Spis HOWTO za začetnike v Emacsu

Jeremy D. Zawodny: *jzawodn@wcnet.org*, prevod v slovenščino Aleš Košir: *ales.kosir@hermes.si* 14. oktober, 1999 ($Različica: 1.7 $)

Ta spis seznanja uporabnika operacijskega sistema Linux z urejevalnikom Emacs. Predpostavlja, da uporabnik pozna urejevalnik `vi` ali podobnega. Zadnja različica tega spisa je na voljo v spletu na naslovu *http://www.wcnet.org/jzawodn/emacs/*.

Kazalo

1 Uvod

1.1 Copyright

1.2 Ciljna skupina in namen

Ta spis je namenjen uporabniku sistema Linux, tistemu, ki bi se rad naučil uporabljati urejevalnik Emacs. Spis je začel nastajati kot povzetek za kratko predstavitve, ki sem jo pripravil za toleško območno srečanje uporabnikov Linuxa (Toledo Area Linux User Group) *http://www.talug.org/*. Spis je kasneje zrasel zaradi pomoči skupnosti, kakor kaže razdelek z zahvalami.

Posebej moram poudariti, da ni v spisu skoraj nič, kar bi veljalo le za Linux. Povedano se nanaša na vse vrste Unixov in celo na Emacs, ki teče v okolju Microsoft Windows. A ker je ta spis del linuxovskega dokumentacijskega projekta (Linux Documentation Project), trdim, da je spis pripravljen za uporabnike Linuxa, ker je bil prav njim resnično namenjen.

Tisti izmed bralcev, ki imate ime GNU/Linux raje kot preprosto „Linux" (preberite *http://www.gnu.org/gnu/linux-and-gnu.html*, da boste zvedeli, zakaj bi to kdo rad razločeval) pa ste vabljeni, da v mislih z nizom GNU/Linux nadomestite vse besede Linux v tem spisu. Četudi ne nasprotujem utemeljitvi in duhu zamisli, ki je v ozadju, ne čutim potrebe, da bi zapisal „GNU/Linux".

1.3 Kaj je Emacs?

Emacs pomeni različnim ljudem različne stvari. Odvisno od tega, koga boste vprašali, boste morebiti dobili naslednje odgovore:

- urejevalnik besedil,
- odjemalec za elektronsko pošto,
- pregledovalnik novičarskih skupin,
- program za stavljenje besedila,
- vero,
- integrirano razvojno okolje,
- karkoli bi vi želeli, da je!

Za naš namen se pretvarjajmo, da je Emacs urejevalnik besedila, in sicer neverjetno zmogljiv urejevalnik besedila, kasneje pa bomo globlje obravnavali to vprašanje. Emacs je napisal Richard Stallman (ustanovitelj fundacije za prosto programje, Free Software Foundation *http://www.fsf.org/*, in projekta GNU *http://www.gnu.org/*) in ga še danes vzdržuje.

Emacs je eden od najbolj razširjenih in zmogljivih urejevalnikov v Linuxu (in Unixu) in je po priljubljenosti takoj za urejevalnikom **vi**. Slovi po ogromnem naboru funkcij, zmožnosti preprostega prilagojevanja in po tem, da nima napak.

Obsežni nabor funkcij in zmožnost prilagojevanja sta posledica načina, na katerega je bil Emacs načrtovan in izveden. Ne da bi se spustil v podrobnosti, trdim, da Emacs ni **le** urejevalnik. Velik del ga je napisanega v programskem jeziku **lisp**. Jedro Emacsa je v jeziku C napisan popolnoma funkcionalen tolmač za jezik lisp in le najbolj osnovni in nizkonivojski del predstavlja prevedena koda C, večina Emacsove kode pa je napisana v jeziku lisp. Trdimo lahko, da je v Emacs vgrajen celotni programski jezik, njegovo kodo pa lahko po potrebi spreminjamo, prilagajamo, dodajamo in tako prikrojujemo Emacsovo obnašanje.

Emacs je hkrati tudi eden od najstarejših urejevalnikov. Dejstvo, da ga zadnjih dvajset let (?) uporabljajo tisoči programerjev, je razlog, da je zanj na voljo veliko dodatnih paketov. Ti paketi omogočajo, da vi postorite stvari, o katerih se Stallmanu, ko je Emacs začel pisati, še sanjalo ni. Več o tem pa v nadaljevanju.

Na voljo je mnogo spletnih mest in spisov, ki dajo boljši pregled Emacsa, njegove zgodovine in podobnega od tega, ki je pred vami. Raje kot da bom poskušal zajeti čimveč znanega, svetujem, da si ogledate nekaj mest, ki jih navajam v razdelku 6 (Drugi viri).

1.3.1 Prilagoditve in različice

Potrebno je omeniti, da obstajata dva različna urejevalnika: GNU Emacs in XEmacs. Oba izvirata iz skupnega prednika in večina funkcij je skupna. To besedilo se nanaša na GNU Emacs (izrecno na različico 20.3), a večina zapisanega velja tudi za XEmacs in zgodnejše različice GNU Emacs. V tem besedilu se sklicujem preprosto na „Emacs”, kar upoštevajte.

1.3.2 Kako pridobimo Emacs?

Emacs zlahka pridobimo. Če uporabljate razširjeno distribucijo Linuxa, kakršna je Debian, RedHat, Slackware ali podobna, je Emacs zelo verjetno že priložen kot paket, ki ga namestimo z našega nosilca za distribucijo. Če Emacsa nimamo, lahko dobimo tudi njegovo izvorno kodo in jo sami prevedemo. Obiščite spletno mesto *http://www.gnu.org/software/emacs/emacs.html*, kjer boste našli točna navodila, kako ga pridobite.

2 Uporaba Emacsa

2.1 Zagon in zapustitev Emacsa

Kot novi uporabnik Emacsa ga bomo sprva verjetno hoteli pognati in preskusiti. A ko smo enkrat v njem in ga želimo zapustiti, tega verjetno sprva ne bomo znali. Če ga nismo še nikoli uporabili, ga poskusimo pognati prav zdaj. V svoji ukazni lupini za pozivnikom vpišimo `emacs` in pritisnimo tipko Enter. Emacs bi se moral zagnati. Če se ne zažene, bodisi ne leži v poti do programov bodisi ni nameščen.

Ko pa smo uspešno pognali Emacs, se moramo naučiti, kako ga zapustimo. Vtipkati moramo ukaz `C-x C-c`. Pisava `C-x` pomeni, da pritisnemo tipko `Ctrl`, jo pridržimo, nato pritisnemo še tipko `x`, zatem pa popustimo `x` in še `Ctrl`. V zgornjem primeru moramo za prvim delom pritisniti in pridržati še tipko `Ctrl` in pritisniti `c`, da zapustimo urejevalnik.

Morebiti se nam bodo sprva zdeli Emacsovi ukazi čudni, nenavadni in celo nerodni, posebej če smo vajeni urejevalnika `vi`. V nasprotju z urejevalnikom `vi` Emacs nima posebnih načinov za vnašanje besedila in ukazov.

Na kratko: ukaz `emacs` požene urejevalnik Emacs. Z Emacsovim ukazom `C-x C-c` urejevalnik zapustimo.

2.1.1 Kaj vidimo?

Ko poženemo Emacs, njegov izpis zasede vse okno X ali zaslon, če ga poganjamo v konzoli. Na vrhu opazimo menujsko letev, v srednjem delu nekaj besedila in nekaj vrstic na dnu.

Zaslon je podoben tej skici z znaki:

```
+----------------------------------------------------------------------+
|Buffers Files Tools Edit Search Mule Help                             |
|                                                                      |
|Welcome to GNU Emacs, one component of a Linux-based GNU system.      |
|                                                                      |
|                                                                      |
|                                                                      |
| ...                                                                  |
|                                                                      |
|---1:---F1  *scratch*         (Lisp Interaction)--L1--All-------------|
|For information about the GNU Project and its goals, type C-h C-p.    |
+----------------------------------------------------------------------+
```

Različica v slovenskem jeziku pa je takšna:

```
+----------------------------------------------------------------------+
|Buffers Files Tools Edit Search Mule Help                             |
|                                                                      |
|Dobrodošli v GNU Emacs, sestavnem delu sistema GNU v Linuxu.          |
|                                                                      |
|                                                                      |
|                                                                      |
| ...                                                                  |
|                                                                      |
|---1:---F1  *scratch*         (Lisp Interaction)--L1--All-------------|
|Pritisnite C-h C-p, da dobite opis projekta GNU in njegovega namena.  |
+----------------------------------------------------------------------+
```

Opomba: Emacs običajno zavzame celotno okno ali zaslon. Zgornjo skico sem okrnil, da bi jo lahko prikazal na omejenem prostoru. Ko prvič poženemo Emacs, bomo opazili pozdravno sporočilo. Da bi prihranil prostor, sem njegovo vsebino izpusti in jo nadomestil z znaki „...”. Pozdravno sporočilo navaja različico Emacsa, ki ga uporabljamo, in nas vabi, da si ogledamo dostopno pomoč na zvezi.

Menujska letev Vrhnja vrstica v Emacsovem vmesniku je menujska letev (ang. menu bar). Če poganjamo Emacs v oknih X, se bodo izbire v menuju obnašala kot običajni potezni menuji (ang. pull-down menu), do katerih pridemo z miško. Drugače pa moramo za dostop do menujev uporabiti tipkovnico z ustreznimi bližnjicami tipk, o čemer tu ne govorimo.

Statusna vrstica in pogovorni vmesnik Zgornja od zadnjih dveh vrstic Emacsovega vmesnika je statusna letev (ang. status bar). Vsebuje podatke o vmesniku, v katerem delamo, o načinu dela v trenutnem vmesniku in podobne koristne informacije. Zaenkrat se zadovoljimo s tem, da se zavedamo obstoja te letve.

Najspodnejša vrstica se imenuje **pogovorni vmesnik** (ang. mini-buffer). Od osrednjega vmesnika (ang. main buffer) ga na zaslonu loči pravkar opisana statusna letev. Pogovorni vmesnik si lahko zamišljamo kot Emacsovo ukazno vrstico. Vanj vnašamo ukaze, ki jih dajemo Emacsu, in v njem nam Emacs izpisuje svoja sporočila in odgovore na naše ukaze.

Ugotovili boste, da tistemu, kar sem pravkar imenoval statusna letev (ang. status bar), Emacsova dokumentacija pravi načinovna vrstica (ang. mode line). To je mesto, kjer Emacs prikazuje podatke o načinu ali načinih dela v trenutnem vmesniku kot tudi trenutni datum in čas, zaporedno številko vrstice, velikost datoteke, torej skoraj vse, kar bi si želeli videti v tej vrstici.

2.2 Nekaj o terminologiji

Ta razdelek obravnava najbolj osnovno terminologijo Emacsa. Potrebno jo je poznati, de bomo lahko brali o Emacsu in ga uporabljali.

2.2.1 Vmesniki in datoteke

Za razliko od nekaterih drugih urejevalnikov datoteka, ki jo odpremo v Emacsu, ne ostane odprta ves čas dela. Namesto tega jo Emacs prebere v **vmesnik** (ang. buffer) v pomnilniku. Ko urejamo datoteko, se njena vsebina na disku ne spremeni, dokler vsebine vmesnika na shranimo na disk. Tak način dela ima svoje prednosti in slabosti, a trenutno je pomembno le, da razumemo ta pristop.

Posledica tega je, da to isto besedo uporabljamo v Emacsovi dokumentaciji, načinih, paketih in tako naprej. Pomeni nam trenutno vsebino datoteke v pomnilniku. Eh, potrebno je omeniti, da vmesnik ni nujno povezan s kakršnokoli datoteko na disku. Velikokrat Emacs ustvari vmesnik, v katerega bo shranil izhod ukazov, ki smo jih izvedli. Nekateri vmesniki bodo vsebovali izpis ukaza, seznam izbir, med katerimi lahko izbiramo, in podobno.

2.2.2 Točka in območje

V Emacsovem žargonu bomo velikokrat slišali za **točko** (ang. point) ali o njej brali. V splošnem je točka utripalka (ang. cursor). Natančen razloček med točko in utripalko na začetku verjetno ni odločilen. A če ste radoveni, si predstavljajte razliko takole: utripalka je vidni prikaz točke. Utripalka je vselej vidna na določenem mestu trenutnega vmesnika. Točka pa nasprotno prebiva v prostoru *med* dvema znakoma trenutnega vmesnika. Če vidimo utripalko na črki ‚o' besede „zob", to pomeni, da je točka v resnici med črkama ‚z' in ‚o'.

Emacs omogoča urejanje vmesnika kot mnogi sodobni urejevalniki z naslednjimi funkcijami: zamikanje (ang. indent), črkovanje (ang. spell-check), preoblikovanje (ang. reformat), izrezovanje (ang. cut), prepisovanje (ang. copy), lepljenje (ang. paste) in podobno na delu besedila trenutnega vmesnika. Označimo (ang. mark) lahko besedilo s poudarjanjem (ang. highlight), tako da uporabimo miško, in nato izvedemo ukaz le na izbranem območju besedila. Tak del besedila se v Emacsu imenuje **območje** (ang. region).

2.2.3 Okna

Res je, nadaljevanje bo zmedlo marsikoga, ki je že uporabljal grafični vmesnik. Spomniti pa se moramo, da je bil Emacs razvit veliko *preden* so postali popularni grafični vmesniki in okenski upravljaniki.

Okno (ang. window) v Emacsu je območje na zaslonu, v njem pa je prikazan vmesnik. Ko Emacs prvič poženemo, dobimo na zaslonu le eno okno. Nekateri ukazi (denimo pomoč ali dokumentacija) v Emacsu odpro dodatna začasna okna na Emacsovem zasonu.

Emacsovo okno nima nič skupnega z okni X v smislu grafičnega vmesnika. Odpremo lahko dodatna okna X, v katerih prikažemo Emacsove vmesnike, denimo za vzporedno primerjavo vsebine dveh datotek, ta nova okna X pa se imenujejo v Emacsovem žargonu **okvirji**. Nadaljujte z branjem!

2.2.4 Okvirji

V Emacsu je **okvir** (ang. frame) ločeno okno X, ki prikazuje nek Emacsov vmesnik. A vsa so del iste Emacsove seje. Obnašajo se skoraj tako, kot če v Netscapovem Navigatorju uporabimo ukaz Alt+N.

2.3 Osnove tipkovnice

Ta razdelek opisuje osnove igranja na tipkovnico v Emacsu. Kot v večini zmogljivih urejevalnikov, je tudi v Emacsu vse, kar lahko v njem storimo, od nas oddaljeno le nekaj pritiskov na tipke.

Če ste uporabnik urejevalnika `vi`, ste se nekaj časa morali privajati na tipke `k, j, l, h`, s katerimi se v njem pomikamo za vrstico navzgor ali navzdol ali za znak naprej ali nazaj. Priznajmo, da smo potrebovali nekaj ur ali tednov, da smo se jim privadili in jih začeli udobno uporabljati za premikanje po datoteki.

Emacs se v tem ne razlikuje. Tudi tu se moramo naučiti različnih pritiskov tipk in ukazov. Kakor v `vi` se moramo naučiti le nekaj osnov, pa že lahko opravimo večino dela. Sčasoma se lahko počasi priučimo tudi drugih ukazov in razširimo svoje znanje, tako da znamo vedno hitreje opraviti svoje stvari.

2.3.1 Ukazne tipke (Meta, Esc, Control in Alt)

Kot se bomo kmalu naučili, Emacs zelo pogosto uporabljamo z večtipkovnimi kombinacijami, akordi. Ker Emacs ni načinovni urejevalnik (ang. modal editor), kakršen je `vi`, nam ni potrebno pomniti, kdaj smo v „ukaznem" (ang. command mode) ali „urejevalnem" načinu (ang. editing mode), da bi premaknili utripalko ali izvedli ukaz. Zato večinoma le pritisnemo pravo zaporedje ukazov in Emacs nas (običajno) takoj uboga.

Tipke, ki jih Emacs uporablja najpogosteje, so v dokumentaciji navedene takole `C` za tipko Control ali Ctrl in `M` za tipko Meta. Medtem ko ima večina sodobnih osebnih računalnikov eno ali več tipk, ki so označene s `Ctrl`, imajo le redki tipko z oznako `Meta`. V mislih vselej nadomestimo tipko Meta z eno od tipk `Esc` ali `Alt`. V večini običajnih primerov se obnašata tipki Esc in Alt enako.

Kadarkoli potemtakem vidite v Emacsovi dokumentaciji sklic na `C-x f`, to pomeni „pritisnite Control-x in zatem tipko f". In če vidite omenjeno kaj podobno temu `M-x shell`, to v resnici pomeni „pritisnite alt-x in vpišite besedo shell".

Za začetnika zelo koristen ukaz je `M-x apropos` ali skrajšano `C-h a`. Ukaz apropos preišče Emacsovo dokumentacijo na zvezi in poišče vse funkcije, katerih imena ustrezajo regularnemu izrazu, ki smo ga vpisali. Na ta način na primer zlahka poiščemo vse ukaze, ki so povezani z okvirji (ang. frame). Preprosto vnesemo `C-h a` in zatem vpišemo `frame`.

2.3.2 Premikanje po vmesniku

Ko smo spoznali, kaj pomenijo čudne okrajšave zaporedij tipk, je čas, da si ogledamo seznam najpogostejših ukazov za premikanje po vmesniku:

```
tipke       učinek
-----------------------------------
C-p         navzgor za vrstico
C-n         navzdol za vrstico
C-f         naprej za znak
C-b         nazaj za znak
C-a         začetek vrstice
C-e         konec vrstice
C-v         navzdol za celo stran
M-v         navzgor za celo stran
M-f         naprej za besedo
M-b         nazaj za besedo
M-<         začetek vmesnika
M->         konec vmesnika
C-g         prekini trenutni ukaz
-----------------------------------
```

In kot bi si želeli, tudi smerne tipke (tiste s puščicami) običajno delujejo po naših pričakovanjih. Tipka `vračalka` (ang. backspace) pa morebiti ne. A to je druga zgodba. `:-(`

2.3.3 Najpomembnejši ukazi

Potem ko smo se naučili, kako se premikamo po vmesniku, ali smo že pripravljeni tudi za odpiranje in shranjevanje datotek? Morebiti celo za iskanje? Ja, tukaj bomo našli teh nekaj najpomembnejših ukazov.

Preden pa se zapodimo med ukaze, naj na kratko pojasnim, kako delujejo.

Vsi „ukazni akordi" v Emacsu (ti, ki jih zapišemo kot `M-nekaj` ali `C-nekaj`) so v resnici le bližnjice do funkcij, ki so del Emacsa. Vsako izmed funkcij lahko pokličemo po njenem imenu, tako da vpišemo `M-x ime-funkcije` in pritisnemo `Enter`. Če pa je na tipkovnico prilepljena bližnjica do te funkcije, zadošča, da uporabimo bližnjico.

Emacsova funkcja, ki shrani vsebino vmesnika na disk, se imenuje `save-buffer`. Privzeto je pripeta na akord `C-x C-s`. Da bi shranili vmesnikovo vsebino, lahko pritisnemo akord ali pa vpišemo `M-x save-buffer` in dosežemo popolnoma enak učinek.

Vse najpogosteje uporabljane funkcije imajo vnaprej prirejene akorde. Nekaj jih navajamo spodaj.

```
akord       funkcija           opis
-------------------------------------------------------------------
C-x C-s     save-buffer        shrani trenutni vmesnik na disk
C-x u       undo               razveljavi zadnji ukaz
C-x C-f     find-file          odpri datoteko na disku
C-s         isearch-forward    poišči niz znakov v smeri naprej
C-r         isearch-backward   poišči niz znakov v smeri nazaj
            replace-string     poišči niz znakov in ga zamenjaj
            replace-regexp     poišči regularni izraz in ga zamenjaj
C-h t       help-with-tutorial poženi učenje Emacsa s prvim berilom
C-h f       describe-function  izpiši opis funkcije
C-h v       describe-variable  izpiši opis spremenljivke
C-h x       describe-key       izpiši pomen akorda
C-h a       apropos            poišči dokumentacijo z regularnim izrazom
C-h F       view-emacs-FAQ     prikaži odgovore na pogosta vprašanja o Emacsu
C-h i       info               izpiši dokumentacijo o Emacsu
C-x r m     bookmark-set       označi trenutno mesto z zaznamkom
C-x r b     bookmark-jump      skoči na zaznamek
-------------------------------------------------------------------
```

Ko bomo uporabljali te funkcije, bomo opazili, da mnoge od nas pričakujejo vnos. Tega vselej vnesemo v pogovornem oknu, kar spominja na ukaze `:` v `vi`.

V Emacs je vgrajenih dobesedno stotine funkcij, ki so vse dosegljive. Zgornji seznam je le izvleček ukazov, ki jih uporabljam najpogosteje. Za bolj podrobne opise, kaj te funkcije počno, in za njihov popolnejši seznam pa si moramo ogledati neposredno pomoč.

2.3.4 Dopolnjevanje s tipko Tab

Podobno kot mnoge priljubljene ukazne lupine v Unixu (bash, csh, tcsh...) tudi Emacs omogoča dopolnjevanje imen ukazov s tipko `Tab`. Resnici na ljubo je treba priznati, da dopolnjevanje ukazov v lupini bash posnema to v Emacsu. Če ga uporabljamo v lupini, se nam bo prikupilo tudi v Emacsu.

Poskusimo ga uporabiti tako, da vnesemo `M-x search` in nato pritisnemo `Tab`. Emacs bo imenu dodal pomišljaj in tako označil, da obstaja mnogo imen, ki se vsa začno s tem nizom in imajo pomišljaj. Ponovimo pritisk na tipko `Tab` in Emacs bo prikazal vsa imena, ki se začno z izpisanim nizom. Opazimo, da Emacs za ta seznam odpre *novo okno*. Emacs začasno prepolovi prikazovalnik v dve okni. Eno vsebuje vmesnik, ki ga urejamo, drugo pa seznam vseh dopolnjenih imen, ki se začno na „search-". Če pritisnemo ukaz za prekinitev `C-g`, zapustimo okno z izbiranjem in se vrnemo v prvotni vmesnik.

2.4 Prvo berilo, pomoč in strani info

Emacs ima za novince vgrajeno prvo berilo, ki nas popelje prek osnovnih ukazov za urejanje in funkcij, ki jih vsi uporabljamo. Tečaj nas tudi nauči, kako uporabljamo druge sisteme pomoči v Emacsu.

Priporočam, da si vzamemo čas in se sprehodimo skozi prvo berilo, če se želimo Emacsa res naučiti. Kot prikazuje zgornja preglednica, poženemo prvo berilo z akordom `C-h t`. Prvo berilo je namenjeno novincem, ki bi se radi samoizobrazili, narejeno pa je kot začetni tečaj.

Če poganjamo Emacs v oknih X, bomo opazili, da je skrajni menu v menujski letvi označen s Help. Ko raziskujemo, kaj nam Help nudi, najprej opazimo, da ima nekaj postavk v menuju na desni strani navedene akorde ukazov, s katerim jih na hitro poženemo.

Za konec si lahko ogledamo, kolikšna količina dokumentacije je na voljo v Emacsu. Vpišimo `M-x info` ali `C-h i`, kar požene sistem Info, brskalnik po dokumentaciji v Emacsu.

3 Načini v Emacsu

Načini (ang. mode) v Emacsu so povezana pravila obnašanja in sklopi funkcij, ki jih lahko vklapljamo ali izklapljamo in tako prilagodimo Emacs za uporabo v različnih okoliščinah. Z načini dosežemo to, da je Emacs lahko enako primeren za pisanje dokumentacije, za programiranje v raznolikih jezikih (C, C++, Perl, Python, Java in še mnogih drugih), za izdelavo domače strani, za elektronsko dopisovanje, za branje novičarskih skupin, za vodenje seznama naših zmenkov in celo za igranje iger.

Načini v Emacsu so preprosto knjižnice s kodo v lispu, s katero razširjamo, prilagajamo, spreminjamo ali na kakšen način izboljšujemo Emacs.

3.1 Glavni načini in podnačini

Obstajata dve vrsti načinov: glavni načini (ang. major mode) in podnačini (ang. minor mode). Razlika med vrstama ni najbolj očitna, dokler se ne poglobimo vanju, a poskusimo jo opisati.

Vsak vmesnik je v nekem trenutku le v enem glavnem načinu, poleg tega pa je lahko v več podnačinih. Glavni načini so v splošnem vezani na programsko okolje ali posebna velika opravila, medtem ko so podnačini manjše prilagoditve in manj specifični priročni programi, ki jih lahko sproti uporabljamo pri različnih večjih opravilih.

Tole je zvenelo precej abstraktno, zato raje ponazorimo s primerom. Ko pišem neformatirano besedilo, uporabljam glavni način, imenovan `text-mode`. Ta način je bil načrtovan za pripravo neformatiranega besedila, kakršno so datoteke README. Način razume, kako mora prepoznavati besede in odstavke ter kaj uporabnik pričakuje pri uporabi običajnih tipk za navigacijo.

Ko pripravljam besedilo, namenjeno ljudem, običajno hočem, da bo tudi njegov videz dober. Besedilo mora biti ovito (ang. word-wrapped), tako da vrstice niso predolge in podobno. Da omogočim ovijanje besedila, poženem podnačin `auto-fill`. Podnačin poskrbi, da se vrstice po „pričakovanju" samodejno pravilno prelomijo, ko vnašam besedilo. Ker je to podnačin, ga lahko uporabim tudi v drugih glavnih načinih. Moje „pričakovanje", kaj naj se zgodi ob zaključku vrstice pa je v `text-mode` različno od tistega, ko sem na primer v načinu `java-mode`. Prav gotovo si ne želim, da bo javanska koda ovita na enak način, kot je besedilo v slovenskem jeziku. Vendar pa *hočem*, da je v komentarjih javanske kode besedilo ovito! Podnačin `auto-fill` je dovolj pameten, da bo to sam razrešil.

Ustvarjalci različnih Emacsovih načinov so se zelo potrudili, da so zagotovili, da so tiste stvari, ki sodijo v podnačine, tudi res podnačini.

Če si ogledamo skico Emacsovega zaslona, opazimo, da načinovna vrstica kaže, v katerem načinu ali načinih je Emacs v danem vmesniku. V prikazanem primeru je Emacs v načinu „Lisp Interaction”, ki je privzeti način. Način je uporaben le za pisanje kode v lispu, a ker je večina Emacsa izvedena v njem, taka odločitev za privzeti način prav gotovo ni slaba, ali pač?

3.2 Načini za programiranje

Najvažnejše, Emacs so ustvarili programerji za programerje. Na voljo so visokokakovostni načini za skoraj vsakega od razširjenih programskih jezikov in tudi za nekaj manj pogostih. Tukaj bom na kratko opisal le nekatere od njih.

Za večino programskih načinov je značilnih nekaj skupnih lastnosti. Pogosto bodo ti načini:

- skladenjsko barvali (ang. syntax highlight) jezikovne elemente,
- samodejno zamikali in formatirali kodo v skladu s programskim jezikom,
- podpirali kontekstno občutljivo in od programskega jezika odvisno pomoč,
- samodejno podpira vmesnik do našega razhroščevalnika,
- v menujsko letev dodaja nove menuje za podporo jeziku.

Poleg tega obstaja nekaj načinov, ki niso v povezavi z jezikom, in pomagajo pri nalogah, ki so skupne mnogim programskim jezikom. Pri tem mislim na podporo za programe za nadzor različic, samodejno dodajanje komentarjev v kodo, izdelavo datotek Makefile, osveževanje dnevniških zapisov (ang. change log) in podobno.

Če združimo vse te načine ter upoštevamo zrelost in stabilnost Emacsove kode, ugotovimo, da je izdelek primerljiv komercialnim paketom integriranih razvojnih okolij (ang. integrated development environment, IDE) za jezike, kot sta C++ in Java. Poleg tega pa je Emacs na voljo brezplačno, seveda!

3.2.1 C, C++ in Java

Zaradi c-jevske skladnje sta jezika C++ in java jeziku C precej podobna, in tudi način, ki podpira vse tri jezike, poleg njih pa še Objective-C in IDL, je le en sam. Paket, v katerem je način podprt, je zelo zrel in lepo zaokrožen ter prihaja s standardno distribucijo Emacsa. Način se imenuje `cc-mode` ali tudi `CC Mode`.

Če iščemo njegovo najnovejšo različico ali če želimo zvedeti podrobnosti, obiščimo spletno stran *http://www.python.org/emacs/*.

3.2.2 Perl

V Emacs sta vgrajena dva načina za urejanje kode v perlu. Prvi se imenuje `perl-mode` (kot seveda pričakujemo), drugi pa `cperl-mode`. Ne poznam zgodovinskih okoliščin in razlogov, zakaj je prišlo do dveh načinov, dokumentacija pa jih tudi ne omenja. Zdi pa se, da je bil `perl-mode` razvit najprej. Je manj zmogljiv kot `cperl-mode` in ne prepozna nekaterih bolj zavitih perlovih jezikovnih tvorb.

Osebno uporabljam in priporočam `cperl-mode`, ki kaže, da se ga vzdržuje precej pogosto, ima pa tudi skoraj prav vse funkcije, ki bi si jih kdajkoli želel. Njegova zadnja različica je dostopna na strani: *ftp://ftp.math.ohio-state.edu/pub/users/ilya/emacs*.

Ne verjemite mi slepo na besedo. Raje preskusite oba načina in se odločite za tistega, ki vam bolj ustreza.

3.2.3 Python

Python, zelo priljubljen skriptni jezik, ima tudi Emacsov način, s katerim je podprt. Kot lahko poročam, se ta način *ne* razširja z GNU Emacs, pač pa le z XEmacs. Deluje pa v obeh urejevalnikih prav dobro.

Paket za podporo načinu `python-mode` lahko pridobimo z uradne pythonove spletne strani *http://www.python.org/emacs/python-mode/*.

3.2.4 Drugi načini

Poleg že naštetih je na voljo programerjem še mnogo mnogo drugih načinov. Ti načini pomagajo pri:

- ukaznih lupinah (Bash, sh, ksh, csh...),
- awk, sed, tcl...
- datotekah Makefile,
- dnevniških zapisih (ang. change logs),
- dokumentaciji,
- razhroščevanju

In še veliko več. Zadnji razdelek tega spisa pojasnjuje, kje lahko najdemo druge načine in dodatke za Emacs.

3.3 Pisanje

Čudoviti Emacsovi načini *niso* omejeni le na pomoč pri pisanju kode. Tudi pisci vsakovrstne dokumentacije lahko s pridom uporabljamo obširen nabor Emacsovih načinov.

3.3.1 Črkovanje (način `ispell`)

Avtorji različnih vrst dokumentacije od časa do časa potrebujemo orodje za preverjanje pravilnosti črkovanja (ang. spellchecking). če smo namestili program **GNU ispell**, lahko uporabimo ukaz `M-x ispell` in poženemo črkovanje trenutnega vmesnika. Če črkovalnik najde besedo, ki je ne pozna, predlaga za zamenjavo podobne besede iz slovarja in nam omogoči, da izmed njih izberemo eno ali nobene. Po zmožnostih je podoben mnogim razširjenim neprostim črkovalnim paketom.

3.3.2 HTML (način `html-helper`)

Če moramo od časa do časa ali celo pogosteje pripraviti spis v zapisu HTML, bomo morebiti uporabili način `html-helper-mode`. Paket je dostopen na strani *http://www.santafe.edu/~nelson/tools/* kot tudi spremljajoča dokumentacija in dodatki.

Kot kaže ime paketa, način `html-helper-mode` predvsem pomaga tistim piscem, ki se lotevajo pisanja dokumentov v zapisu HTML na roko, na star način.

3.3.3 TeX (`tex-mode`)

Če pišemo v TeXu, je priročno, da nam Emacs sproti barva ukaze, besedilo v nekaterih oklepajih in kar je še te oblikovne krame. Način `tex-mode` se potrudi in stori to samodejno.

Čeprav ne pišem več pogosto v TeXu, mi je nekoč ta način močno pomagal, da je postalo pisanje preglednejše.

3.3.4 SGML (`sgml-mode`)

Spis, ki ga pravkar berete, je bil pripravljen v jeziku SGML in kasneje zelo verjetno pretvorjen v drug zapis, ki ga berete. Način `sgml-mode` podpira osnove dokumentov SGML: preverjanje njihove skladenjske pravilnosti, skladenjsko barvanje, premikanje po značkah in poleg tega še veliko več. Način je del standardne distribucije Emacs.

3.4 Drug načini

Seveda, poleg naštetih obstaja še mnogo drugih načinov, ki nam olajšujejo življenje. Vzorec bolj priljubljenih za pokušino:

3.4.1 Nadzor različic (način `vc`)

Način `vc` je vmesnik do razširjenih sistemov za nadzor različic (RCS, SCCS in CVS). Omogoča preprost vnos in povrnitev nadzorovanih datotek, pomaga pri upravljanju z različicami in izdajami programov ter podobno. Je del standardne distribucije Emacsa in je obširno opisan v Emacsovi dokumentaciji.

3.4.2 Način za ukazno lupino

Zakaj bi morali skočiti v drugo okno X ali navidezno konzolo zgolj zaradi tega, ker bi radi pognali nekaj ukazov v ukazni lupini? Storimo to v Emacsu in si prihranimo nekaj truda. `:-)`

Ukaz `M-x shell` v Emacsovem vmesniku požene ukazno lupino. V njej lahko postorimo skoraj vse, kar lahko v pravi ukazni lupini. Ne moremo pa poganjati programov, ki zahtevajo celo okno, taka sta na primer `vi` ali `pine`. Emacs se s pravo ukazno lupino pogovarja v ozadju.

Ker je ta način del standardne distribucije, najdemo navodila za njegovo uporabo v priloženi dokumentaciji.

3.4.3 Telnet in FTP

Zakaj bi morali preklopiti v drugo okno X ali navidezno konzolo zgolj zaradi tega, ker bi radi pognali programa telnet ali ftp? Tudi za to ni pravega razloga, oboje lahko storimo v Emacsu in si prihranimo nekaj truda. (Ali ste že opazili vzorec?)

Enako kot ukazno lupino v Emacsu poženemo tudi programa telnet ali ftp. Uporabimo ukaza `M-x telnet` ali `M-x ftp` in ju preskusimo sami. Dokumentacija pa nam bo pomagala pri razreševanju grenkih podrobnosti.

3.4.4 Man

Zakaj bi morali preklopiti v drugo okno X ali navidezno konzolo zgolj zaradi tega, ker bi radi prebrali strani man? To lahko storimo iz Emacsa in si prihranimo nekaj truda. (Obljubljam, da se ne bom več ponovil.)

Enako kot ukazno lupino v Emacsu poženemo tudi pregledovalnik strani man. Uporabimo ukaz `M-x man` in ga preskusimo. Več pa lahko preberemo v dokumentaciji.

3.4.5 Ange-FTP

Citat iz navodil za `ange-ftp`:

> Ta paket se trudi narediti dostop iz GNU Emacs do datotek in imenikov prek protokola FTP kar najbolj preprost in pregleden. Podmnožica skupnih rutin za delo z datotekami je razširjena za delo s programom FTP.

To pomeni, da do datotek v oddaljenih sistemih dostopamo, kot da bi bile v lokalnem sistemu. Če moramo urediti datoteko v drugem sistemu, Emacsu z imenom sistema v poti do datoteke povemo, kje naj jo odpre, Emacs pa bo poskrbel za vse ostalo in pridobil datoteko z oddaljenega sistema. Ko datoteko shranimo z običajnim ukazom `C-x C-s`, bo `ange-ftp` prestregel klic funkcije za shranjevanje in sam datoteko zapisal v oddaljeni sistem.

Skladnja za označevanje datotek je takole razširjena... Datoteko „moja-datoteka.txt" uporabnice „metka" v njenem imeniku „naloga/podatki" oddaljenega sistema „kjer.koli.si" odpremo z ukazom `C-x f`, tako da za pot navedemo:

```
/metka@kjer.koli.si:naloga/podatki/moja-datoteka.txt
```

Tudi ta paket je del standardne distribucije Emacsa in obširnejša navodila zanj najdemo v Emacsovi standardni dokumentaciji.

Zahvaljujem se Etiennu Grossmannu (*etienne@anonimo.isr.ist.utl.pt*) za zgornji primer.

4 Emacs po meri

Emacs prilagodimo po svoji meri, tako da spremenimo kodo v lispu. Nastavimo lahko spremenljivke, ki vplivajo na obnašanje Emacsa ali dodamo nove funkcije, s katerimi dosežemo nove učinke ali pa z njimi nadomestimo obstoječe funkcije.

4.1 Začasno prilagojevanje

Če poskušamo Emacs prilagoditi, bomo zelo verjetno to najprej storili tako, da bodo spremembe le začasne. Če ga pri prilagojevanju (ang. customization) zelo močno polomimo, zadošča, da Emacs zapustimo z običajnim ukazom `C-x C-c` in ga ponovno poženemo.

Ko pa se odločimo, katere spremembe želimo ohraniti za vselej, jih lahko dodamo v svojo osebno datoteko `.emacs` in njihov učinek se bo poznal ob vsakem zagonu Emacsa. O tem govorimo v naslednjih razdelkih.

4.1.1 Prirejanje vrednosti spremenljivkam

Emacs najpreprosteje spremenimo, tako da spremenljivkam, od katerih je odvisno njegovo obnašanje, določimo prave vrednosti. Lispova koda, s katero spremenljivki priredimo vrednost, je videti takole:

```
(setq ime-spremenljivke nova-vrednost)
```

Pri tem je `ime-spremenljivke` ime spremeljivke, ki ji prirejamo vrednost, `nova-vrednost` pa vrednost, ki jo prirejamo (V lispovem žargonu pravimo, da spremenljivko povežemo z vrednostjo.) Lispova funkcija `setq` ustreza prireditvenemu operatorju (običajno predstavljenim z znakom =) v drugih programskih jezikih.

Opomba: Na tem mestu preskakujem mnoge podrobnosti, da bi ostal dovolj razumljiv. Mnogi poleg te uporabljamo še podobne lispove funkcije, kakršna je `set` ali celo `setq-default`. Če ste resnično radovedni, čemu so namenjene, si oglejte Emacsov priročnik o jeziku lisp.

Oglejmo si vrstico iz moje datoteke `.emacs`:

```
(setq-default transient-mark-mode t)
```

Spremenljivka `transient-mark-mode` nadzoruje, ali bo območje, ki ga izberemo, poudarjeno označeno ali ne. V mnogih uporabniških programih v GUI se klik miške in njen premik uporablja za označevanje besedila, pri čemer to postane označeno z inverznim videom ali ustreznimi barvami. Emacs stori podobno, če je spremenljivka `transient-mark-mode` nastavljena na neprazno (ang. non-nil) vrednost.

Kakšno vrednost?

Počasi. Potrebna je kratka zastranitev. Večina programskih jezikov pozna logični vrednosti resnično in neresnično. V jezikih C in C++ je logična vrednost spremenljivke enaka resnično, če je njena vrednost različna od nič. V perlu je logična vrednost enaka resnično, če je vrednost spremenljivke različna od nič ali od null. ???????Prevod??????? V lispu gre za enako idejo, le imena in oznake so drugačne.

Logična vrednost resnično je v lispu običajno označena kot `t` in neresnično (ali null) kot `nil`. Enako kot v drugih jezikih je vsaka vrednost, ki je različna od neresnično, vzeta kot resnično.

Da si ogledamo popolni opis, na kaj vse vpliva spremenljivka `transient-mark-mode`, uporabimo pomoč na zvezi. Vpišimo `C-h v` ali `M-x describe-variable` in nato `transient-mark-mode`. Če smo leni, si lahko pomagamo s tipko za dopolnjevanje `Tab`. Vpišimo le začetni kos imena spremenljivke in pritisnimo tipko `Tab`. Če smo že zapisali dovoljšen del njenega imena, da jo lahko Emacs enoznačno določi, bomo takoj dobili njeno ime izpisano v celoti.

Druga spremenljikva, ki jo pogosto nastavljamo, je `fill-column`. Spremenljivka pove, kako širok je zaslon, da more Emacs v načinu `auto-fill-mode` glede na to primerno ovijati besedilo. Da bi si najbolj nazorno ogledali njen učinek, jo nastavimo na absurdno majhno vrednost:

```
(setq fill-column 20)
```

A zgolj zapis tega izraza še ne stori nič. Emacsu moramo povedati, naj zapisani ukaz tudi **izvede** (ang. evaluate). Utripalko postavimo na konec izraza in vtipkajmo `C-x C-e`, kar pokliče funkcijo `eval-last-sexp`, če vas že zanima. Ko storimo to, lahko opazimo, da se je v pogovornem oknu na dnu zaslona izpisala vrednost `20` (ali karkoli ste uporabili za nastavitev). To je tudi vrednost funkcije, ki smo jo pravkar uporabili.

Da se prepričamo, kako ovijanje po novem deluje, vpišimo stavek ali dva. Če imamo omogočen podnačin `auto-fill-mode` (čeprav ga verjetno nimamo), lahko opazimo, da se besedilo ovija pred 20 stolpcem. Drugače pa zatem, ko smo vnesli nekaj besedila, pritisnemo `M-q`, kar pokliče funkcijo `fill-paragraph`. Ta pa ovije besedilo zadnjega odstavka.

4.1.2 Zveze z datotekami

Emacs lahko tako prilagodimo, da bo določeno opravilo samodejno naredil vsakič, ko odpremo datoteko določenega tipa. To je podobno nekaterim GUI, ki samodejno poženejo namenski program (ang. application) vsakič, ko kliknemo na ikono datoteke določenega tipa. Ali želimo na primer, da Emacs samodejno preklopi glavni način v `text-mode` vsakič, ko odpremo datoteko s končnico `.txt`? V resnici se je to že zgodilo. `:-)` No, pa naročimo Emacsu, naj datoteko „README” odpre vselej v načinu `text-mode`.

```
(setq auto-mode-alist (cons '("README" . text-mode) auto-mode-alist))
```

Hmm?

Ne da bi se poglobili v programiranje v lispu, ki za občutek razumevanja ni nujno potrebno, a škodi tudi ne, če se ga naučimo, zaenkrat zadošča, da povemo, da spremenljivka `auto-mode-alist` vsebuje seznam parov. Vsak par je sestavljen iz regularnega izraza in iz imena Emacsovega načina. Če ime datoteke, ki jo odpiramo, ustreza kateremu od regularnih izrazov (v našem primeru nizu „README”), bo Emacs pognal način, ki smo ga podali.

Na prvi pogled zamotana skladnja zgornjega primera je takšna zaradi tega, ker dodajamo seznamu parov nov par. Najbrž ne bi želeli, da spremenljivki `auto-mode-alist` zgolj priredimo novo vrednost in pri tem izgubimo vse že nastavljene pare.

In če bi želeli, da Emacs samodejno uporabi način `html-helper-mode` vsakič, ko odpremo datoteko s končnico `.html` ali `.htm`, bomo v svojo datoteko .emacs dodali vrstici:

```
(setq auto-mode-alist (cons '("\\.html$" . html-helper-mode) auto-mode-alist))
(setq auto-mode-alist (cons '("\\.htm$" . html-helper-mode) auto-mode-alist))
```

Število možnosti je resnično neomejeno.

4.2 Uporaba datoteke `.emacs`

Po tem ko smo v Emacsu že preživeli nekaj časa in dobili občutek, kaj lahko storimo s prilagojevanjem, bomo morebiti želeli prilagoditve narediti za trajne (ali vsaj do naslednje spremembe). če uporabljamo Emacs vsak dan, bomo opazili, da velikost datoteke .emacs s časom narašča. To je *zelo drobro*, saj kaže, da smo ugotovili, kako naj nam Emacs služi tako, kot **hočemo**, da bo. Velika škoda je, da se enako ne obnaša še več programov.

če se morebiti še nismo vprašali, vsakič ko poženemo Emacs, ta najprej v našem domačem imeniku poišče datoteko `.emacs`. V njej je koda v lispu, ki jo želimo pognati ob vsakem zagonu programa Emacs, s čimer omogočimo prilagojevanje, o katerem smo pravkar govorili.

še en primer iz moje datoteke `.emacs`:

```
(setq inhibit-startup-message t)
```

Spremenljivka `inhibit-startup-message` odloča, ali bo Emacs ob zagonu izpisal pozdravno sporočilo ali pa ga ne bo. Ko bomo Emacs uporabljali že nekaj časa, nam bo pozdravljanje verjetno šlo pošteno na živce. Sam sem znal uporabiti pomoč in poiskati način, kako pozdravljanje izklopim.

Za nalogo v domačem imeniku ustvarimo datoteko `.emacs` in vanjo dodajmo navedeno vrstico. Zapustimo Emacs in ga ponovno poženimo. Pozdravnega sporočila ob novem zagonu ne bi smelo biti.

Pogosto bomo v zvezi z Emacsom ali kakšnim od spremljajočih paketov prebrali nasvet, kako lahko s primerno kodo v datoteki `.emacs` prilagodimo njihovo delovanje.

Datoteka z odgovori na pogosto zastavljena vprašanja, dosegljiva z `C-h F`, vsebuje nekaj predlogov v zvezi s prilagojevanjem, kar nas bo morebiti zanimalo.

4.3 Paket za prilagojevanje

Ko je Emacs s časom vse bolj rasel in se razvijal, se je nekdo domislil, da „mora obstajati lažji način za novince, da si ga prilagodijo”. Nastal je paket za prilagojevanje `customize`.

Paket ponuja preprostejše prilagojevanje delov Emacsa. Preskusimo ga tako, da v glavnem menuju `Help` obiščemo podmenu `Customize` ali pa vtipkamo `M-x customize`.

Možnosti za prilagojevanje so razvrščene v logične skupine, denimo „Editing”, „Programming”, „Files” in podobno. Nekatere skupine vsebujejo nove podskupine.

če si Emacs prilagodimo s pomočjo tega vmesnika, bo Emacs spremembe vpisal v našo datoteko `.emacs`. To je zelo priročno, saj si lahko spremembe ogledamo in jih po potrebi sami popravimo.

Ker ne uporabljam vmesnika za prilagojevanje „Customize”, žal ne morem veliko povedati o njem.

4.4 Zaslon oken X

Kot vsak lepo vzgojen namenski program tudi Emacs upošteva sredstva (ang. resource) oken X. S tem poudarjamo, da lahko začetne nastavitve, denimo za barve, geometrijo okna in podobne lastnosti oken X, opravimo na enak način kot pri programih `xterm`, `nxterm` in sorodnih.

V ilustracijo navajam ustrezne vrstice iz svoje datoteke `~/.Xdefaults`:

```
emacs*Background: DarkSlateGray
emacs*Foreground: Wheat
emacs*pointerColor: Orchid
emacs*cursorColor: Orchid
emacs*bitmapIcon: on
emacs*font: fixed
emacs.geometry: 80x25
```

Opomba prevajalca: Uporabniki, ki pišemo v razporedu ISO-8859-2, imenovanem tudi Latin-2, bomo raje uporabili enakovredno pisavo s šumniki, če jo seveda imamo nameščeno v sistemu:

```
emacs*font: -misc-fixed-medium-r-semicondensed--13-120-75-75-c-60-iso8859-2
```

Za obširnejša navodila o sredstvih X si moramo ogledati strani man, ki so posvečene oknom X, `man X`.

Chris Gray (*cgray4@po-box.mcgill.ca*) pripominja:

> Zdi se, da se v distribuciji Debian datoteka `~/.Xdefaults` ne uporablja. Namesto vanjo lahko postavimo ukaze z enakim uspehom v datoteko `/etc/X11/Xresources/emacs`, pa bomo dobili enako pisane barve, kot ko smo uporabljali RedHat.

5 Priljubljeni paketi

Poleg mnogih Emacsovih načinov je na voljo še veliko dodatnih **paketov** (ang. package). Imenujem jih paketi, ker so več kot le novi načini. Običajno vključujejo posebne priročne programe (ang. utility) ali pa so tako obsežni, da bi bila oznaka „načini" do njih krivična. V nekaterih primerih pa gre za tako programsko opremo, ki razširja ali združuje ostale Emacsove načine ali pakete. Definicija ni popolnoma jasna, a kakršna je, takšna zadošča.

5.1 VM (poštni program)

Navedek iz odgovorov na pogosto zastavljena vprašanja o VM:

> VM (View Mail) je podsklop v Emacsu za branje in odpošiljanje pisem znotraj Emacsa. Vsebuje ukaze, ki jih pričakujemo in potrebujemo za delo s pošto, na primer pripravo odgovorov na pisma, shranjevanje pisem v mape, brisanje pisem in podobno. Poleg teh obstajajo še ukazi za pripravo izvlečkov in povzetkov (ang. digest), prepošiljanje sporočil, njihovo urejanje glede na različne kriterije...

Ko sem začenjal z Emacsom, sem nekaj časa uporabljal VM. Zdelo se mi je, da je prava zamenjava za Pine, Elm ali večino podobnih poštnih programov. A za branje pošte in novic želim uporabljati le en program. VM se razvija še danes in je dobro podprt.

Na voljo je na spletni strani *http://www.wonderworks.com/vm/*.

5.2 Gnus (program za pošto in novice)

Navedek iz priročnika za GNUS:

> Gnus je laboratorij za branje sporočil. Omogoča, da si skoraj vse stvari ogledamo, kot bi bile novice iz novičarske skupine. Z njim lahko beremo elektronsko pošto, pregledujemo imenike ali uporabljamo program ftp, z njim lahko celo beremo novičarske skupine.
>
> Gnus poskuša ponuditi uporabnikom, ki berejo novičarske skupine, tisto, kar daje Emacs uporabnikom, ki urejajo besedilo. Gnus uporabnika pri tem nikakor ne omejuje in mu pomaga razširiti Gnus, da se obnaša, kakor uporabnik želi. Program ne sme upravljati z ljudmi, ti morajo imeti možnost, da s programom storijo ali starejo, karkoli želijo!

Kot ste morebiti opazili, trenutno uporabljam za branje pošte in novičarskih skupin paket GNUS. GNUS se še vedno razvija in je dobro podprt.

Na voljo je na spletni strani: *http://www.gnus.org/*.

5.3 BBDB (rollodex)

BBDB je ime za Insidious Big Brother Database, program tipa rollodex. Deluje dobro v povezavi z drugimi Emacsovimi programi za pošto (vključno z VM in GNUS).

Na voljo je na spletni strani: *http://pweb.netcom.com/~simmonmt/bbdb/index.html*.

5.4 AucTeX (še en način za TeX)

AucTeX je še en način za urejanje datotek TeX.

Navedek s spletne strani za AucTeX:

> AUC TeX je razširljivi paket za pisanje in oblikovanje datotek v zapisu TeX za večino različic urejevalnika GNU Emacs. Podprtih je mnogo različnih paketov s texovimi makroukazi, vključno s paketi AMS TeX, LaTeX in TeXinfo.

Na voljo je na spletni strani: *http://sunsite.auc.dk/auctex/*.

6 Drugi viri

Ta razdelek govori o knjigah, spletnih straneh, novičarskih skupinah, seznamih elektronskih naslovov in podobnih krajih, kjer je najti gradivo o Emacsu.

6.1 Knjige

Na voljo je le nekaj zares dobrih knjig za učenje Emacsa. Poleg tega pa bomo ugotovili, da mnoge knjige o Linuxu in Unixu vsebujejo poglavje, ki je namenjeno Emacsu (in urejevalniku `vi`).

6.1.1 Learning GNU Emacs

Avtorji: Debra Cameron, Bill Rosenblatt, Eric S. Raymond

Izdajatelj: O'Reilly & Associates - *http://www.ora.com/*

Knjigo lahko kupimo s popustom pri Amazon.com z njihovim družabniškim programom: *http://www.amazon.com/exec/obidos/ASIN/1565921526/*

Komentar: To je verjetno najboljša knjiga za novince. Potem ko smo si ogledali pričujoči HOWTO in se sprehodili po odgovorih na pogosto postavljena vprašanja, je ta knjiga najprimernejši in najbolj izčrpni vodnik.

6.1.2 Writing GNU Emacs Extensions

Avtor: Bob Glickstein

Izdajatelj: O'Reilly & Associates - *http://www.ora.com/*

Knjigo lahko kupimo s popustom pri Amazon.com z njihovim družabniškim programom: *http://www.amazon.com/exec/obidos/ASIN/1565922611/*

Komentar Potem ko smo Emacs že uporabljali nekaj časa in se odločili, da napišemo svoj lastni način ali pa opravimo bolj zahtevno prilagoditev, nam bo morebiti koristila ta knjiga. Četudi nas ne poskuša naučiti jezika lisp, vsebuje kratek uvod v ta jezik.

6.1.3 Programming in Emacs Lisp: An Introduction

Avtor: Robert J. Chassell

Iz datoteke README:

> To je osnovni uvod v programiranje v Emacsovem jeziku lisp za ljudi, ki niso programerji in ki jih programiranje morda niti ne zanima, a bi si radi prilagodili ali razširili svoje računalniško okolje.

Priročnik je v celoti na voljo za brezimni ftp v strežniku GNU FTP: *ftp://prep.ai.mit.edu/gnu/emacs/*.

Knjiga v lepem tisku je na voljo tudi pri Amazon.com prek njihovega družabniškega programoma: *http://www.amazon.com/exec/obidos/ASIN/1882114418/jeremydzawodny/*.

Komentar: Knjiga je dober uvodni priročnik v Emacsov lisp in je primerna tudi za tiste, ki nismo dolgometražni programerji.

6.1.4 The GNU Emacs Lisp Reference Manual

Avtor: Richard Stallman

Izdajatelj: The Free Software Foundation - *http://www.fsf.org/*

Priročnik je v celoti na voljo za brezimni ftp v strežniku GNU FTP: *ftp://prep.ai.mit.edu/gnu/emacs/*.

Komentar: Knjiga je ključni vodnik (ang. definitive guide) za programski jezik Emacs Lisp.

6.2 Spletna mesta

6.2.1 EMACSulation

EMACSulation je ime kolumna, ki ga piše Eric Marsden za revijo na zvezi Linux Gazette, dostopno na spletnem mestu *http://www.linuxgazette.com/*. Ob času tega pisanja je bil zadnji članek *http://www.linuxgazette.com/issue39/marsden.html*. Ob koncu članka so napotki, kje najti prejšnje dele.

6.3 Novičarske skupine

Povežimo se s svojim krajevnim strežnikom za novičarske skupine in poiščimo tiste, ki v imenu vsebujejo niz „emacs", pa jih bomo gotovo našli nekaj. Moj strežnik posreduje naslednje skupine:

- comp.emacs
- comp.emacs.sources
- gnu.emacs
- gnu.emacs.bug
- gnu.emacs.help
- gnu.emacs.sources

6.4 Seznami elektronskih naslovov

Edini seznam elektronskih naslovov, za katerega vem, da je namenjen uporabnikom Emacsa, združuje uporabnike različice Emacsa za Micro$oftov sistem Windows NT. Več o tem v seznamu odgovorov na pogosto zastavljena vprašanja o NT-Emacs *http://www.cs.washington.edu/homes/voelker/ntemacs.html*.

6.5 Arhiv za Emacsov lisp

Navedek iz datoteke README iz arhiva za Emacsov lisp:

> Arhivi za Emacsov lisp v strežniku ftp.cis.ohio-state.edu vsebujejo različne koščke in pakete kode v Emacsovem lispu. Emacsov lisp je jezik za razširitev urejevalnika GNU Emacs, ki ga izdaja Free Software Foundation. Čeprav je večina kode za Emacsov lisp vključena v distribucijo za GNU Emacs, je mnogo ljudi napisalo pakete za vmesnike do drugih sistemov, za boljšo podporo urejanju v programskih jezikih, ki jih uporabljajo, dodalo povsem nove funkcije ali pa spremenilo privzeto obnašanje Emacsa. Večino vsebine tega arhiva so pripravili posamezniki, ki so jo distribuirali v javnost prek internetnih seznamov elektronskih naslovov „info-emacs" ali „info-gnu-emacs" in novičarskih skupin „comp.emacs", „gnu.emacs" ali „gnu.emacs.sources".

Arhivi so na voljo za brezimni dostop ftp na spletnem mestu *ftp://ftp.cis.ohio-state.edu/pub/emacs-lisp/*.

Opomba: Kolikor vem, arhiv za Emacsov lisp počasi zastareva. Nove ali osvežene pakete na njem vidim le poredko, čeprav vem, da obstajajo in da so celo priobčeni v novičarski skupini `comp.emacs.sources`. (Prosim, popravite me, če se motim.)

7 Zahvale

Naslednji pisci so pomagali pri nastajanju tega spisa.

- Robert Vollmert *rvollmer@gmx.net*
- Larry Brasfield *larrybr@seanet.com*
- Etienne Grossmann *etienne@anonimo.isr.ist.utl.pt*
- Thomas Weinell *kf6mli@amsat.org*
- Adam C. Finnefrock *adam@bigbro.biophys.cornell.edu*
- Chris Gray *cgray4@po-box.mcgill.ca*
- Robert J. Chassell *bob@rattlesnake.com*
- Isaac To *kkto@csis.hku.hk*
- Matteo Valsasna *valsasna@elet.polimi.it*
- Tijs van Bakel *smoke@casema.net*