06/98

Linuxové noviny

Úvodem

Pavel Janík ml., 12. června 1998

Pomalu ale jistě se blíží ona doba, kdy školy zejí prázdnotou, koupaliště jsou naopak nacpána až k prasknutí a k tomu to odporné horko. A navíc si ještě Red Hat Software vypustí novou verzi distribuce Red Hat. Jak to má obyčejný člověk stíhat? Musí chodit na zkoušky (já jich mám osm), musí se také někdy ukázat v práci, musí se také někdy ukázat doma, a to už nemluvím o tom, že musí být také se svojí přítelkyní (a jak rád). A ještě k tomu musí dělat spoustu jiných věcí (psát Linuxové noviny, nainstalovat si nový teTEX s novým pdfTEXem, napsat podporu obrázků ve formátu TIFF pro pdfTEX atd.). Ano, máte pravdu, přestanu si vylévat svoji duši a začneme hned od začátku naplno.

V červnovém čísle Linuxových novin najdete spoustu zajímavých informací, počínaje pravidelnými rubrikami Měsíc v comp.os.linux.announce, Co nového na sunsite.unc.edu?, a konče výsledky soutěže z minulého čísla Linuxových novin v článku Zasmáli jsme se!.

Karel Žák a Miroslav Janiš nás ve svých článcích seznámí s programátorskými editory Xwpe, Xfte, Grasp a Source Navigator.

Dan Ohnesorg (za spolupráce s Milanem Keršlágerem) vytvořili malé HOWTO pro čtení pošty pomocí programu Pegasus Mail na Linuxu.

Vydavatelství Computer Press vydalo novou knihu. Jmenuje se *Programovací jazyky GNU*, a protože jsme knihu dostali k dispozici, můžete si přečíst její recenzi od Yenyi.

Chcete psát lokalizované programy a nevíte, jak na to? Zkuste si přečíst článek autorů Michaela Mráky a Vládi Michla, třeba vám pomůže.

Kolektiv redakce Linuxových novin (nyní v počtu pěti osob — Ondřej „Koala“ Vácha, Pavel Juran, Bohumil Chalupa, Hanuš Adler a já) Vám přeje příjemné počtení a příjemně strávenou dovolenou s vašimi blízkými. Nejen počítači živ je člověk, pamatujte na to a bavte se (a už se prosím nikdy nehádejte v konferencích, které čtu…). ■

Měsíc v comp.os.linux.announce

Pavel Janík ml., 1. června 1998

Tak jako každý měsíc přinesla oznámení v konferenci *comp.os.linux.announce* spoustu novinek ze světa operačního systému Linux. Tento měsíc byl navíc poznamenán výstavou Linux Expo 98, a tak jsme si mohli přečíst spoustu výhodných nabídek komerčních společností. Například společnost Red Hat Software Inc. zde nabízela poslední verzi své distribuce — RedHat Linux 5.1 (kódové označení Manhattan).

Steve Emms (webmaster@linuxlinks.com) oznámil novou adresu pro službu nazvanou Linux Links (1), která shromažďuje odkazy na WWW servery věnující se Linuxu. V současné době jich má asi 1500.

Robert S. Maier (rsm@math.arizona.edu) vytvořil novou verzi balíku GNU plotting utilities (2).

Jean-Luc Fontaine zveřejnil další verzi svého zajímavého modulárního spreadsheetu (tabulkového procesoru) pod názvem `moodss-3.3`. Najdete jej na adrese (3). Na této adrese naleznete i obrázek spreadsheetu v akci.

Mark Stacey (heathclf@skynet.csn.ul.ie) uvolnil testovací verzi patche pro podporu protokolu T/TCP pro kernel 2.0.32. Protokol T/TCP je definován v dokumentu RFC 1644 a umožňuje snížit počet přenášených paketů při malých přenosech dat až o 66 %. Patch naleznete na adrese (4).

Společnost ObjectSoftware Inc. vyvinula utilitu ObjectManual, která slouží ke generování dokumentace ze zdrojových textů v jazyce C nebo C++ (podobně jako `javadoc` z jazyka Java). Program je free pro nekomerční organizace. Je k dispozici na adrese (5) pro Linux a Microsoft Windows NT.

Fabrice Bellard (bellard@email.enst.fr) vytvořil patch pro audio přehrávač mpg123, který prý umožňuje soubory mp3 přehrávat i na pomalých počítačích s procesorem i486. Patch naleznete na adrese (6).

Dennis Payne (payned@rpi.edu) oznámil novou verzi programátorského prostředí Xwpe (7), bližší informace o tomto prostředí naleznete také v článku Programátorské editory (X)WPE a (X)FTE.

Společnost Linux Central (8) oznámila zahájení prodeje nové knihy autorů Michaela K. Johnsona a Erika W. Troana s názvem *Linux Application Development*. Kdo zná oba autory, jistě si knihu nenechá ujít. Najde se nějaké vydavatelství, které knihu přeloží a vydá? Snad ano, nechejme se překvapit…Další novou knihou je kniha *Linux Network Toolkit* autora Paula G. Seryho.

Andries Brouwer (Andries.Brouwer@cwi.nl) oznámil novou verzi balíku kbd, tentokráte s číslem 0.96. Balík najdete na adrese (9).

Elmer Joandi (elmer_j@madli.ut.ee) nalezl poměrně zajímavou chybičku v produktu Source Navigator (viz článek Source Navigator) společnosti Cygnus. Source Navigator totiž poslouchá na prvním volném portu a akceptuje příkazy jako `exec cat /etc/passwd`.

Thomas Boutell (boutell@boutell.com) známý to autor aplikací pro WWW (vzpomeňme třeba `Count.cgi`) oznámil novou aplikaci, která umožňuje sdílení kalendářů — jedná se tedy o tzv. *Group Scheduling* aplikaci, která umožňuje skupinám pracovníků lépe organizovat čas.

Pavel Janík ml. (to jsem já ;-) oznámil přesun aplikace *Linux Source Driver* na novou adresu (10).

John Lindal (jafl@cco.caltech.edu) oznámil novou verzi balíku Code Crusader(11).

Markku Rossi (mtr@ngs.fi) vytvořil interpreter jazyka JavaScript, který je volně šiřitelný — najdete jej na adrese (12).

Společnost Red Hat Software Inc. (13) oznámila zahájení prodeje distribuce Red Hat Linux 5.1 (Manhattan).

Linuxové noviny připravuje a vydává České sdružení uživatelů operačního systému Linux

Frodo Looijaard (frodol@dds.nl) vytvořil zajímavý dokument zabývající se instalací knihovny `glibc` jako druhé v systému (14). ■

1 Linux Links
http://www.linuxlinks.com/
2 GNU plotting utilities
ftp://ftp.gnu.org/pub/gnu/plotutils-2.1.tar.gz
3 Moodss–3.3
http://www.mygale.org/~jfontain/
4 T/TCP patch pro kernel 2.0.32
http://www.csn.ul.ie/~heathclf/fyp
5 ObjectManual
http://www.obsoft.com
6 Patch pro mpg123
http://www-stud.enst.fr/~bellard/mpg123-486.html
7 Xwpe
http://www.rpi.edu/~payned/xwpe/alpha.html
8 Linux Central
http://linuxcentral.com
9 Kbd
ftp://ftp.win.tue.nl/pub/linux/util/kbd
10 Linux Source Driver
http://lsd.linux.cz
11 Code Crusader
http://www.cco.caltech.edu/~jafl/jcc/
12 NGS JavaScript Interpreter
http://www.ngs.fi/js/
13 Red Hat Software Inc.
http://www.redhat.com
14 Glibc
http://huizen.dds.nl/~frodol/glibc

Co nového na sunsite.unc.edu?

Pavel Janík ml., 1. června 1998

X11

X11/clocks/asclock.tar.gz — hodiny vypadající jako NeXTovská aplikace date/time

X11/screensavers/xlockmore-4.10.tgz — screensaver pro X-Window

X11/toolbars/xfce-1.2.9.tar.gz — XFCE je jednoduchý a jednoduše konfigurovatelný panel pro X-Window

X11/xutils/kjoy-0.4.tgz — utilitka pro nastavení joysticku pro KDE

X11/xutils/wmnet-1.03.tar.gz — TCP/IP loadmeter

apps

apps/crypto/mailcrypt-3.5b2.tar.gz — rozhraní Emacs/PGP pro PGP 2.6.* a PGP 5.0

apps/database/www/www-sql-0.5.1.tar.gz — CGI rozhraní k mySQL

apps/doctools/man/vmanpg-1.0.tar.gz — prohlížeč manuálových stránek pod SVGAlib

apps/editors/X/xwpe-1.5.9a.tgz — klon Borland IDE pro X-Window, ale i pro textový terminál

apps/editors/X/yudit-1.0.tar.gz — textový editor podporující unicode

apps/math/fractals/kmandel-0.36.tgz — Mandelbrotova množina pod KDE

commercial

commercial/dbox-1.62.tgz — BBS pro Linux

devel

devel/lang/objc/objc-1.8.18.tar.gz — Portable Objective C Compiler sources

games

games/multiplayer/rgXnetchess-1.0.tgz — šachy v provedení server/klient

kernel

kernel/patches/console/mda-0.15.tar.gz — ovladač pro monitor MDA/Hercules

libs

libs/X/c++libs/Xclasses-public-0.30.bin.Linux.tar.gz — binárky knihovny Xclasses

libs/graphics/t1lib-0.7-beta.tar.gz — rasterizér pro fonty ve formátu Type1

libs/libmmx-980501.tgz — rozhraní pro podporu MMX v jazyce C

science

science/visualization/plotting/ksciplot-0.3.2.src.tar.gz — grafy funkcí pod KDE

science/visualization/plotting/xFgraphs-1.1.tar.gz — grafy matematických funkcí pod X-Window

system

system/admin/frontends/kwatch-0.3.tar.gz — prohlížeč log souborů pro KDE/Qt

system/bbs/rocat-1.07.tar.gz — plnohodnotná BBS pro Linux

system/boot/ethernet/etherboot-4.0.tar.gz — ethernet boot loader

system/daemons/watchdog/softdog-1.2.tar.gz — ovladač pro programový watchdog

system/emulators/Sim8051_1.0.tgz — simulátor 8051

system/mail/pop/fetchmail-4.4.7.tar.gz — výkonný démon pro vzdálené čtení pošty

system/network/admin/firewallct-1.0.8.p3.tar.gz — HTML konfigurace firewall

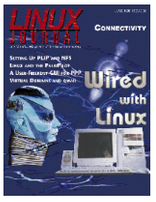
utils

utils/disk-management/mtools-3.9.1.tar.gz — známé to mtools

■

Linux Journal

Petr Bárta, 7. června 1998



Květnové číslo Linux Journalu se zabývá programováním aplikací přenositelných na jiné systémy (*cross platform development*). Prvním z článků je nástin tvorby widgetů pro Javu od R. J. Celestina.

Přehled prostředků a nástrojů pro tvorbu grafického uživatelského rozhraní nezávislého na použitém systému přináší Michael Babcock.

Pokud používáte PERL, Jeremy Impson vám ukáže, jak při ladění použít PERLovský debugger.

John Kominek používá pro programování přenositelných aplikací jazyk *Modula-3*. Ve svém článku se věnuje příkladu, jak vytvořit distribuovanou databázi.

Carl Nobile přináší ukázku ATD (*Abstract Data Type*).

O své zkušenosti s využitím Linuxu při získávání dat na německých železnicích se dělí Harald Kirsch.

Richard Schwaninger popisuje svůj pohled na využití *Tcl/Tk* pro rychlou tvorbu zákaznických aplikací.

Nevíte, jak nastavit RAID na Linuxu? Možná vám poradí Jay Munsterman ve svém článku o RAID0.

Nejen pro uživatele CDE (*Common Desktop Enviroment*) je určen článek Georga Krafta o ToolTalku a jeho využití při programování vzájemně komunikujících aplikací.

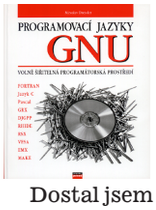
Pokud potřebujete spojit Python a databáze, a nevíte nic o DB-API pro Python, přečtěte si článek Andrewa M. Kuchlinga.

Hledáte nejpomalejší místo vašeho programu a nemůžete ho najít? A zkusili jste už některý z profilerů? Podívejte se na další článek ze série *Take Command*.

Linux Journal samozřejmě jako vždy obsahuje další články, recenze, informace, reklamy apod. Jen heslovitě: KDE, Gnome, recenze na knihu *A Practical Guide to Linux*, ochrana WWW stránek heslem. ■

Programovací jazyky? GNU!

Jan Kasprzak, 11. června 1998



Miroslav Dressler: Programovací jazyky GNU
Podtitul: Volně šiřitelná programátorská prostředí
Computer Press, 1998
ISBN 80-7226-070-7
Cena 275 Kč/303 Sk

Dostal jsem k recenzi zajímavou knihu. Titul tohoto článku je parafrází na její název — „Programovací jazyky GNU“. Autorem je Miroslav Dressler a vydalo ji nakladatelství Computer Press (1). Kniha popisuje práci s volně dostupnými vývojovými prostředky z projektu GNU — kompilátory `gcc` (překladač jazyka C/C++/Objective C), `gpc` (překladač Pascalu), `g77` (překladač Fortranu 77), program `gmake` (zajišťuje kompilaci a sestavování rozsáhlejších programů) a ladění programů debuggerem `gdb`.

Největším překvapením, které mi tato kniha způsobila, byl fakt, že kniha není zaměřena na výše zmíněné softwarové produkty obecně, ale na jejich verzi pro konkrétní prostředí — OS/2 a DOS extender EMX. V knize je zmíněna existence `gcc` a dalších produktů pod UN*Xem, existence prostředí DJGPP pod DOSem, ale hlavní zaměření knihy je prostředí EMX. Tento fakt mě celkem překvapil — proč vydávat manuál k portabilnímu programu pouze pro jedno prostředí. Vždyť obecný manuál ke GNU C a dalším programům by se podle mého názoru měl prodávat lépe.

Na druhé straně je docela možné, že u Computer Pressu přesně vědí, co dělají. Tuto knihu si zcela jistě mohou koupit uživatelé DOSu, OS/2 nebo Windows, a naleznou tam pro svůj systém detailní instrukce. A uživatel UN*Xu prostě přeskočí první kapitolu o EMX a dvě poslední kapitoly, zaměřené na integrovaná vývojová prostředí a grafickou knihovnu VESA. Uživatel UN*Xu bude také pravděpodobně natolik inteligentní, že si `GCC`, resp. `GCC.EXE` přeloží jako `gcc`.

Můj názor na knihu je v podstatě kladný a myslím si, že je dobrá jak pro uživatele UN*Xu, tak pro uživatele zmiňovaných proprietárních systémů. Já osobně jsem si velmi rád přečetl kapitolu o `gdb` (tedy pardon, o `GDB` :-) a našel jsem tam několik vlastností, které jsem ještě neznal. Je také dobré, že se touto knihou dostává informace o dalších volně šiřitelných programech do povědomí uživatelů proprietárních systémů.

K některým rysům knihy mám také své výhrady:

- autor má zřejmě jen velmi mlhavou představu o tom, co je GNU, co je Linux a jak tyto pojmy spolu souvisí. Cituji z úvodu knihy:

> Základní myšlenkou projektu GNU (*GNU's not UNIX*) je bezplatně zpřístupnit programové vybavení původně určené pro operační systém UNIX uživatelům ostatních operačních systémů. Tato myšlenka byla nejsilněji akcentována v době vytváření operačního systému Linux, se kterým se současně vyvíjel překladač programovacího jazyka C.

O tom co je hlavní myšlenkou projektu GNU lze samozřejmě diskutovat, ale podle mého názoru je to spíše vývoj programového vybavení *pro UN*X*, nikoli (jen) jeho zpřístupnění uživatelům jiných systémů. A fakt, že překladač `gcc` je o nějakých pět šest let starší než Linux, je také obecně známý.

- písmo Garamond považuji za dost okoukané, ale na druhou stranu je celkem hezké. Ovšem tímto písmem by měla být sázena celá kniha (vzorec na straně 93 je zřejmě v něčem jako Times — Quark nejspíš neumí matematiku v jiném fontu). Navíc kombinace poměrně výrazného Garamondu s lehkým Courierem je podle mého názoru do očí bijící. Ostatně, už když si v Courieru napíšete něco jako `-fomit-frame-pointer`, vypadá to dost divně.

- v některých místech chyby (sazeče?) zapříčiňují i mylnou interpretaci textu (např. strana 78, třetí řádek shora má být odsazen — levostranné mezery v `Makefile` nejsou nevýznamné).

- kniha má na obalu obrázek kompaktního disku; myslím, že by daleko prospěšnější bylo, kdyby kniha obsahovala skutečné CD s instalací GNU vývojového prostředí třeba pod EMX (i když si myslím, že na CD by se navíc mohlo vejít i DJGPP a zdrojové texty GNU programů pro UN*X, a možná i dokumentace, převedená z `info` do HTML.

Výše uvedené výhrady považuji za málo významné a knihu doporučuji ke koupi každému, kdo zmíněné vývojové prostředky používá nejen pod DOSem (OS/2, Windows, ...), ale i v nativním UN*Xovém prostředí. Existence této knihy je pro mě důkazem, že Open Source software se i u nás dostává do popředí zájmu uživatelů natolik, že se nakladatelům vyplatí vydávat k tomuto softwaru českou dokumentaci. ■

1 Computer Press
http://www.cpress.cz

Programátorské editory (X)WPE a (X)FTE

Miroslav Janiš, 9. června 1998

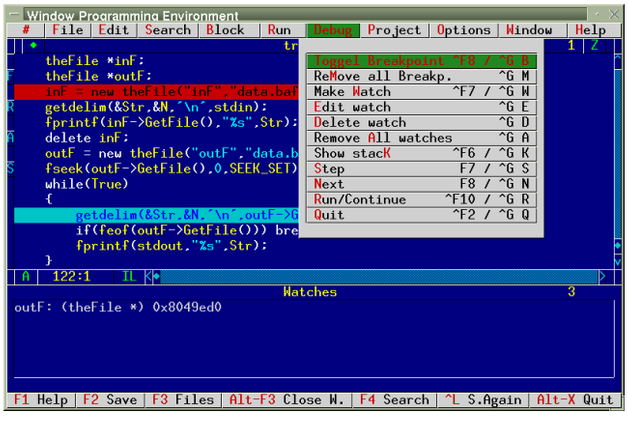
Abstract

Editory `wpe` a `fte` budou patrně bližší těm, kteří přecházejí na Linux z DO$u a UNIXové editory typu `vi` nebo emacs jsou pro ně záhadou. Oba popisované editory se více či méně úspěšně snaží napodobit „standardní“ Borlandí ovládání (které podle mne nepatří mezi nejhorší). Hned na začátku bych však zmínil i nevýhody těchto editorů — `fte` nemá verzi pro terminál a u `wpe` se mi ji vůbec nepodařilo rozchodit. Takže jejich použitelnost je zúžena na X-Window System a u `fte` ještě lze použít verzi pro textovou konzoli. (BTW: pokud používáte KDE, tak vás možná potěší verze pro tento X desktop (není ale součástí standardní distribuce `fte`).)

Editor XWPE

Téměř dokonalá kopie Borlandího rozhraní; umožňuje vše (velmi nadneseně řečeno), co si zmlsaný programátor může přát.

Stvořil ho pan Fred Kruse a ve verzi 1.4.2 existuje bez změny již od roku 1993. Teprve v nedávné době byly zdrojáky oprášeny a byl zahájen projekt „za lepší WPE“ (just kidding). Podívejte se na adresu (1). Další text se bez výjimek bude vztahovat ke staré verzi 1.4.2.



Podobnou simulací Borlandů je program RHIDE, který byl napsán pro `djgpp`, což je dosový port překladače GNU C (produkuje tak dobrý kód, že byl použit IDSoftwarem k překladu QUAKEa místo Watcom C). RHIDE existuje jak pod DO$, tak pro Linux. Linuxovou verzi jsem však nezkoušel a o té dosové si pamatuji jen to, že když jsem hned na začátek zdrojáku napsal komentář, tak při listování spolehlivě zatuhl :-) Takže toto rozhraní rád přenechám povolanějším...

Obrázek ukazuje editor `xwpe` při krokování programu (těsně před tím, než mi spadl :-)).

Stručný výčet schopností WPE

- vytváření projektů s definicí parametrů pro překladač i linker
- definice breakpointů a krokování přímo ve zdrojovém kódu, klasické výpisy obsahu proměnných (*Watch*) a zásobníku programu (*Stack*)
- odskakování na chyby a „warningy“ objevené překladačem a zpřístupněné pomocí okénka *Messages*
- možnost nakonfigurovat a využít různé překladače — kromě C(++) třeba Fortran, Pascal či LaTeX (hmm... tady je trochu problém, protože `xwpe` „cpe“ zcela svévolně libovolnému překladači spuštěnému z menu příkazem *Run/Compile* parametr `-c`, takže je třeba spouštět skript, který ignoruje první parametr (`-c`) a spouští `latex` až s tím druhým (`filename`))
- zobrazování manových stránek i GNU info — umožňuje pohodlnou navigaci dokumentem, leč tu a tam špatně interpretuje odkaz (spíš ignoruje)...
- „obarvování“ syntaxe (snaha o český překlad (spíš přepis) slova *highlight*) a konfigurace barev editoru a syntaxe
- konfigurace chování editoru (tabelátory, autoindent, autosave...), ale jinak je konfigurace možností editace dost omezená.
- a nakonec umí také zcela nečekaně v ten nejnevhodnější okamžik spolehlivě chcípnout (hlavně při prohlížení manových stránek) ;-) (ale většinou se mu podaří rozdělanou práci uložit ...)

Zhodnocení

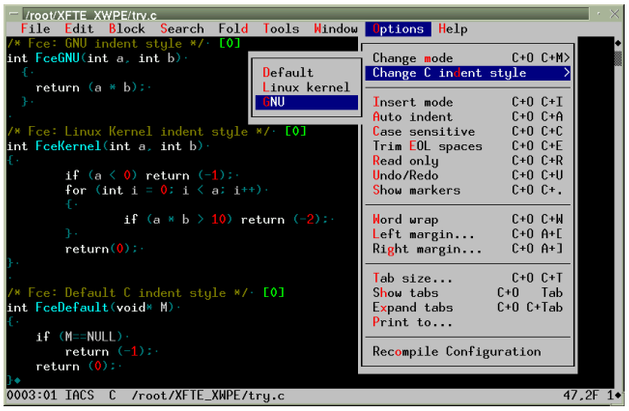
Obecně lze říci, že k programování (spíše ke zkoušení programovat) je tento editor celkem použitelný — napíšete kód, rovnou ho přeložíte, opravíte syntaktické a jiné chyby odhalené překladačem a můžete ho vesele krokovat a provádět s ním různé jiné kejkle bez nutnosti vytvářet `Makefile` a spouštět debugger. Ovšem k většímu projektu je `Makefile` stejně nutno vytvořit (co s nepřenositelným „projektfajlem“? (Nějakým kouzlem (skriptíkem) by z něho ten `Makefile` vytvořit možná šlo...)) a chyby je v zásadě lépe vůbec nedělat — ušetří to spoustu práce s následným honěním nějakého stupidního bugu ;-).

Jinak se mi zdá tento editor dost omezený — barevná syntaxe funguje jen pro pár vyvolených (C(++), Pascal, Fortran(?)...), indent je dost hloupý, a editor vůbec neumí kompletaci slov (doplňování slov podle předchozího kontextu). O definování klávesových zkratek pro vkládání často používaných výrazů si při používání tohoto editoru můžete nechat jen zdát. A právě tyto vlastnosti má editor `fte`.

Editor XFTE

Editor `fte` si neklade za cíl kopírovat IDE. Je to jen editor. Ovšem editor vhodný pro programátory zvyklé Borlandímu ovládání. Zdá se, že byl původně stvořen pod OS/2 a teprve postupem času portován na ostatní systémy i „systémy“ — DOS, Linux, Win32. Napsal (a naprogramoval :-)) ho pan Mirko Macek a je možné ho získat na (2).

Zde existuje ve verzi pro X-Window a linuxovou konzoli (hovoříme-li o možnostech týkajících se Linuxu). Někde na adrese (3) lze též najít port pro KDE od jiného autora. Tato verze má tu výhodu, že pokud máte správně nainstalovanou českou klávesnici pod X-Window (čili chodí-li vám třeba pod KDE aplikacemi), tak vám zde bude chodit též.



`Xfte` v plné kráse — ukázka různých indent stylů, ty `[0]` značí foldy. Celý text by po zavření foldů měl tři řádky…

Méně stručný výčet schopností FTE

- Konfigurovatelné zvýraznění syntaxe (highlighting) — můžete přidávat vlastní definice různých formátů souborů. Nadefinovány jsou tyto jazyky/formáty: C(++), Java, REXX, Perl, Pascal, Ada, UNIX shell skripty, tcl, HTML, LaTeX, E-mailové zprávy, DIFF a další. Ne všude funguje hilite tak, jak by si člověk představoval — u HTML nejsou nadefinována některá klíčová slova a JavaScript to vůbec nerozezná :-) V shellovských skriptech je zase jako příkaz zvýrazněno jen první slovo, takže pokud voláte nějaký program celou cestou, tak je zvýrazněn jen první adresář (kromě lenosti však nic nebrání odstranění těchto nedostatků přímo v konfiguračních souborech).
- Speciální indent pro C(++), Perl, Javu, REXX, přičemž pro C existují tři módy — standardní, GNU a Linux kernel styl. Céčkovský indent mne opravdu nadchl — na patřičné místo na řádku kursor odskakuje hned po stisknutí klávesy Enter a inteligentně rozpoznává složené závorky. To znamená, že zdroják píšete bez jakýchkoli starostí o formátování textu — vše se provádí automaticky. Další pomůcka usnadňující orientaci v textu je, že při najetí kursorem na závorku (jakoukoli — `[`, `]`, `{`, `}`, `(` a `)`) se zvýrazní odpovídající závorka.
- Schopnost kompletování slov na základě předchozího kontextu — pokud se v předchozím textu již dané slovo vyskytuje, stačí napsat několik prvních písmen a stiskem kláves `Ctrl+Tab` se slovo automaticky dopíše (pro lenochy jako jsem já je to přímo terno…). Bohužel to již nefunguje, když něco dopisujete na začátku zdrojáku a slovo, které byste chtěli doplnit, se sice v textu již vyskytuje, ale nachází se za inkriminovaným místem :-(
- *Folds* (čti faldy ;-)) — ačkoli je pro ně vytvořeno speciální menu, teprve nedávno (cca půl hodiny) jsem pochopil, k čemu tohle zvíře vlastně je, neboť jsem měl čas, chuť a hlavně důvod (psaní tohoto článku) se tím zabývat. Zdá se, že je to velmi užitečná schopnost zpřehlednit vytvářený text — např. v C zdrojáku můžete pomocí menu *Fold/Create folds at routines* vytvořit „faldík“ pro každou funkci. Potom stiskem kláves `Alt+Gray /` všechny foldy zavřete. Text se poté jeví jako seznam funkcí — zobrazí se totiž pouze první řádek ve foldu. Na libovolný fold lze najet kursorem a kombinací kláves `Ctrl+Gray +` a `Ctrl+Gray -` je lze otvírat či zavírat. Foldy lze vytvářet i pomocí regexpu či jak se vám zachce — viz menu *Fold*. Není však vše zlato, co se třpytí… U foldů lze definovat pouze začátek. Ukončení zařídí buď definice dalšího foldu nebo konec souboru. To vede např. při „ofoldování“ zdrojáku podle funkcí (viz výše) k tomu, že se komentáře před funkcí, které se jí většinou také týkají, dostanou na konec foldu, ve kterém je předcházející funkce. Je zde však stále možnost použít regexp — takže pokud píšete komentáře k funkcím ve stylu `/* Fce: … */`, můžete vytvořit foldy přes regexp pomocí výrazu `Fce:`.
- Chování editoru lze ovlivňovat na poměrně nízké úrovni (víte, jak to myslím — assembler je také programovací jazyk nízké úrovně, ale lze v něm udělat všechno). Pro každý typ dokumentu je vytvořen zvláštní konfigurační soubor, ve kterém je nadefinován způsob obarvování syntaxe, speciální klávesové zkratky, přídavná menu (viz HTML), formátování textu, okraje a spousta dalších vlastností. Vše lze libovolně měnit a upravovat podle vlastních představ. Je však pravda, že je v těch konfiguračních souborech trochu chaos, ale když začnete souborem `main.fte`, tak se snadno zorientujete. A editor `fte` má pro tyto konfiguráky zvláštní mód s barvičkami :-). Po editaci konfiguračních souborů je třeba ještě vytvořit „překlad“ pomocí prográmku `cfte` a překopírovat ho do souboru `$HOME/.fterc`. Potom znovu spustíte `fte` a je to.
- Překlad a opravy zdrojových textů — fte umožňuje spouštět překladač na pozadí (implicitně spouští `make`, takže je třeba vytvořit i `Makefile` (pro který je též zvláštní mód)) a odchytává si jeho chybové hlášky. Potom lze pomocí okna *Messages* skákati po chybách a opravovati. Žádné debugovací schopnosti (krokování či breakpointy) to však nemá — je ovšem otázka, zda je to bug nebo feature (`fprintf(stderr, "Err: blah\n");` také není k zahození).

Zhodnocení

Ačkoli ladící schopnosti editoru fte jsou zredukovány na okénko s chybovými hláškami, přesto si tento editor snadno získal mé srdce — jeho implicitní nastavení mi totiž maximálně vyhovuje, a tak změny v konfiguraci nebyly nějak dalekosáhlé. Přidal jsem si pouze mód pro českou klávesnici, zkratku pro ` ` v HTML módu (`Ctrl+Space`) a ještě pár maličkostí jako používání českého fontu nebo předefinování zkratky `Ctrl+Tab`

na Shift+Tab kvůli KDE. Samozřejmě, že i tento editor jako většina jiných má své nedostatky, o kterých jsem se zmiňoval výše, ale tyto nedostatky bohatě vyvažují možnosti, které tento editor poskytuje. ■

1 XWPE
http://www.rpi.edu/~payned/xwpe/
2 XFTE
http://ixtas.fri.uni-lj.si/~markom/fte
3 K Desktop Environment
http://www.kde.org

Grasp

Karel Žák, 9. června 1998

A toto je můj favorit — GRASP neboli *Graphical Representations of Algorithms, Structures and Processes*. Nejedná se o žádného vývojářského obra, ale naopak o docela úhledný editor, který svou funkci pro programování projektů o několika souborech a pár stech řádcích plní na výbornou. Osobně ho používám asi 1 rok a jsem nadšen.

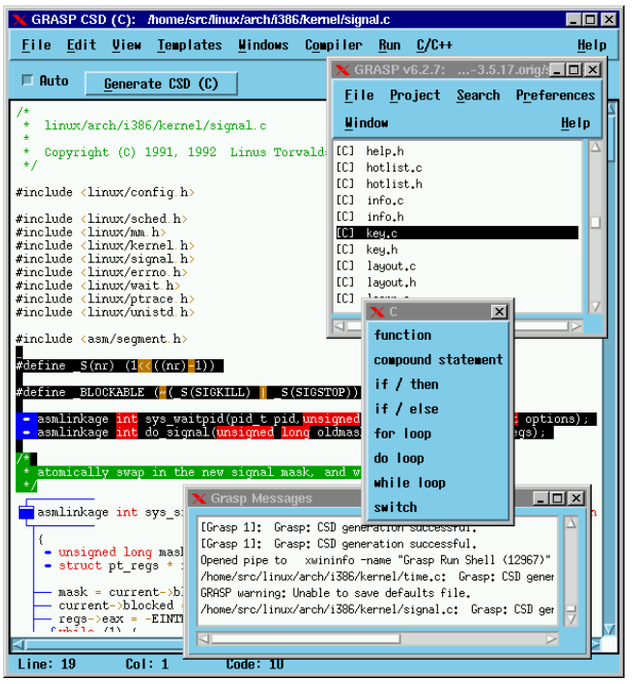
Současná verze je 6.2.7 pro Unix (29. dubna 1998) a 6.2.8 pro Win95/NT (25. března 1998).

Program je napsán pro X Window za použití Motif 1.1.2. Pochopitelně autoři nezapomínají ani na ty co nemají Motif a program lze stáhnout na adrese (1) jako staticky slinkované binárky.

GRASP je připraven pro jazyky Ada 95, C, C++, Java a VHDL. GRASP je integrován do GNU rodiny kompilátorů pro Adu, C a Sunovského `javac` pro Javu (dle informací na Webu).

```
/* kompilovat jedine KDYZ M$ uvolni source od wokenic */
int ta_prava_laska() {
  int i;
  for(i=0; i=10; i++) {
    if (i<5)
      printf("Ctete LN pravidelne?\n");
    else  {
      printf("Cteme LN moc radi a pravidelne !\n");
      printf("...a nikdy neprestaneme\n");
    }
    switch (i) {
    case 1:
      /* neco */
      break;
    case 2:
      /* neco */
      break;
    }
  }
  printf("..proto rikejme:\"kdo je nejhorsi?\"\n");
  while (0) {
    printf("De-Bill Gatesuuuu (a patri na /dev/null)..\n");
  }
  return 0;
}
```

Jako první se při spuštění objeví okno obsahující soubory projektu, pokud žádný projekt nemáte tak logo programu. Zde se lze rozhodnout jestli si založíte nový projekt nebo si prostě jen tak zeditujete nějaký ten soubor. Tato možnost mi tak trochu chybí u Source Navigátoru, který předpokládá, že pokud někdo něco chce, musí si nejdříve vytvořit projekt. Pro své interní informace si GRASP vytváří soubor v adresáři, ve kterém se nacházejí editované soubory.



Pak už stačí jen vybrat jazyk, ve kterém bude vámi editovaný soubor, nebo přímo vybrat soubor v souborech zařazených do projektu, a objeví se vlastní editor.

Editor podporuje všechny obvyklé funkce (tedy obarvování klíčových slov, zarovnávání, spouštění `make`, run programu atd.), ale hlavně CSD. A to je ten důvod proč o GRASPu píši.

CSD — *Control Structure Diagram* je nádherná věc, kterou můžete vidět na přiloženém obrázku. Ano, to jsou ty „čáry“, které nám krásně zpřehledňují zdrojový kód. Na první pohled je ihned patrné kdy a kam se vracíte, kde je jaký cyklus, `if` nebo něco jiného. Součástí značení textu může též být možnost číslování řádků.

Co se mi také líbí je, že pokud uděláte chybu (např. zapomenete středník), tak vás program taktně upozorní, že s takovým kódem CSD prostě neudělá (máte-li tedy CSD v celém souboru v pořádku, je pravděpodobné, že i základní syntaxe kódu je OK). O případných chybách GRASP informuje programátora v okně vzkazů.

Další zajímavou vlastností GRASPu je expanze maker. To znamená, že editor makra ukrytá v souborech includovaných do editovaného souboru pro vlastní kontrolu expanduje (ale na obrazovce a v souboru vám pochopitelně zůstane to co jste napsali). Tak např. pokud bude `soubor.h` a v něm makro `AHOJ` a my toto makro budeme používat v souborech `.c`, bude GRASP kontrolovat jak toto makro „zapadá“ do souborů `.c`. Pokud tedy bude v makru `AHOJ` chyba, budeme na tuto skutečnost upozorněni. GRASP tedy použité `AHOJ` nebere jen jako text, ale jako opravdové makro. Lze nastavit, z jakých `#include` souborů bude expanze probíhat (lze zahrnout i `/usr/include`).

V GRASPu si můžete také definovat vlastní šablony. Já jsem to například používal při psaní programů používajících knihovnu mSQL, kdy se mi nechtělo psát pořád ty šílené názvy funkcí. Jen stačilo nadefinovat funkce nebo i cykly (a ostatně cokoliv) do souboru se šablonami, a v menu editoru se vše objevilo. Pak pokud chcete něco z toho co jste nadefinovali, jen zamyšujete do tohoto menu a text ukrytý pod touto položkou je GRASPem napsán. Na dalším obráz-

ku je toto menu se šablonami dodanými s programem to celé světle modré (…také čtete černobílý výtisk?) okénko.

Mezi další příjemnosti patří barevný tisk do PostScriptu, kdy na výtisku je pochopitelně možné mít i CSD (to by mohlo být užitečné pro všechny tvůrce literatury o programovaní (už vidím ty krásné a přehledné příručky a ne ten šedý chaos, na kterém oči umdlévají — ale vraťme se z obláčku na zem…)).

Nově je v GRASPu interface pro `grep`, ale to si už asi každý představí sám. A dobrá zpráva na konec, v další verzi má být podpora Tcl a Perlu, a tak si přijdou na své i nekompilovanci… ■

1 Grasp
http://www.eng.auburn.edu/department/cse/research/grasp/

Source Navigator

Karel Žák, 9. června 1998

Pokud se u GRASPu zmiňuji o tom, že se hodí i pro psaní malých prográmků, tak Source Navigator (SN) bych zařadil do přesně opačné kategorie IDE programů. Pochopitelně i SN lze použít na psaní několika málo řádek, ale pravděpodobně při něm nevyužijete všechny jeho možnosti a jak dále uvidíme, bylo by to tak trochu jako používat „dělo na komáry“.

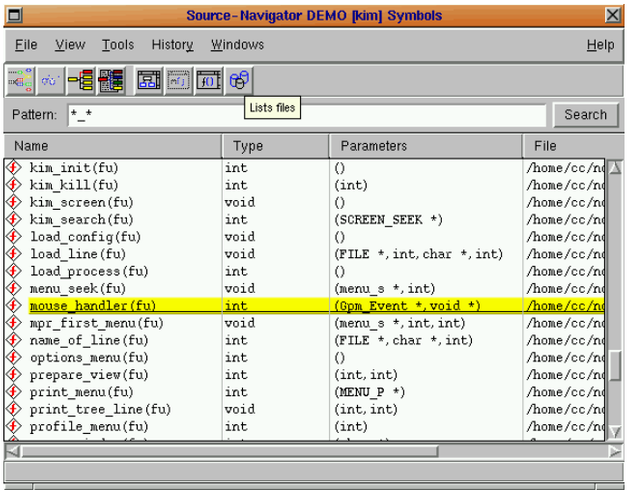
Verze kterou já používám je tzv. „lite version“, tuto lze získat na (1). Její omezení (na které jsem narazil) je neochota pracovat s projekty o více než 50 000 řádkách. Zájemcům o plnou verzi sám SN napoví hned při spuštění, kam se obrátit a pro koho si dolárky připravit (ano, ano, žádné GNU).

SN je kompletním prostředím pro vývoj jedním vývojářem (např. program C-Forge to umí i pro tým vývojářů rozesetých po síti, ale o tom až někdy příště). Součástí balíku je manažer projektů, manažer souborů a funkcí, interface pro debugger a `make` a hlavně editor. Vše je velice snadno a různorodě konfigurovatelné.

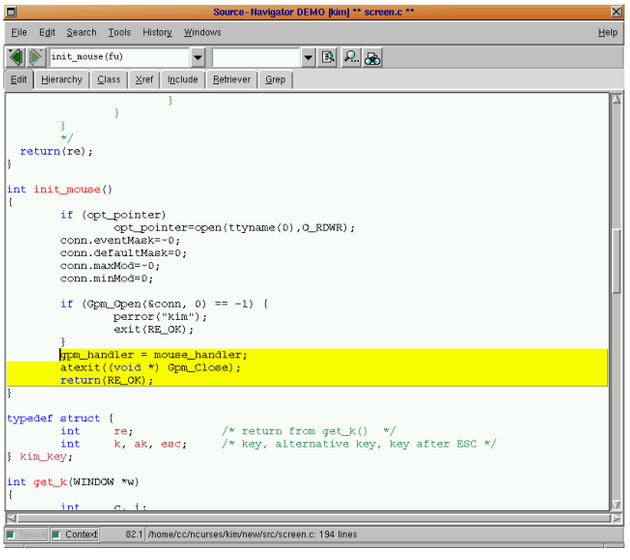
SN umí aplikovat všechny své možnosti na Tcl, Javu, Fortran, Cobol, C/C++, asm. Ke všem jazykům je možno nastavit i externí editor a specifikovat koncovky souborů. Bohužel k C++ a C přistupuje stejně. Možná by nebylo od věci, pokud je editován `soubor.c`, nenutit programátorovi na pracovní plochu nástroje na práci s objekty, ale uvidíme, vše se vyvíjí.

Jak je dobrým zvykem, přistupuje i SN k psaní programů jako k tvorbě projektů. Projekt je možné vytvořit i z existujících souborů. K tomu slouží něco co lze nazvat manažerem projektů. Zde je možné přidávat a ubírat adresáře, soubory do projektu, ale i spojovat projekty atd. Při vytvoření projektu udělá SN v určeném adresáři jeden soubor a adresář, ve kterých si udržuje databáze o zdrojovém kódu. Pro každý projekt lze vytvořit specielní nastavení všeho nastavitelného. Tedy krátce:

- barvy a fonty pro asi 40 druhů textů (např. komentáře, deklarace, …), oken a menu
- tisk
- HTML viewer
- bug report
- debugger
- make
- version control system (výběr s RCS, CVS, SCCS).
- tvorba záloh (.bak)
- automatické zarovnávání
- převod tabulátorů na mezery
- nastavení cache
- a ještě moc a moc věciček…



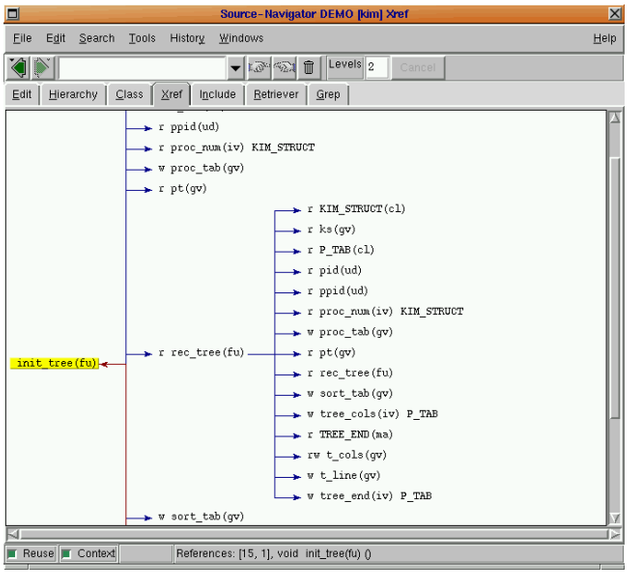
Po vytvoření projektu se objeví něco jako manažer souborů. Ten je schopen zobrazovat buď seznam souborů nebo velice pěkný seznam funkcí (na obrázku). Právě schopnost zacházet s projektem z hlediska funkcí a ne jen souborů je velice efektivní. Programátor pak už nemusí přemýšlet, kde v jakém souboru je jaká funkce, ale jen v seznamu funkcí, který lze třídit dle nastavitelného filtru, najít tu pravou a editovat.



Editor v sobě sdružuje hned několik hezkých nástrojů (některé lze volat i z manažera souborů). Editor je schopen zarovnávat, zvýrazňovat klíčová slova, kopírovat, zacházet s bloky a hlavně umožňuje hledat a hledat (v aktuálním souboru nebo v celém projektu).

Pokud kliknete pravým tlačítkem myši na jakoukoliv funkci nebo proměnnou, editor vám nabídne menu, pomocí něhož se přemístíte na implementace nebo deklarace dané funkce nebo proměnné. Nevíte-li tedy, co to je např. za položku `blabla` ve struktuře `něco`, stačí kliknout na `něco->blabla` a podívat se, kde a co a jak je ve vašem projektu spojeno s touto položkou. V horní části editoru je opět přístupný seznam se všemi objekty, proměnnými, makry a funkcemi. Dále pak seznam toho, co bylo naposledy hledané. Aby se vývojář při práci neztratil, jsou dostupná i tlačítka (šipky) back/forward, které vás vrátí na vaši předchozí/následující pozici v projektu.

Editor umožňuje i jiná zobrazení než jen klasické (záložka Edit) zobrazení (textu) kódu. Na ostatních záložkách je možné zacházet s Class, Retriever, Hierarchy (bohužel pro svou nedotčenost C++ tuto část jaksi taktně vynechám, ale z hlediska koncepce SN lze předpokládat, že i tato oblast je zde velice dobře propracovaná) a pak (a to se mi moc líbí) záložky Xref a Include.



V Xref je možné nechat si dle nastavené hloubky vnoření doslova nakreslit jaké funkce a proměnné jsou volány a použity v dané funkci. Tak např. na dalším obrázku je znázorněno, že funkce `init_tree` volá funkci `rec_tree` a u obou funkcí i použité proměnné. Prakticky tak lze udělat mapu vzájemných provázaností funkcí v celém programu, pokud jako začátek nastavíte `main()` a levels dostatečně velký (a procesor máte dostatečně výkonný). Pochopitelně je možné se kliknutím přemístit do dané funkce. Include umožňuje to samé, ale u souborů. Použití záložky Grep je myslím jasné.

SN, jak už zaznělo, má i interface pro debugování. Jako debugger používá defaultně gdb. Aktuálně prováděný kód zobrazuje SN v okně editoru, debugovat lze prakticky bez psaní příkazů na příkazovou řádku gdb — stačí jen klikat. V tomto mne SN nějak neoslovil a kvalitou ho pravděpodobně předčí DDD. Já osobně raději používám samotné gdb.

Převratné není ani rozhraní pro `make`. Obsahuje vše co lze standardně očekávat (ukazování na chyby odhalené při kompilaci, nastavení parametrů pro `make` atd.).

Co mi u SN chybí — a je např. u Emacsu — je skrývání podmíněně kompilovaných částí kódu dle direktiv `#ifdef` (bylo v emacsu pěkně předvedeno na Cikháji). Další trochu nepříjemnou věcí je doba ukládání souborů — SN si totiž znova vytváří databáze informací o zdrojovém kódu a to trochu trvá.

To je velice zhruba vše. Pravděpodobně jsem vynechal mnoho detailů, které SN umožňuje, ale to není náplní Linuxových novin, to se musí zažít při práci s SN. Na závěr lze říci, že se jedná o velice dobrý IDE software i pro opravdu velké projekty, který by bylo zajímavé vidět v plné verzi (na domovské stránce ho mají vyfocený v úhledné, ale bohužel typicky komerční krabici) a vnutit mu např. celý kernel Linuxu — a já myslím, že i s takovým balíkem zdrojáků by bez větších problémů pracoval pro zdar svého majitele. ■

1 Cygnus
http://www.cygnus.com

Kterak poštu na Linuxu Pmailem čísti

Dan Ohnesorg a Milan Keršláger, 12. června 1998

Pokud se rozhlédnete po českých školách, zjistíte, že téměř všude se používá jako klient elektronické pošty Pegasus Mail, dílo to Davida Harrise. Jeho popularita je celosvětově obrovská a jeden čas byl dokonce prostřednictvím firmy Novell prodáván pod názvem First Mail jako oficiální klient novellského poštovního systému MHS, posléze ho v této funkci nahradil GroupWise. Pegasus Mail je šířen jako freeware, s několika výhradami — nejsou k dispozici zdrojové texty, program se nesmí prodávat (může být součástí nějakého placeného balíku, ale nesmí být prodán sám o sobě), nesmí se používat pro rozesílání nevyžádaných reklamních mailů a nesmí být modifikován bez předchozího svolení autora.

Velikou výhodou je úzká integrace se serverem Novell Netware. Pegasus mail si načítá konfiguraci přímo ze serveru, umí lokálně doručovat přímo uživatelům, bez potřeby dalšího programu běžícího na serveru, umí vyhledávat lokální uživatele podle jmen uložených v tzv. *bindery* databázi (mnohem složitější obdoba souboru `/etc/passwd`, dostupná ke čtení a zápisu jedině službami běžícími na serveru (no za tuto větu by se mi každý hacker vysmál, ale pro naše účely toto zjednodušení stačí)). Podporuje tzv. *nástěnky*, které fungují asi jako news, ale pouze lokálně. Je velice dobře rozšiřitelný, umožňuje uživatelskou definici menu, tím pádem také překlad do jiných jazyků, uživatel si definuje vlastní tabulky pro překlad mezi znakovou sadou svého počítače a ISO znakovými sadami, používanými pro transport pošty po síti. V současné době jsou k dispozici kompletně české překlady jak menu, tak nápovědy. Pmail je k dispozici ve verzích pro DOS, Windows, Windows 95/NT a Macintosh. Obrovskou výhodou je identický formát používaných souborů, takže si můžete poštu číst současně pod všemi zmíněnými operačními systémy s výhradou, že verze pro Macintosh není lokalizovaná a asi v dohledné době ani nebude. Vyrovná se i s tím, že je spuštěn vícekrát na více stanicích v síti pod stejným uživatelem. Není ale možné ze dvou počítačů modifikovat stejnou složku, kam se pošta ukládá, to ale obvykle příliš nevadí, protože novou poštu lze číst bez omezení.

Logika jeho práce je velmi prostá a dobře zdokumentovaná. Mějme uživatele A, uživatel A má pod Novellem číslo AN, které je mu přiděleno při vytvoření účtu na serveru. Tímto číslem je uživatel identifikován v bindery databázi.

```
#!/bin/sh

SMTP_DIR="/home/nwserv.sys/mail/smtp"

for file in $SMTP_DIR/*
do
  if [ -e $file ]; then
    FROM=‘grep "^From:" $file | head -1 | sed -e ’s/.*<//’ -e
    ’s/>.*//’‘ TO=‘fromdos < $file | sed -n ’2,/^$/p’‘ echo -n "
    $FROM -> $TO ..." fromdos < $file | sed ’1,/^$/d’ | sendmail $TO
    echo " done." rm $file
  fi
done
```

Výpis č. 1: Script pro odesílání pošty do Internetu

Na disku, který se obvykle jmenuje SYS: existuje adresář \MAIL\AN, který má uživatel sám pro sebe. Tam se ukládá veškerá přicházející pošta, každá zpráva do jednoho souboru, který se jmenuje nějakým náhodným číslem (podle data, hodiny a výsledku fce `random()`). Tomuto souboru je přidělena přípona `.cnm`. Pmail po startu hledá všechny souboru `*.cnm` a zobrazí je. Cnm soubory obsahují čistou mailovou zprávu ve formátu podle RFC 822 a navazujících norem (MIME a pod.). Uživatel si dopisy potom přesouvá do složek podle své volby, ve složce je již více dopisů v jednou souboru, nebo může využít služeb automatického filtrování a přesouvat dopisy do jednotlivých složek při otevření nové pošty pomocí pravidel, která si sám nadefinuje. Verze pro Windows navíc umožňuje aplikovat filtrovací pravidla na odesílané dopisy nebo na jednotlivé složky. Také při odesílání lze na dopis také aplikovat pravidla, která rozhodnou o osudu zprávy. Pokud je zpráva lokální, Pegasus mail zjistí pro jméno adresáta jeho číslo a uloží mu zprávu do jeho adresáře \MAIL\číslo. Pokud zpráva splní nějaké jiné filtrovací pravidlo, adresa třeba začíná FAX: je např. spuštěn externí program. Pokud je vyhodnocena jako Internetová, je uložena do předem vybraného adresáře kde je zpracována typicky dalším Harrisovým programem — Mercurym, který zajistí doručení. Toto chování je implicitní, ale dá se změnit, třeba při instalaci na server Windows NT se místo čísel používají přímo jména uživatelů.

Co z předchozího vyplývá pro nás. Je zřejmé, že by nebylo složité došlou poštu ukládat do adresáře MAIL a odesílanou vyzvedávat z adresáře určeného pro Mercuryho prostředky Linuxu. Uživatelům by to přineslo „svobodu“ lokálního zpracování pošty, snadné ukládání příloh dopisů, rychle startující program, nepotřebující služby často problematického telnetu a bezproblémovou transparentní podporu češtiny. To samozřejmě nemohlo nezůstat nepovšimnuto a tak bych vás rád seznámil s řešením, které použil Milan Keršláger. Výhodou je i to, že budete moci použít Linux i na mnoho dalších funkcí, pro který byste jinak potřebovali další stroj.

Budete potřebovat emulátor služeb serveru Novell Netware, který se pod Linuxem jmenuje Mars_nwe. Dále samozřejmě fungující IPX na stanicích a Pegasus mail. Mars na Linuxu asi máte, je součástí běžných distribucí, jak se instaluje IPX na stanice je popsáno v dokumentaci Marsu.

Pegasus mail pro DOS (spolu s `pconfig.exe`) najdeme na URL: (1)

Doporučuji však používat spustitelný soubor z poslední preview, podle naši zkušeností je dostatečně stabilní a spolehlivý: (2)

Instalace Pmailu spočívá v rozbalení do adresáře, který si vyberete. Potom nastavíte spuštěním programu `pconfig.exe` cesty k adresáři pro poštu, jméno serveru a podobně, tato část je popsána v konfiguraci serveru. Spustíte program `pmail.exe` a kouknete se jak funguje. Při tom vznikne v poštovním adresáři soubor `pmail.ini`, ten si prohlédnete, nastavíte konfigurační volby podle vlastní úvahy, především ale nastavíte jako kódování místo ISO-8859-1 ISO-8852-2. Potom soubor okopírujete pod názvem `pmdflts.ini` do adresáře, kde je uložen `pmail.exe` a smažete nepoužité konfigurační parametry. Podle této šablony se budou zakládat konfigurace uživatelů. Pro korektní funkci češtiny bude ještě potřeba stáhnout českou podporu ze stránek pana Jiřího Kuchty na (3) nebo ze stejného adresáře odkud jste stáhli základní instalaci `pmailu`. Instalace popsána tamtéž. Tím máme `pmail` funkční a dáme se do serveru.

Na serveru je konfigurace následující, Pmail umí odchozí poštu směrovat do tiskové fronty nebo do sdíleného adresáře. Pro naši potřebu je výhodný sdílený adresář, protože do něj Pegasus odchozí dopisy jednoduše umístí jako soubory, které můžeme snadno zpracovávat skriptem. Informace o umístění tohoto adresáře je uložena v bindery databázi a tak se nemusíme starat o jednotlivé konfigurace pro každého uživatele zvlášť. K jejímu nastavení slouží program `pconfig.exe`, který je součástí distribuce Pegasus mailu. Toto nastavení respektují obě verze Pegasa (DOSová i pro Windows). Jako SUPERVISOR (ekvivalent roota na serveru Novell) spustíme `pconfig.exe` a nastavíme v menu: „SMTP Internet mail Interface:“

```
Spool interface?:   Y
Output path:        SYS:MAIL\SMTP
Enabled?:           Y
Preferred?:         N
Use always?:        N
This server’s name: server.domena.cz
Our time zone:      +0100 MET
Organization:       Nějaké jméno naší organizace
```

Umístění `spool` adresáře je prakticky libovolné, pokud ho chceme skrýt, není špatné ho umístit jako klasický mailový adresář do SYS:MAIL. Uživatele potřebují do tohoto adresáře pouze právo C (*Create*), což je v Linuxovém prostředí `drwx-wx-wx`. Položka „This server’s name“ určuje, co je přidáno za jméno a zavináč v adrese odesílatele. Může to být konkrétní jméno stroje nebo jen doména — záleží na konfiguraci našeho poštovního systému. Časová zóna je ponechána na nás. Uvedená hodnota je starší, poslední

```
#!/usr/bin/perl

$mail_dir  = "/home/nwserv.sys/mail/user";
$spoll_dir = "/var/spool/mail";

sub cnm_from_file {
  $number = 0;
  $close  = 0;
  while ( $line = <VSTUP> ) {
    chop ($line);
    if ( $line =~ /^From / ) {
      while ( -e "${mail_dir}/${user_name}/${number}.cnm" ) {
        $number++;
      }
      if ( $close != 0 ) {
        close (VYSTUP);
      } else {
        $close = 1;
      }
      if (! open (VYSTUP, ">${mail_dir}/${user_name}/${number}.cnm") )
      {
        print "nemohu zapsat do souboru
        ${mail_dir}/${user_name}/${number}.cnm\n"; return 1;
      } else {
        print "  vytvarim ${user_name}/${number}.cnm ...\n";
      }
    }
    print VYSTUP "$line\r\n";
  }
  close (VYSTUP);
  return 0;
}

opendir(SPOOL, $spoll_dir) || \
   die "nemohu cist z adresare $spoll_dir";
while ( $user_name = readdir(SPOOL) ) {
  if ( -f "${spoll_dir}/${user_name}" && -s
  "${spoll_dir}/${user_name}") {
    rename ("${spoll_dir}/${user_name}",
    "${spoll_dir}/${user_name}.lock") || \
         die "nemohu prejmenovat soubor ${spoll_dir}/${user_name}";
    open (VSTUP, "${spoll_dir}/${user_name}.lock") || \
         die "nemohu otevrit soubor ${spoll_dir}/${user_name}.lock";
    if ( &cnm_from_file ) {
      close (VSTUP);
      rename ("${spoll_dir}/${user_name}.lock",
      "${spoll_dir}/${user_name}");
    } else {
      close (VSTUP);
      unlink ("${spoll_dir}/${user_name}.lock") || \
           die "nemohu smazat soubor ${spoll_dir}/${user_name}.lock";
    }
  }
}
exit 0;
```

Výpis č. 2: Třídění pošty

dobou se setkávám se zkratkou CEST, ale jistě si vyberete formát dle vlastní úvahy. Mezi letní a zimní časovou zónou musíte přepínat „ručně“. Jméno organizace je volitelné a bude také přidáno do hlaviček odesílaných dopisů.

Měl bych se také zmínit o bezpečnosti. Obvyklá situace je taková, že všichni uživatelé mají právo zápisu do poštovních adresářů všech ostatních uživatelů. To není šťastné řešení, zvláště pokud dbáme na bezpečnost. Umožňuje totiž kromě přímého doručení dopisu (jednoduché vytvoření `*.cnm` souboru v adresátově mailovém adresáři) také podvrhnout komukoliv jakýkoliv soubor, který uživatel ještě v poštovním adresáři nemá. Nemusím zdůrazňovat, jak

to může být nepříjemné. Řešením může být odebrání tohoto práva a ponechání jediného přístupného adresáře pro všechny uživatele — spool adresáře pro odchozí dopisy. Pak je možné doručit všechny dopisy jen přes externí poštovní systém (náš script nebo Mercury) a odstraníme tak zmíněné potenciální nebezpečí. V tomto případě je však nutné nastavit v `pconfig.exe` volbu „Use always?“ na „Y“, aby se Pegasus nepokoušel místní poštu doručovat přímo, ale použil spool adresář.

Celá finta, jak poštu přijímat a odesílat, je realizována dvěma scripty. Jistě existují i elegantnější řešení přes `procmail` a podobně, ale tohle je vyzkoušené a funkční.

Začneme tím jednodušším scriptem, který slouží na odesílání pošty do internetu. Je na výpise Script pro odesílání pošty do Internetu.

A co vlastně dělá? Musí zajistit předání pošty ze sdíleného adresáře do fronty `sendmailu`. Pmail odesílanou poštu ukládá do adresáře, kterému se v `pconfigu` říká spool directory, v našem scriptu je to `SMTP_DIR` a skript z něj vybere všechny soubory a v nich nalezne položku `From:` a `To:` (tyto položky jsou v souboru dvakrát, jednou pro Mercuryho a podruhé v těle zprávy, script vybírá první variantu, druhá nemusí být např. u BCC: validní, z těla zprávy je vybíráno `From:` ale jen pro diagnostické účely). Poté je vyvolán program `sendmail`, který poštu uloží do fronty.

A teď složitější script Třídění pošty.

Ten funguje takto:

Vybírá poštu z adresářů `/var/spool/mail/jmeno` a rozláme ji na jednotlivé soubory. Z těch vyrobí `*.cnm` soubory, které uloží do poštovního adresáře uživatele. Volí trochu odlišný postup než Pmail, vytváří soubory `0.cnm`, `1.cnm` a pokud již takový existuje, inkrementuje název, dokud se netrefí do volné pozice. Mail v unixovém mailboxu začíná `^From jméno\` od začátku řádku. (Není tam dvojtečka.) Aby se nemohlo stát, že dopis obsahující od začátku řádku `From` bude rozdělen, je před uložením do mailboxu `From` quotováno `>From` , tzv. vránu je tedy potřeba odstranit před uložením do `cnm` souboru. Script se nezabývá zamykáním mailboxu a jednoduše frontu přejmenuje na `*.lock`. Po zpracování je pošta na linuxu smazána. Script využívá toho, že Mars zakládá v poštovním adresáři adresář `user` s adresáři dle jmen uživatelů a linkuje je oproti adresářům `MAIL\cislo_uzivatele`. Dalším předpokladem úspěšného běhu je, že uživatelé mají na Novellu a v Linuxu stejná jména, a že vytvoříme link root->supervisor, nebo budeme poštu pro roota forwardovat jinému uživateli (klasickým souborem `~/.forward`). Vytvoření linků povolíme marsu v sekci 16, konfiguračního souboru Marsu, kde nastavíme hodnotu buď 1 nebo 2. Česká dokumenace k Marsu je na (4).

A teď již třešnička na dortu, jak to funguje celé dohromady. Nejdříve spustíme první script. Ten nám předá poštu od uživatelů do fronty, potom nahodíme `sendmail`, UUCP, ETRN či co vlastně používáme a poštu odešleme a přijmeme. A do třetice spustíme druhý script, ten nám poštu rozdělí mezi jednotlivé uživatele. První script není špatné pouštět pravidelně a často z `cronu`. Pmail totiž i lokální poštu, pokud obsahuje @ ukládá do fronty pro odesílání do Internetu.

A má to nějaké chyby? Jistě, nemohu tvrdit, že Pmail je naprosto bezchybný produkt, ale při běžném používání jich moc nenajdete. Pmailu pro Windows třeba nechutnají některé speciální případy MIME kódování, třeba BASE64 v hlavičkách, zato ale narozdíl od Netscape mailu umí odesílat hlavičky s diakritikou.

Za určitou závadu prezentovaného řešení lze považovat to, že k poště není nadále možné přistupovat přes POP3 a IMAP4 protokol. O podpoře pro POP3 se uvažuje, o IMAPu zatím ne, ale je zde samozřejmě možnost, že tento článek přiláká někoho, kdo by měl zájem s námi řešení dále rozvíjet a zdokonalovat. Kdo ví, třeba se jednou stane integrální součástí MARS NWE. Pokud máte pocit, že mluvím právě o Vás, určitě se nám ozvěte. ■

1 Pegasus Mail pro DOS
http://risc.ua.edu/pub/network/pegasus/pmail340.zip
2 Pegasus Mail Preview
ftp://risc.ua.edu/pegasus-previews/pmdos_pv.zip
3 Česká podpora pro Pegasus Mail
http://www.fee.vutbr.cz/~kuchta/pmail
4 Česká dokumentace k MARSu
http://www.spsselib.hiedu.cz/homedirs/~kerslage/manuals/linux/

Jak psát lokalizované programy

Michael Mráka a Vladimír Michl, 9. června 1998

Tento článek by měl přinést (pokud možno jednoduchý) návod, jak psát programy snadno použitelné v různých jazykových mutacích.

Prvním předpokladem pro správné chování takového programu je fungující lokalizace. Většina současných distribucí Linuxu je vystavěna nad knihovnami `glibc`, které by již s lokalizací neměly mít problémy. Zda je tomu skutečně tak, lze zkontrolovat pohledem do adresáře `/usr/share/locale`, který by měl obsahovat podadresář `cs_CZ*` (na mém počítači je to `cs_CZ.ISO-8859-2`) a v něm soubory `LC_COLLATE`, `LC_CTYPE`, `LC_MONETARY`, `LC_NUMERIC`, `LC_TIME` a adresář `LC_MESSAGES`. Pokud tomu tak není, ale v systému existuje alespoň definiční soubor `cs_CZ` (bývá standardně zahrnut do balíku `glibc` jako `/usr/share/i18n/locale/cs_CZ`), je možné lokalizační soubory pomocí příkazu

```
$ localedef -i cs_CZ -f ISO-8859-2 \
    cs_CZ.ISO-8859-2
```

vygenerovat. Pokud systém nevlastní ani uvedený definiční soubor, je vhodné začít například na adrese (1), kde najdete více informací :-).

Nyní přistupme k tvorbě vlastního programu Příklad lc_example.c (...). Nejprve je potřeba program přimět, aby používal lokalizaci; to provedeme pomocí funkce `setlocale(LC_ALL, "")` (viz funkce `init()`) — parametry uvedené v příkladu sdělí programu, aby použil nastavení určené proměnnými prostředí `LC_*` a `LANG`. Dále již stačí používat standardní knihovní funkce pro práci s lokalizovaným prostředím.

Pro jednotlivé kategorie jsou k dispozici následující funkce:

- `LC_COLLATE` — lexikografické třídění (viz funkce `lc_collate()`).
  - `int strcoll(const char *s1, const char *s2)` — porovnávání řetězců s ohledem na lokalizaci; syntaxe a návratové hodnoty odpovídají funkci `strcmp()`.
  - `size_t strxfrm(char *dest, const char *src, size_t n)` — transformace řetězce tak,

```
#include <locale.h>
#include <libintl.h>
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include <time.h>
#include <monetary.h>

#define LOCALEDIR "/usr/share/locale"
#define PACKAGE "lc_example"

#define BUF 80
#define _(str) gettext(str)

void init() {
   setlocale(LC_ALL,"");
   bindtextdomain (PACKAGE, LOCALEDIR);
   textdomain (PACKAGE);
}

void invite() {
   printf(_("Hello!\n"));
}

void lc_collate() {
#define N 5
   char *words[] = {"plaňka", "pláně", "Plášil", "cikáda", "chroust"};
   char *p;
   int i,j;

   for (i = 0; i < N; i++)
      for (j = 0; j < N; j++) {
         if (strcoll(words[i],words[j]) < 0) {
            p = words[i];
            words[i] = words[j];
            words[j] = p;
         }
      }
   printf(_("Sorted words:"));

   for (i = 0; i < N; i++)
      printf(" %s", words[i]);

   printf("\n");
}

void lc_time() {
   char buf[BUF];
   time_t t = time(NULL);

   strftime(buf, BUF, "%A", localtime(&t));
   printf(_("%s - I hate it!\n"), buf);
}
```

Výpis č. 3: Příklad lc_example.c (...)

aby porovnání dvou takto vzniklých řetězců pomocí `strcmp()` mělo stejný výsledek, jako porovnání původních pomocí `strcoll()`.

- LC_CTYPE — rozdělení znaků do tříd (malá a velká písmena, oddělovače, čísla, bílá místa, ...) (viz funkce `lc_ctype()`).
- `int isalpha (int c)` — písmena
- `int isascii (int c)` — 7-bitová unsigned char hodnota z ASCII.
- (podobně `iscntrl()`, `isdigit()`, `isgraph()`, `islower()`, `isprint()`, `ispunct()`, `isspace()`, `isupper()`, `isxdigit()`)

```
void lc_ctype() {
   char buf[BUF];
   int i;

   printf(_("Wrote some text, please:\n"));
   fgets(buf, BUF, stdin);

   for (i = 0; i < BUF && buf[i] != '\0'; i++) {
      if (isalnum(buf[i]))
         buf[i] = '+';
      else if (isspace(buf[i]))
              buf[i] = '_';
           else
              buf[i] = '-';
   }
   printf("%s\n", buf);
}

void lc_numeric() {
   printf(_("Numbers: %'d   %'f\n"),
      (int) 12345678, (float) 1234567.1235678);
}

void lc_monetary() {
   char buf[BUF];

   strfmon(buf, BUF, _("You have forgotten to pay %n to\
the author of this program.\n"), (double) 1234);
   printf("%s", buf);
}

main(int argc, char **argv) {
   init();
   invite();
   lc_collate();
   lc_time();
   lc_numeric();
   lc_ctype();
   lc_monetary();
}
```

Výpis č. 3: Příklad lc_example.c (pokračování)

- LC_TIME — časové údaje (viz funkce `lc_time()`).
  - `size_t strftime(char *s, size_t max, const char *format, const struct tm *tm)` — nahradí sekvence %x v řetězci `format` časovými údaji podle časové zóny a zvolené lokalizace. Např. %A = den v týdnu, %a = zkratka dne v týdnu, %B = měsíc, ...„Nahrazuje“ funkce `ctime()`, `asctime()`.
- LC_NUMERIC — formátování čísel (viz funkce `lc_numeric()`).
  - `int printf(const char *format, ...)` — pokud %-sekvence pro čísla obsahují apostrof (např. %'d) vytiskne oddělovač desetinných míst a tisícovek s ohledem na lokalizaci (nefunguje až v `libc.5`).
- LC_MONETARY — formátování peněžních informací (viz funkce `lc_monetary()`).
  - `ssize_t strfmon(char *s, size_t maxsize, const char *format, ...)` — nahradí sekvence %x peněžními údaji podle zvolené lokalizace. Ve formátovacím řetězci lze zadat: %i = mezinárodní symbol měny s peněžní částkou (CZK), %n = národní symbol měny s peněžní částkou (Kč). Peněžní částky musí být čísla typu `double` nebo `long double`. O této funkci se bohužel v info dokumentaci nedozvíte, navíc v `libc.5` tato funkce vůbec nebyla. Více informací naleznete v souboru `strfmon.man3`.

Pokud by pro některou z kategorií neexistovaly standardní funkce, je možné formátování udělat „ručně“ za pomocí struktury `lconv` získané z funkce `localeconv()`.

Samostatnou kapitolu pak tvoří překlad (nejen chybových) zpráv programu. V systémech s `libc.6` (`glibc`) se provádí pomocí balíku `gettext`. (Ve starších systémech s `libc.5`, případně na jiných operačních systémech se k tomuto účelu používají funkce `gencat`; více viz (2).)

To, že program má používat katalog s českými překlady zpráv (kategorie LC_MESSAGES), už akceptoval při volání `setlocale()`; je ovšem potřeba ještě specifikovat jméno katalogu. To uděláme na začátku programu voláním funkcí `bindtextdomain(package, localedir)` — cesta ke katalogu — a `textdomain(package)` — jméno katalogu, který používáme (viz funkce `init()`). Všechny řetězce, které mají mít svůj ekvivalent v katalogu, je nutné „prohnat“ funkcí `gettext()` (viz funkce `invite()`).

Ještě zbývá napsat české překlady zpráv. Nejprve je potřeba pomocí programu `xgettext` vygenerovat katalog obsahující všechny originální zprávy. Ve vzorovém programu byl k tomu použit příkaz

```
$ xgettext --default-domain=lc_example \
--output-dir=. --add-comments
--keyword=_ lc_example.c
```

kde

- `--default-domain=lc_example` – ulož výsledek do souboru lc_example.po
- `--output-dir=.` – cílový adresář
- `--add-comments` – napiš, kde se jednotlivé řetězce vyskytují
- `--keyword=_` – místo gettext použij klíčové slovo _

Nyní lze ke každému řetězci připsat za klíčové slovo `msgstr` příslušný překlad. Pokud by váš katalog byl větší je velmi výhodné používat na překlad Emacs v tzv. po-módu (potřebný kód do Emacsu je přiložen v balíku `gettext`). Hlavní výhoda je, že po-mode umí na požádání zobrazit přímo místo zdrojového kódu, kde se zpráva nachází (klávesa `s`). Dále umožňuje kontrolu, zda jsou zprávy dobře přeloženy (zda sedí počet formátovacích značek, zda zprávy začínají a končí novým řádkem stejně), automatické vyplnění úvodní hlavičky i s datem revize (vše `Shift-v`). Samozřejmě editovat překlad lze pomocí klávesy `Enter`, ukončení `Ctrl-c Ctrl-c`. Pokud se vám v katalogu objeví zpráva označená jako „fuzzy“, pak by tato zpráva měla být neúplně přeložena. Toto označení lze odstranit klávesou `Tab`. Zprávu lze jako „fuzzy“ označit pomocí `BackSpace`.

Pro instalaci po-módu je třeba umístit soubor `po-mode.el` (`po-mode.elc`) do adresáře `/usr/share/emacs/site-lisp` a následně do souboru `$HOME/.emacs` přidat:

```
(setq auto-mode-alist
  (cons '("\\.po[tx]?\\'\\|\\.po\\." . po-mode)\
  auto-mode-alist))
(autoload 'po-mode "po-mode")
```

Máme tedy přeložen celý katalog a pomocí

```
$ msgfmt -v -o lc_example.mo lc_example.po
```

ho přeložíme do binární podoby. Pak už stačí jen přesunout `lc_example.mo` do adresáře `/usr/share/locale/cs_CZ*/LC_MESSAGES`, zkompilovat program a vyzkoušet.

```
$ gcc -o lc_example lc_example.c

$ LC_ALL=cs_CZ.ISO-8859-2 ./lc_example
Dobrý den!
Setříděná slova: cikáda chroust pláně \
plaňka Plášil
Úterý - to nesnáším!
Čísla: 12 345 678   1 234 567,125000
Napište nějaký text, prosím:
Umíš česky, ježku?
++++_+++++-_+++++-_
Zapomněl jste zaplatit  1 234,00Kč autorovi \
tohoto programu.

$ LC_ALL=en_US ./lc_example
Hello!
Sorted words: chroust cikáda Plášil \
pláně plaňka
Tuesday - I hate it!
Numbers: 12,345,678   1,234,567.125000
Wrote some text, please:
Umíš česky, ježku?
++--_-++++-_++-++-_
You have forgotten to pay  $1,234.00 to the \
author of this program.
```

Je vidět, že funkce `strfmon()` zatím není dokonalá (zbytečná mezera na začátku, mezi částkou a Kč není mezera, ač by podle definice v souboru `cs_CZ` být měla), stejně jako funkce `printf()` (desetinná čísla nejsou dělena do skupin). Časem se ale tyto chyby určitě vyladí.

A na závěr ještě jedna užitečná funkce, která se může při lokalizaci hodit a to

```
#include <langinfo.h>
char *nl_langinfo(nl_item item);
```

Tato funkce vrátí hodnotu, kterou po ní požadujeme – např: první den v týdnu `nl_langinfo(DAY_1)` nebo první měsíc v roce `nl_langinfo(MON_1)`. Argumenty které můžete požadovat naleznete v souboru `/usr/include/langinfo.h`. ■

1 Definiční soubory pro české locales
ftp://ftp.fi.muni.cz/pub/localization/locale
2 Czech-HOWTO
ftp://ftp.fi.muni.cz/pub/linux/local/czech-howto/

Vytváření RPM balíků

Jan Kasprzak, 12. června 1998

V dnešním díle našeho seriálu budu pokračovat ve výkladu problematiky tvorby RPM balíků. Nejprve doplním o některé podrobnosti syntaxi příkazu `rpm -b`, a pak začnu detailněji rozebírat strukturu spec-souboru.

rpm -ba — detaily

V předchozím čísle jsem uvedl základní způsoby použití příkazu `rpm -b` pro stavbu RPM balíků. Nyní uvádím další, méně užívané volby:

- `--buildarch` *architektura* — vytvoří RPM balík pro danou architekturu místo architektury, na které právě systém běží. Lze použít zejména při cross-kompilaci.
- `--buildos` *operační systém* — totéž, ale změní operační systém, pro který bude balík určen.
- `--test` — vytvoří skripty pro jednotlivé fáze kompilace (uloží je do adresáře `/var/tmp`) a skončí. Tvůr-

ce balíku pak má možnost si tyto skripty prohlédnout a případně i editovat.

- `--buildroot` *adresář* — pro instalaci balíku se použije tento adresář, místo adresáře uvedeného v hlavičce `BuildRoot:` ve spec-souboru.

Uvnitř spec-souboru

Zaměřme se nyní na syntaxi spec-souboru, který řídí celou výstavbu RPM balíku. Jedna z věcí, kterou spec-soubor může obsahovat, je komentář. Komentáře zde začínají znakem křížek (#) na začátku řádku a pokračují do konce řádku. Komentáře jsou programem `rpm` ignorovány.

Další částí je hlavička. Syntaxe se trochu podobá hlavičce internetové zprávy podle RFC 822: Hlavička je umístěna na začátku spec-souboru a obsahuje položky ve formě řádků. Každý řádek definuje hodnotu jednoho *tagu* (česky atributu? Asi ne.) a má tuto syntaxi:

tag`:`*hodnota*

U jména tagu se nerozlišují malá a velká písmena a kolem dvojtečky mohou být bílé znaky (mezery, tabulátory, ...). Na druhé straně syntaxe hodnoty tagu závisí na konkrétním tagu a obvykle se rozlišuje velikost písmen. Některé tagy mohou mít jen číselné hodnoty, některé nesmějí obsahovat určité znaky (třeba pomlčku), atd. Nyní uvedu, které tagy systém RPM rozpoznává, a jak je použít.

Tagy pro pojmenování balíků

- `name` — název softwaru, který se balí do balíku. Doporučuje se, aby tento tag byl stejný (včetně velikosti písmen), jako u zdrojového archívu příslušného softwaru. Příklady: `gcc`, `ImageMagick`.
- `version` — verze softwaru (to jest verze specifikovaná autorem softwaru). Příklady: `2.7.2.1`, `19980302beta`.
- `release` — verze RPM balíku daného softwaru. Často lze pomocí release rozlišit verze RPM balíku pro různé distribuce nebo různá prostředí. Takže pro secure shell s RSAREF knihovnou může být release `2us`, čili pro použití ve Spojených státech. Mezinárodní verze balíku pak může mít tento tag nastavený na hodnotu `2i`.

Popisné tagy

- `summary` — jednořádková informace o balíku a jeho použití. Již dříve jsem uvedl, že podrobnější informaci lze nalézt v sekci `%description` příslušného spec-souboru. Příklad: `Secure Shell -- encrypts network communications.`
- `copyright` — krátká informace o podmínkách šíření balíku. Nejde o kompletní licenci, ale o její shrnutí nebo název. Příklady: GPL, BSD, `distributable`, `postcardware`.
- `distribution` — název distribuce, do které balík patří. Příklady: `Red Hat Linux Rembrandt`, `Red Hat Power Tools 5.0`.
- `icon` — jméno souboru, ve kterém je uložena ikona balíku. RPM systém tento tag nepoužívá, ale může být použit například grafickým správcem balíků, jako je například `glint`. Jde o grafický soubor například ve formátu `xpm` nebo `gif`.
- `vendor` — název organizace, která je zodpovědná za vznik tohoto balíku. Příklad: `Red Hat Software, Inc.`.
- `url` — domovská stránka balíku. RPM tento tag nepoužívá, slouží pouze uživatelům balíku jako odkaz na další dokumentaci případně novější verze softwaru.
- `group` — tematická skupina, do které tento balík patří. Seznam všech platných skupin i jejich podskupin lze nalézt v souboru `groups` v dokumentačním adresáři RPM (obvykle `/usr/doc/rpm-`*verze*). Příklady: `Libraries`, `X11/Games/Strategy`.
- `packager` — jméno a kontaktní informace (e-mailová adresa, telefonní číslo) člověka, který konkrétní balík vyrobil (případně adresa technické podpory, jde-li o větší firmu).

Závislosti mezi balíky

- `serial` — sériové číslo softwaru. Pro potřeby instalace nových verzí balíku (upgradování), ale i pro závislosti mezi balíky (kdy balík může vyžadovat například „balík `pam` verze novější než *x.y.z*“), je potřeba určit, která verze softwaru je novější než jiná. V některých případech je však „oficiální“ číslování daného softwaru natolik kryptické, že není možné algoritmicky určit, které označení znamená novější verzi. Proto je možné RPM balíku přiřadit sériové číslo a toto číslo používat při (časovém) porovnávání verzí namísto skutečného označení verze. Poznamenávám ještě, že balíky, obsahující tag `serial` se považují vždy za novější než balíky bez tohoto tagu.
- `provides` — jméno jednoho nebo více „virtuálních balíků“, které příslušný balík poskytuje. Některé balíky totiž nepotřebují ke své činnosti jiný konkrétní balík, ale určitý typ balíku. Například programy pro doručování pošty mohou vyžadovat existenci programu pro čtení pošty. A je lhostejné, jde-li o `exmh`, `mutt` nebo třeba `gnus`. Takový program pro doručování pošty pak vyžaduje *virtuální balík* s názvem (například) `mail-reader`. A každý program pro čtení pošty bude mít ve svém spec-souboru uveden tag `Provides: mail-reader`.
- `requires` — seznam balíků, které náš balík vyžaduje ke korektní činnosti. Některé závislosti jsou generovány automaticky (zejména závislosti na sdílených knihovnách a na interpreterech skriptů), jiné lze specifikovat tímto tagem. Příklad: `requires: pam >= 0.51` říká, že ke správnému provozu daného balíku musí být v systému balík `pam` verze aspoň `0.51`. Je také možno vyžadovat verzi balíku podle sériového čísla: `requires: playmidi =S~4` vyžaduje jmenovaný balík sériového čísla právě 4. Může zde kromě jména balíku být uveden i jen soubor, který je v systému vyžadován pro korektní činnost balíku. Tento způsob závislostí se specifikuje řetězcem, který začíná lomítkem. Například poštovní klient, odesílající poštu na standardní vstup programu `sendmail` v podstatě nepotřebuje balík `sendmail`, ale stačí jakýkoli balík, který poskytne program

/usr/sbin/sendmail. Do spec-souboru se pak napíše requires: /usr/sbin/sendmail.

- conflicts — seznam balíků, které jsou konfliktní (to jest nemohou být v systému instalovány zároveň) s tímto balíkem. Tento tag není příliš často používán, protože většina konfliktů je detekována systémem RPM už z toho důvodu, že navzájem konfliktní balíky obsahují soubory stejného jména a tedy není možné je přímo a zároveň nainstalovat. Příklad conflicts: inn může být uvedeno ve spec-souboru news serveru LeafNode, nechce-li tento být instalován zároveň se serverem inn.
- autoreqprov — zapíná/vypíná automatické generování položek do seznamu requires (sdílené knihovny ze spustitelných programů a interpretery skriptů) a do seznamu provides (.so-jména všech sdílených knihoven, zabalených v tomto balíku). Povolené hodnoty jsou yes a no.

Tagy architektury a operačního systému

- excludearch — upozorní RPM, aby se nepokoušel kompilovat balík na vyjmenovaných architekturách, protože je známo, že na těchto platformách daný balík nefunguje. Příklad: není-li balík schopen běžet na 64-bitových architekturách, můžeme napsat excludearch: alpha sparc64.
- exclusivearch — říká, že balík může být postaven pouze na vyjmenovaných platformách. Například balík dosemu může běžet jen na procesorech x86, tedy exclusivearch: i386 je na místě.
- buildarchitecture — specifikuje architekturu, pro kterou se balík má vytvářet, bez ohledu na to, na které platformě momentálně běží program rpm, který jej vytváří. Tento tag se používá zejména pro tvorbu balíků, nezávislých na architektuře. Například balík s fonty, které jsou pricipiálně přenositelné mezi platformami, může obsahovat tag buildarchitecture: noarch. Při použití rpm -ba na takovýto spec-soubor na libovolné platformě se vytvoří balík s koncovkou noarch.rpm.
- excludeos — totéž jako excludearch, ale týká se operačního systému místo architektury. Příklad: balík WINE by klidně mohl mít excludeos: windows95 :-)
- exclusiveos — analogie k exclusivearch. Příklad: balík util-linux jistě může mít exclusiveos: linux.
- buildos — analogie k buildarchitecture.

Adresáře

- prefix — používá se při tvorbě *relokovatelných* balíků (tj. nezávislých na umístění). Je-li tento tag uveden, musí každá cesta v sekci %files začínat daným adresářem. Relokovatelný balík pak správce může instalovat do adresáře podle své potřeby (například zvolit mezi /usr/local a /opt). Vytvořit relokovatelný balík ovšem není tak úplně jednoduché – samotný software musí být nezávislý na umístění. Nesmí tedy například obsahovat zakompilovanou cestu ke konfiguračním souborům, PID-souborům a podobně.
- buildroot — s tímto tagem jsme se setkali. Navzdory svému jménu neoznačuje adresář, ve kterém probíhá kompilace, ale adresář, do kterého skript %install uloží zkompilované soubory. Je dobré mít tento tag v balíku, aby i běžný uživatel mohl provádět jeho rekompilaci. Typicky se zde uvádí jméno ve stylu /var/tmp/*balík*-root. Bez tohoto tagu by běžnému uživateli selhala sekce %install, jelikož běžný uživatel pravděpodobně nemůže zapisovat do adresářů jako /usr/bin nebo /usr/doc. Zároveň s definicí tohoto tagu je ovšem nutno upravit skript %install tak, aby instaloval do zde specifikovaného adresáře. Zde již tedy nestačí prosté make install. Často ale změna není až tak složitá. Někdy postačuje příkaz typu make PREFIX=$RPM_BUILD_ROOT/usr install.

Zdrojové soubory a záplaty

- source — jméno nebo URL, odkazující na zdrojový archív, ze kterého se daný balík vytváří. Tagů source lze uvést i více (další se pak jmenují source1, source2 atd. Místo source lze také použít logický ekvivalent source0. Program rpm z hodnoty tohoto tagu bere v úvahu pouze část za posledním lomítkem. Soubor tohoto jména se hledá v adresáři /usr/src/redhat/SOURCES. URL nebo cesta slouží pouze jako odkaz na místo, odkud byl zdrojový archív získán. Příklad: balík ssh má tyto zdrojové soubory:

```
Source0: ftp://ftp.cs.hut.fi/pub/ssh/\
ssh-1.2.25.tar.gz
Source1: ftp://ftp.funet.fi/pub/crypt/mirrors/...
Source2: sshd.init.rh50
Source3: ssh.pam
Source4: ftp://ftp.cs.hut.fi/pub/ssh/\
ssh-1.2.25.tar.gz.sig
```

Zde vyjmenované soubory mohou, ale nemusejí být použity ke tvorbě binárního RPM balíku, každopádně ale jsou zařazeny do zdrojového RPM balíku (to je třeba případ výše uvedeného source4, který obsahuje PGP podpis zdrojového archívu a je distribuován pouze pro usnadnění ověření pravosti tohoto archívu.

- patch — odkazuje záplatu, používanou při sestavování daného softwaru. Podobně jako u tagu source i záplat může být více a jsou označeny tagy patch0 (což je ekvivalent tagu patch), patch1, patch2 a tak dále.
- nosource — seznam čísel zdrojových souborů, které nemají být zařazeny do src.rpm balíku. Pokud licence daného softwaru nepovoluje šíření zdrojové podoby balíku v jiné než původní podobě, nebo pokud z jiného důvodu nechceme některý ze zdrojových archívů zahrnout do zdrojového RPM souboru, napíšeme jeho číslo do tagu nosource. Pokud si pak uživatel chce postavit binární RPM soubor, stačí příslušný zdrojový archív uložit do /usr/src/redhat/SOURCES a spustit rpm --rebuild na příslušný src.rpm soubor. Příklad: pokud by například RPM balík pgp byl distribuován z USA, podle tamějších zákonů by se vlastně mohl

distribuovat jen předpis ke kompilaci — zdrojový RPM soubor. Použily by se tyto tagy:

```
Source0: ftp://.../pgp50.tar.gz
Source1: ftp://.../rsaref.tar.gz
NoSource: 0 1
```

- `nopatch` — analogie k `nosource` pro soubory se záplatami.

Takto tedy vypadají jednotlivé položky hlavičky spec-souboru. V příští části se podrobněji zaměříme na další sekce tohoto souboru a na jednotlivé fáze tvorby RPM balíku. ■

Zasmáli jsme se!

Pavel Janík ml., 12. června 1998

Ano, i v červnu jsme se smáli. Já osobně například při horlivé diskusi na téma zda diakritika v konferenci *linux@muni.cz* ano či ne, zda se pokusíme změnit uznávanou normu, zda má či nemá při bootování Linuxu na obrazovce mrkat tučňák, jestli má kernel mít něco jako `/proc/config`, který bude obsahovat aktuální konfiguraci kernelu, kdy už Linus konečně vydá další pre-patch a kdy už konečně bude opravena chyba při `make menuconfig` a zvuk v kernelu (*linux-kernel@vger.rutgers.edu*), jestli má pdfTEX podporovat i jiné grafické formáty než PNG a JPG apod.

Máme to vůbec zapotřebí se takhle hádat? Já jsem již zjistil, že nikoli a vždy místo toho, abych podobné „války slov“ četl a zaplétal se tak do diskusí, raději něco udělám. Někdy už ale tyto nesmysly vydržeti nemohu a potom se opět směju, ale tentokráte již sám sobě. Jsem vlastně stejný jako oni… Ach jo.

Objevil jsem zajímavý obrázek. Uhádnete, jak se jmenuje tučňák na obrázku? Pokud ne, odpověď naleznete na adrese (1).

V minulém čísle Linuxových novin jsme vyhlásili malou soutěž. Jestlipak si pamatujete, jak zněla soutěžní otázka? Pokud ano, čtěte dále, pokud ne, přečtěte si minulé číslo. Shromáždilo se mi několik odpovědí, z těch nejvtipnějších vybírám:

> Co se stane, když Microsoft uvolní zdrojové kódy OS Windows?
> Hackeři odstraní nestabilní část a Billovi bude zase stačit 640kB.
> Miro Bobovsky

> Co se stane? To je jasný:
> Většina programátorů Microsoftu si bude muset najít úplně jiné zaměstnání, protože jako programátory už je nikdo nezaměstná.
> Texty budou vydány knižně a zapsány do Guinnessovy knihy rekordů jako nejdelší vtip na světě.
> Petr Zima

> Co se stane?
> Bohužel tím asi zlikvidují většinu vývojářů otevřených systémů, protože ti si buď:
>
> - zlomí vaz od vehementního kroucení hlavy
> - uchechtají se k smrti
> - upláčou se k smrti
> - nebo prasknou vzteky, když najdou kousky svých vlastních kódů naiplementovaných do MS Windows zcela nefunkčním způsobem…
>
> Pavel Martak

Posledním zajímavým příspěvkem je neobvyklý soubor, který najdete při Linuxových novinách. Jmenuje se `CoSeStane.mpg` a jedná se o Mpeg Video. Prosím prohlédněte si jej a radujte se, vždyť na světě je tolik krásných věcí (pravda, co může být hezčího než Linux 2.1.106 na Quad Pentiu II ;-). Tento příspěvek, či spíše jeho autor — v naší soutěži zvítězil. Prosím jej, aby mi napsal. ■

1 Tučňák
`http://www.chalice.gen.nz/linpic/`

Linuxové noviny a jejich šíření

Linuxové noviny vydává České sdružení uživatelů operačního systému Linux (1) pro své příznivce a sympatizanty.

Linuxové noviny jsou otevřeny každému, kdo by chtěl našim čtenářům sdělit něco zajímavého. Příspěvky (ve formátu čistého textu v kódování ISO 8859-2) posílejte na adresu (2). Autor nemá nárok na finanční odměnu a souhlasí s podmínkami uvedenými v tomto odstavci. Vydavatelé si vyhrazují právo rozhodnout, zda Váš příspěvek uveřejní, či nikoli.

Chtěl bych poděkovat Fakultě informatiky Masarykovy university v Brně, INET, a.s., Juraji Bednárovi a společnosti OptiCom za poskytnutí diskového prostoru pro Linuxové noviny.

Linuxové noviny můžete najít na akademické síti CESNET (3), na síti IBM Global Network na adrese (4), na serveru časopisu Netáčik (5), který je připojen do slovenského SIXu, případně na serveru společnosti OptiCom (6).

Linuxové noviny jsou k dispozici také ve formátu HTML na adrese (7). ■

1 České sdružení uživatelů operačního systému Linux
http://www.linux.cz/czlug
2 Adresa redakce
mailto:noviny@linux.cz
3 Linuxové noviny na síti CESNET
ftp://ftp.fi.muni.cz/pub/linux/local/noviny
4 Linuxové noviny na síti IBM Global Network
ftp://ftp.inet.cz/pub/People/Pavel.Janik/noviny
5 Slovenské zrcadlo Linuxových novin
ftp://netacik.sk/pub/linux/cz-noviny
6 Linuxové noviny – OptiCom
http://www.mathew.sk/noviny
7 Linuxové noviny ve formátu HTML
http://www.linux.cz/noviny

Šéfredaktor: Pavel Janík ml.
mailto:Pavel.Janik@linux.cz
sazba: Ondřej Koala Vácha
mailto:koala@informatics.muni.cz
jazykové korekce: Bohumil Chalupa
mailto:bochal@met.mff.cuni.cz
překlady: Hanuš Adler
mailto:had@pdas.cz
převod do HTML: Pavel Juran
mailto:xjuran@cs.felk.cvut.cz