozó részek lesznek nagybetűkkel feltüntetve.

Nagyobb alkalmazások készítésénél használjuk ki a forráskód részekre bontásának és önálló állományokban való tárolásának lehetőségét! A különálló fájlok használata jelentősen elősegítheti egy összetettebb alkalmazás átláthatóságát és további fejlesztését. Az egymással összefüggésben lévő alprogramokat és egyéb programrészeket (például a nagyobb tömbtípusú konstansokat) mindenképpen ajánlott modulokban vagy *include* állományokban tárolni. E kettő közül az utóbbi használata lényegesen egyszerűbb, például:

```
{$include programresz1.inc}
```

A forráskódban elhelyezhetünk megjegyzéseket is, ezzel is továbbsegítve magunk illetve mások tájékozódását. A Free Pascal

háromféle módot ajánl fel megjegyzés beszúrására, melyek mindegyikét teljesen figyelmen kívül hagyja a fordító, mintha ott sem lennének:

```
(* Ez egy régi stílusú megjegyzés *)

{ Ez egy Turbo Pascal szabványú megjegyzés }

// Ez egy Delphi-s megjegyzés; egy teljes sorra vonatkozik
```

2.4.2 A felhasználói felület

Törekedjünk a grafikus alkalmazásunk minél kifogástalanabb megjelenésére! A GTK+ eszköztárában megtalálhatjuk mindazokat a látványelemeket, amelyek a felhasználót beavatkozásra „csábítják". Ennek ellenére fontos, hogy mindig szem előtt tartsuk azt az egyensúlyt, hogy a túlságosan csinos megjelenés ne menjen a funkcionalitás rovására. Tanácsos mellőzni a túlzottan egyedi kinézet erőltetett kialakítását is, hiszen ettől szintúgy összezavarodhat a program felhasználója.

Léteznek alkalmazások, amelyek rengeteg információval árasztják el a felhasználót. Ezzel kapcsolatban sem árt az óvatosság, hiszen a tapasztaltabb felhasználókat sem szabad figyelmen kívül hagynunk, akiket éppen ezek az információk zavarhatnak a leginkább. Összességében talán az a legjobb megoldás, ha egy hétköznapi, átlagos szinten hozzáértő felhasználó helyébe képzeljük magunkat és ennek megfelelően alakítjuk ki a felhasználói felületet.

2.4.3 Dokumentációk keresése

A GTK+ alkalmazásunk fejlesztése során több helyre is fordulhatunk segítségért, ha elakadnánk, vagy újabb lehetőségek után kutatnánk. A C nyelvű GTK+ programozást segítő hivatalos, mindenkori legfrissebb HTML alapú dokumentáció (*GTK+ Reference Manual*) megtalálható az interneten a *http://library.gnome.org/devel/gtk/* címen, de a teljes GTK+ csomaggal együtt letöltve, offline is böngészhető. Annak ellenére, hogy esetünkben nagyobb segítséget jelentene egy Pascal nyelvű dokumentáció, rendkívül jól áttekinthető magyarázatokat találhatunk benne, elsősorban a különféle függvények megismeréséhez.

A másik hasznos információforrás nem más, mint a Free Pascal szabadon letölthető forráskódja. Ebben pontosan végigkövethetjük