© Kiskapu Kft. Minden jog fenntartva

Programhonosítás a gettext csomaggal

Azt mondják, hogy a Microsoft annak idején azzal hódította meg a piacot, hogy a Windows 3.1-et a világ szinte minden nyelvén hozzáférhetővé tette. Ezáltal a számítástechnika széles rétegek számára lett elérhető.

A Linux terjedését gátló egyik fő ok régebben ugyanez volt. Nem várható el, hogy a gépet csak egyszerűen használni kívánó felhasználók az angol nyelvet is ismerjék. Sőt a számítástechnika szaknyelvében rengeteg olyan kifejezés akad, amit még az angolul egyébként jól tudóknak is külön meg kell tanulni. De a nemzetközi irodalmi művek is csak akkor válhattak egy nyelvi kultúra részévé, ha az adott nyelven is elérhetők lettek (gondoljunk csak a Biblia-fordításokra vagy Shakespeare műveire).

Igaz, az internet térhódításával néhány éven belül eljuthatunk oda, hogy az angol nyelvtudás olyan természetes lesz, mint a jogosítvány, mégis, egészen más érzés az anyanyelvünkön értő programmal dolgozni. Ezt a kereskedelmi programok készítői is pontosan tudják.

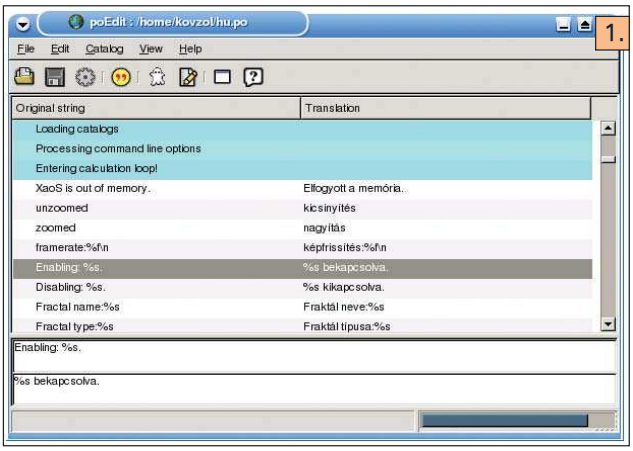
Az elmúlt években több nagy fajsúlyú program (KDE, Gnome, OpenOffice, Mozilla) magyarítása is nagy nyilvánosságot kapott. Bár Magyarország kis ország, és a honosítás előnyeit hasznosító felhasználók száma aránylag alacsony, e programok magyarra fordítása nagy előrelépés a Linux elterjedésében.

A honosítás olyan munka, amit – a programozói tudással ellentétben – csupán felhasználói ismeretekkel is el lehet kezdeni, és ez tovább növeli a Linux fejlesztőinek táborát. S mivel többen magukénak érzik a honosított programot, még többen vannak, akik a barátaikat is meg akarják győzni arról, hogy „ez a jobb, ezt használd!”. Izgalmas (s megmosolyogtató) látni azt a lelkesedést, amellyel a sokszor avatatlan újságírók a Linuxról írnak – önmagában az a tény, hogy az UHU Linux százszázalékosan magyar fejlesztés, olyasmi, amire egyszerűen büszkék lehetünk, és végre Magyarország is kihúzhatja magát, hogy van saját fejlesztésű operációs rendszere.

A lelkesedést a megfelelő mederben tartva valóban minőségi honosítások jöhetnek létre. A ➲ http://forditas.fsf.hu/html/Utmutato.html címen részletes nyelvi útmutatót olvashatunk ***Koblinger Egmont*** és ***Tímár András*** jóvoltából. Hasznosnak és fontosnak tartom a Linuxvilág Szótár rovatát is, amely szintén a nyelvi egységesítés felé mutat jó irányt.

Ha eldöntöttük, hogy csatlakozunk valamelyik program fordításához, akkor a legegyszerűbb felvenni a kapcsolatot a fejlesztői csapattal. Előfordulhat, hogy a programhoz még nincs magyar fordítás, de az is lehet, hogy már létezik, csak a fordító esetleg már nem foglalkozik a karbantartásával. Mindez az olvasó számára azt sejteti, hogy a fordító munkája akár hosszú évekig is elnyúlhat, amint a program új szövegekkel gyarapszik, esetleg bizonyos szövegrészek módosulnak benne.

Némelyik program fejlesztői erős iramot diktálnak komolyan betartandó határidőkkel (ami természetesen a fordítóra is vonatkozhat), más programoknál a lendület kisebb, és a fordítás akár évekig is elhúzódik. Azt mindenesetre tudnunk kell, hogy a programozást és a fordítást is többnyire önkéntesek végzik, szabadidőben; ez azonban nem ad felmentést az alól, hogy amennyiben egy fordítási munkát elvállalunk, azt



1.

2.

felelősséggel végezzük el. A nagyobb programoknál több fordítóra is szükség lehet a lefordítandó szövegek nagy száma miatt, így számítsunk arra, hogy csapatmunka várhat ránk. Bizonyos programok lefordításához komoly szakmai tudásra is szükség lehet. Készüljünk fel arra, hogy szinte minden programban akadnak olyan kifejezések, amelyeket csak akkor fogunk tudni lefordítani, ha a programot angol eredetijében már teljes egészében megismertük – csak ezután érdemes a munkának teljes erőbedobással nekivágni.

Technikai részletek

A programok nemzetköziesítésére több módszer is kínálkozik. Úgy tűnik, hogy a legéletképesebb és legelterjedtebb kezdeményezés a GNU gettext projekt, amely jelenleg a 0.12.1-es változatnál tart. A ➲ http://www.kde.org/fordfaq/forditas.html és a ➲ http://www.szabilinux.hu/forditasok/gnome-faq/i18n.html címeken elindulva jó alapokra tehetünk szert a témában, magyar nyelven, röviden. (A gettext hosszú, alapos leírása talán sokakat elriaszt ettől az egyébként nagyon kényelmesen kezelhető eszköztől.)

A gettext alapú fordítási folyamat lényege a következő: egy úgynevezett sablonfájlban (pl.: ***messages.pot***) a program szerzői az összes lefordítandó szöveget felsorolják angol nyelven. Egy ilyen szöveg lehet egy-egy menü neve, esetleg egy rövid felirat, vagy akár egy hosszabb, több mondaton is átnyúló magyarázó leírás. A sablonfájlt lemásolva a fordító létrehozza a majdani saját nyelvű fordításokat tartalmazó fájlt (például ***hu.po*** néven), egy-két perc alatt módosítja a fájl legelején található leíró fejlécet, majd az egyes angol szövegek alá begépeli a magyar nyelvű megfelelőt. Ezután a szerzőknek a ***hu.po*** fájlt elküldve a programba már könnyen befűzhetők a magyar nyelvű üzenetek (a befűzést a fejlesztők rendszerint maguk elvégzik). Nézzünk egy példát! A XaoS fraktálrajzoló program 3.1-es változatában a forrásprogram ***src/i18n*** könyvtárában található ***hu.po*** fájl így indul:

```
# XaoS NLS file for Hungarian language.
# Copyright (C) 2002 Free Software Foundation,
# Inc.
# Zoltan Kovacs <kovzol@math.u-szeged.hu>, 2002.
#
msgid ""
msgstr ""
"Project-Id-Version: XaoS 3.1\n"
"POT-Creation-Date: 2002-09-25 21:17+0200\n"
"PO-Revision-Date: 2002-08-17 21:44+0200\n"
"Last-Translator: Zoltan Kovacs
➥<kovzol@math.u-szeged.hu>\n"
"Language-Team: Hungarian\n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain;
➥charset=ISO-8859-2\n"
"Content-Transfer-Encoding: 8-bit\n"

#: ../ui/ui.c:310
msgid "XaoS is out of memory."
msgstr "Elfogyott a memória."

#: ../ui/ui.c:360
msgid "%s %.2f times (%.1fE) %2.2f
➥frames/sec %c %i %i %i %i                "
msgstr ""
"%2$.2f-szeres %1$s (%3$.1fE) %4$2.2f kép/mp"
" %5$c %6$i %7$i %8$i %9H$i                "

#: ../ui/ui.c:361
msgid "unzoomed"
msgstr "kicsinyítés"

#: ../ui/ui.c:361
msgid "zoomed"
msgstr "nagyítás"

#: ../ui/ui.c:368
#, c-format
msgid "framerate:%f\n"
msgstr "képfrissítés:%f\n"
```

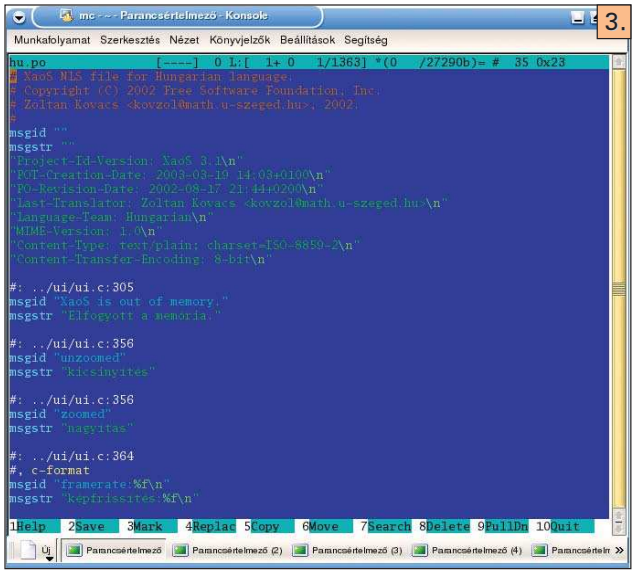
A # (kettős kereszttel) kezdődő sorok megjegyzések, tartalmuk másodlagos. Az angol nyelvű szövegeket az `msgid` kezdetű sorokban olvassuk, idézőjelek között. (Amennyiben a szöveg egy sornál hosszabb, úgy az a következő sorban is folytatható, ismételt idézőjelek között.) Az `msgstr` kezdetű sorok adják meg a szöveg magyar nyelvű megfelelőjét. Látható, hogy a ***.po*** fájl első „éles” bejegyzése az üres "" szöveghez kapcsolódik: a szabvány szerint az ehhez tartozó „fordítás” ad információt a ***hu.po*** fájl tartalmáról. (Ennek a formátuma eléggé kötött, és hibaüzenetet kapunk, ha valamelyik kötelező rovatát nem töltöttük ki helyesen.) Akik a C nyelvben kevésbé járatosak, azoknak könnyű a mind az angol szövegekben, mind a fordításukban megjelenő formázó részszövegeket felismerniük. Ilyen például a `\n`, ami az új sor jele, vagy a `%s` és a `%f`, amelyek arra utalnak, hogy a helyükbe egy másik szöveget vagy egy valós számot kell behelyettesíteni. Arra is mód nyílik, hogy a magyar változatban az egyes számokat, értékeket az angol eredetihez képest más sorrendben helyettesítsük be. Például az

```
msgid "Today is %s %s in the year %s."
msgstr "A mai dátum: %3$s. %2$s %1$s."
```

eredeti-fordítás párral elérhetjük, hogy az angol szöveg tükörfordítása helyett a magyar szokás szerinti megfelelő jelenjen meg a programunkban. Munkánk során azonban többnyire egyszerűbb esetekkel találkozunk. Az sem baj, ha kezdetben nem is fordítunk le minden szöveget, csak a legfontosabbakat: a kihagyott esetekben mindig az angol eredeti fog a magyar változatban is megjelenni. Látható az is, hogy a fenti szövegek többnyire nem szó szerint lettek lefordítva. Ennek oka sokszor az, hogy a magyar nyelv logikája néha egészen más, mint az angolé, s így kerülőutakra kényszerülünk.
A fordítási lépéssorozatot több grafikus segédprogram is megkönnyítheti. A régebben készült `ktranslator` program helyett újabban talán többen használják a poEdit többfelületes alkalmazást (ez ugyanis Windowson is elérhető), illetve a KDE alatt futó kbabel programot (*1–2. kép*). Akik mégis a „fapados” megoldásokat szeretik, azoknak viszont teljes szívvel ajánlható például a Midnight Commander szövegszerkesztője (*3. kép*).

A puding próbája

Sokáig – akár hónapokig is – eltarthat, amíg az elkészült ***hu.po*** fájlunk a fejlesztők jóvoltából belekerül az időszerű változatba, és élesben kipróbálhatjuk. Ha azonban ügyeskedünk egy kicsit, már a ***hu.po*** fájl készítése közben belepillanthatunk, hogyan is néz ki a honosított program magyarul, akár csak részben lefordított szövegekkel.



3.

© Kiskapu Kft. Minden jog fenntartva

© Kiskapu Kft. Minden jog fenntartva

4.

A ***hu.po*** fájlból a gettext csomag egyik segédprogramja egy ***.mo*** kiterjesztésű fájlt készít, amelyet fejlesztett programunk bináris változata könnyedén értelmezni tud. Ezt a fájlt (legyen most a neve ***ize.mo***, arra utalva, hogy az „izé” nevű programot fordítjuk magyarra) rendszerint a ***/usr/share/locale/hu_HU/LC_MESSAGES*** könyvtárban helyezik el (a Linux-változatunktól és a telepített programoktól függően ettől különböző, illetve ezenkívül más, hasonló nevű könyvtárak is szóba jöhetnek). Keressük meg a gépünkön ezt a könyvtárat, és nézzünk utána, hogy programjaink közül melyiknek van telepítve a magyar nyelvű változata.
A fenti segédprogram neve: `msgfmt`, futtatása a következő módon javasolt:

```
$ msgfmt -o /usr/share/locale/hu_HU/
➥LC_MESSAGES/ize.mo hu.po
```

Világos, hogy ehhez rendszergazdai jogosultság is szükséges. Ha ilyennel nem rendelkezünk, a saját könyvtárunkban egyszerűen hozzunk létre egy ***hu_HU/LC_MESSAGES*** alkönyvtárat, s oda dolgozzunk – ebben az esetben azonban lehet, hogy a fejlesztett programba is bele kell nyúlnunk, hogy tudassuk vele: a fordításokat nem az alapértelmezett könyvtárban kell keresnie.

Egy mintapélda

Elképzelhető, hogy egy teljesen új programot akarunk írni, amit nemzetközi támogatással szeretnénk ellátni. Programunk, az izé 1.0-s változatának ***ize.c*** nevű forráskódja a következőképpen néz ki:

```
#include <locale.h>
#define _(szoveg) gettext(szoveg)

main()
{
     setlocale(LC_MESSAGES, "");
#ifdef HELYBEN
     bindtextdomain("ize", getenv("HOME"));
#endif
     textdomain("ize");
     printf(_("Welcome to the program %s!"),
               ➥"ize");
     printf("\n");
}
```

Az első sorban tudatjuk a C-fordítóval, hogy a gettext csomaggal dolgozunk. A másodikban egy olyan rövidítést adunk meg, amellyel a későbbiekben elérhetjük, hogy minden szövegünk nemzetközi változata a `_("szöveg")` hívással is megjelenhessen (ez szabványos rövidítés, a gettext leírása is ezt a módszert javasolja, sőt a gettext segédprogramjai alapértelmezésben ezt a rövidítést támogatják). A `setlocale()` függvénnyel kapcsolunk nemzetközi üzemmódba (lehetőség volna arra is, hogy például a számokat, illetve a pénznemeket is magyar szabvány alapján jelenítsük meg, az `LC_NUMERIC` és `LC_MONETARY` változók beállításával; erről bővebben a gettext leírásában olvashatunk). Amennyiben nincs rendszergazdai jogosultságunk, az ***ize.mo*** fájlt a ***~/hu_HU/LC_MESSAGES*** könyvtárból olvassuk majd be, ezt engedélyezzük a `#ifdef` és a `#endif` közötti `bindtextdomain()` függvénnyel. A kiíratás a C nyelvben megszokott módon, a `printf()` függvénnyel történik.
Ezek után megadjuk a programunkban egyelőre egyetlen angol nyelvű szöveg fordítását a ***hu.po*** fájlban:

```
msgid ""
msgstr ""
"Project-Id-Version: ize 1.0\n"
"POT-Creation-Date: 2003-07-20 08:10+0200\n"
"PO-Revision-Date: 2003-07-20 08:32+0200\n"
"Last-Translator: Zoltan Kovacs
<kovzol@math.u-szeged.hu>\n"
"Language-Team: Hungarian\n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain; charset=ISO-8859-2\n"
"Content-Transfer-Encoding: 8-bit\n"

msgid "Welcome to the program %s!"
msgstr "Köszönti Önt a(z) %s program!"
```

Az alábbiakban két `Makefile` olvasható. Ha van rendszergazdai jogosultságunk, használhatjuk az elsőt, máskülönben csak a második fog működni. A beljebb kezdett sorokat tabulátorral kell kezdeni.

```
all: ize ize.mo

ize: ize.c
      gcc -o ize ize.c

ize.mo: hu.po
      msgfmt -o
/usr/share/locale/hu_HU/LC_MESSAGES/ize.mo hu.po
```

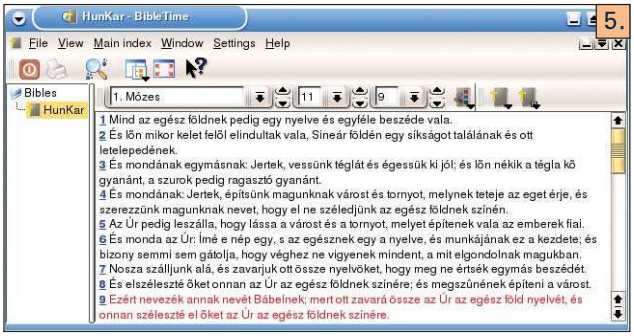
```
all: ize ize.mo

ize: ize.c
      gcc -o ize ize.c -D HELYBEN

ize.mo: hu.po
      msgfmt -o
      ➥$(HOME)/hu_HU/LC_MESSAGES/ize.mo hu.po
```

Ezek után adjuk ki:

```
$ make
$ ./ize
```

Ekkor a

```
Köszönti Önt a(z) ize program!
```



5.

szöveget kell látnunk. Ha a későbbiekben akár a *.c*, akár a *.po* fájlt módosítjuk, egyszerűen csak a `make` parancsot kell újra kiadnunk, amivel *ize* és *ize.mo* nevű bináris fájljaink önműködően frissülni fognak.

Többnyelvűség

A szép az egészben az, hogy a programunk innentől kezdve többnyelvű. Az egyes hozzáférhető nyelvek közül a program indítása előtt választhatunk:

```
$ LANG=de_DE; ./ize
```

Így például németül futna a programunk, ha a megfelelő könyvtárban megvolna az *ize.mo* fájl német nyelvű változata (mivel nincs, a feliratok angol nyelvűek lesznek).
Próbáljuk németül elindítani a Midnight Commandert, a Gimpet pedig spanyolul és csehül.
Mostantól más nemzetiségű fejlesztők, fordítók is csatlakozhatnak projektünkhöz, ha a *hu.po* fájlt például *de.po*-ra nevezik át, s az `msgstr` sorokat a megfelelő német fordításokkal helyettesítik. (Jobb, ha létezik egy központi *messages.pot* sablonfájl, amelyben az `msgstr` sorok üresek, s így a német fordítónak nem kell még a magyar fordítások kitörlésével is bajlódnia.)
Ha egy program magyar fordításával elégedetlenek vagyunk, az eredeti angol szövegeket úgy jeleníthetjük meg, ha a `LANG` változót `C`-re kapcsoljuk át (az `en_EN` érték ugyanúgy megfelelő).

Tippek és trükkök

A nemzetköziesítés éles használata számos gyakorlati kérdést vet fel. Ilyenek például a következők:

- Hogyan tehermentesíthető a fordító abban az esetben, ha a már lefordított szövegek időről időre részben (1–2 karakterben, szóban) megváltoznak? Azaz van-e mód arra, hogy ezekben az esetekben a fordítónak ne kelljen minden megváltoztatott szöveget teljes egészében újrafordítania?
- Megoldható-e, hogy a C nyelvű programban előforduló szövegekből önműködően jöjjön létre a megfelelő sablonfájl? Azaz megoldható-e, hogy a fejlesztőkre kevesebb karbantartói munka háruljon, vagyis a *messages.pot* fájlt robot hozza létre?
- Hogyan követhető nyomon, hogy a fejlesztők mely új szövegeket adták hozzá a program újabb változatához, és a fordító hogyan értesülhet a leggyorsabban arról, ha új szövegeket kell lefordítania?
- Megoldható-e, hogy már létező C-programokat gyorsan, önműködően nemzetköziesítsünk?

Mindezeket a kérdéseket (és számos továbbit) a gettext fejlesztői nagyrészt megoldották, és több segédprogramot bocsátanak rendelkezésünkre. Ilyen például az msgmerge program, amellyel a *hu.po* fájlt frissíthetjük az új változatú *messages.pot* fájl új üzeneteivel (esetlegesen arra is figyelve, hogy bizonyos szövegek időközben megváltoztak); vagy az xgettext, amellyel a C (vagy számos egyéb, például Java, Python vagy PHP) nyelvű programból az összes angol nyelvű szöveget „kirobbanthatjuk”, s azokból sablonfájlt szervezhetünk. Hasznos eszköz a gettextize is, amivel egy C nyelvű program könnyen átszabható úgy, hogy támogassa a gettext szabványt. A gyakorlatban a fejlesztők gyakran CVS-rendszert is bevonnak a programozói munkába, hogy a különböző változatok közötti eltéréseket könnyebben átláthassák (a nemzetköziesítéstől függetlenül is).
A fenti kérdésekre minden egyes program fejlesztésekor valamiféle választ szokás adni, de ez a projektektől függően más és más lehet. A megoldások többnyire a gettext használati utasításának ajánlásait követik (ám bizonyos részletekben gyakran el is térnek attól). Lássunk egy példát! A PostgreSQL adatbáziskezelő 7.3-as változatához egy éve új felügyeleti program készül pgAdmin3 néven. A nyelvi fordítás a fejlesztéssel párhuzamosan zajlik, azaz a fordítóknak rendszeresen új sablonfájlokat kell összefésülniük az általuk szerkesztett *.po* fájllal.
A nyelvi koordinátor a ➲ http://snake.pgadmin.org/pgadmin3/translation.php címen (*4. kép*) időről időre frissíti a sablonfájlt, amit a 26 fordító mindegyike, például a poEdit program *Catalog* menüjének *Update from POT file* lehetőségével fűzhet össze a saját *.po* fájljával. Ezután az új és a megváltozott szövegek honosításának megtörténtével az újonnan kapott *.po* fájlt vissza kell küldeni a koordinátorhoz, aki a program hivatalos forráskódjában (CVS-rendszeren) is rögzíti a változtatásokat. Ezt az eljárást mindannyiszor meg kell ismételni, ahányszor a sablonfájl módosul. (A gyakorlatban a fejlesztők mindig előbb járnak, mint a fordítók, így fordulhat elő, hogy a magyar fordítás szinte egyetlen programnál sem százszázalékos.)

A jövő

A gettext olyannyira jól bevált, hogy számos programozási nyelv alapértelmezésben támogatja a használatát. Ennek ellenére a bábeli zűrzavar teljes megoldása még hosszú távon is lehetetlen feladatnak tűnik. Évtizedek kemény munkájának tapasztalata alapján úgy látszik, hogy a fordítási munkafolyamat csak részben tehető önműködővé. Még egy számítógépes program fordítása során is sokszor szinte művészi tehetség kell, hogy a szövegeket anyanyelvünkön szépen, pontosan, csattanósan fogalmazzuk meg. Emiatt a fordító feladata fáradságos, hosszú, sziszifuszi munka lehet. Ezt mindenesetre kárpótolhatja az a tény, hogy a magyar fordításon felnőtt járatlan felhasználók kényelmesebben, nagyobb örömmel dolgoznak majd az adott programmal. Így a fordító közvetve igen sok új honfitárs felhasználót nyerhet meg nemcsak az adott programnak, hanem a Linuxnak is.

Tipp: töltsük le az internetről a BibleTime programot, és telepítsük mellé Károli Gáspár Biblia-fordítását (HunKar.zip). Keressük ki azt a részt Mózes 1. könyvéből, amelyik Bábel tornyának történetét meséli el (*5. kép*).

Kovács Zoltán
(kovzol@math.u-szeged.hu)
Tanársegéd a Szegedi Tudományegyetem Bolyai Intézetében az Analízis Tanszéken, matematikát és számítástechnikát tanít óraadóként a szegedi Radnóti Miklós Kísérleti Gimnáziumban.

© Kiskapu Kft. Minden jog fenntartva