Cui i-e frică de Emacs?

Mihai BUDIU – budiu@cs.cornell.edu
http://www.cs.cornell.edu/Info/People/budiu/budiu.html

21 martie 1997

Cuprins

Subiect: Comenzile esenţiale ale editorului Emacs

Cuvinte cheie: editor de texte

Cunoştinţe necesare: un editor de texte

Sunt un mare fan al editorului de texte Emacs. Nu am devenit dintr-o dată: prima oară cînd mi s-a sugerat să învăţ să-l folosesc am strîmbat din nas după 15 minute de citit manualul introductiv. "Prea multe lucruri de memorat, nici o regulă, nu aranjează frumos în pagină, ce-mi trebuie mie ceva aşa de sofisticat" — motive ca să amîn.

Ei bine, recunosc, am greşit. Emacs este un program foarte simpatic, foarte eficient şi foarte flexibil. Atît de flexibil încît mai am foarte multe lucruri de învăţat despre el, deşi îl folosesc de 4 ani şi adaug mereu cunoştinţe.

Nu, nu plecaţi. Nu am spus asta ca să vă descurajez, ci dimpotrivă! Ideea pe care vreau s-o pun în evidenţă este că se pot învăţa foarte repede toate funcţiile esenţiale din Emacs, care sunt utile (şi suficiente) pentru treburile de zi cu zi. Dacă folosiţi în mod regulat un calculator pentru a scrie texte (de orice fel: programe, articole, scrisori), ar fi o idee bună să învăţaţi să folosiţi Emacs, pentru că vă va permite ca în scurtă vreme să faceţi aceste lucruri mai repede şi mai eficient decît multe alte editoare.

Dacă nu sunteţi un sofisticat nimeni nu vă împinge la a folosi cantitatea nesfîrşită de resurse pe care Emacs le pune la dispoziţie: puteţi să vă mulţumiţi cu puţin. Dacă sunteţi însă un împătimit al calculatoarelor, Emacs se poate dovedi o sculă foarte eficientă pentru multe dintre activităţile dumneavoastră, şi atunci merită într-adevăr să-l învăţaţi.

Articolul acesta este foarte elementar; nu îşi propune deloc să pună în evidenţă toată puterea care se ascunde sub acest motor de programe interactive (Emacs este folosit pentru multe alte lucruri decît scrierea de texte: cam toate activitățile de natură interactivă: căutarea prin fişiere, plimbatul pe web, depanarea programelor sunt toate activități la care Emacs poate ajuta). Din această cauză foarte adesea nu vom spune tot adevărul cu privire la anumite lucruri, ci doar atît cît este nevoie.

Există mai multe "digest"-uri (sumare) cu comenzile esenţiale Emacs în circulaţie. Întîmplător nici unul nu mi s-a părut foarte sintetic, aşa că m-am apucat să mai fac eu încă unul.

1 Terminologie

Probabil cea mai grea este pentru un începător problema terminologiei; Emacs are nume foarte precise pentru tot felul de componente, pe care le foloseşte în mod uniform în toate documentaţiile. De aceea este foarte important să ştim ce este un "buffer", o fereastră ("window"), marcă ("mark") şi punctul ("point").

Două motive mă îndeamnă să păstrez numele originale ale comenzilor şi termenilor, în engleză:

- Toată lumea aşa le foloseşte; o traducere nu ar putea ajuta la comunicare, pentru că nimeni nu ar înţelege mai bine despre ce e vorba;
- Numele abrevierilor/tastelor sunt mnemonice (uşor de reţinut) în engleză.

Din păcate aceste două motive vor face textul românesc să fie amplu pigmentat cu cuvinte englezeşti. Cred că aceasta va fi multă vreme soarta scrierilor despre calculatoare.

Buffer = zonă de lucru separată. Una din trăsăturile foarte utile ale lui Emacs este că poate lucra cu mai multe fişiere simultan. Fiecare fişier este deschis pentru inspectare/modificare într-un "buffer" diferit. La un moment dat numai unele dintre buffere sunt vizibile utilizatorului; celelalte sunt ascunse undeva "în spate", dar conţin fişierele aşa cum au fost ele modificate. În fiecare clipă unul dintr buffere este cel "curent": în care se petrece acţiunea.

Window = un buffer vizibil. Fiecare fişier este prelucrat într-un buffer care-i aparţine. Numai unele dintre buffere sunt însă vizibile. Emacs afişează la un moment dat la un moment numai unele dintre buffere. Fiecare buffer vizibil este arătat într-o fereastră ("window"). Numărul de ferestre vizibile este variabil, controlat de utilizator. Bufferul curent este întotdeauna într-o fereastră.

Cursor = locul curent unde se acţionează Noţiunea de cursor este familiară tuturor editoarelor de texte; cursorul este locul unde următorul caracter va fi inserat/şters. Emacs încearcă să arate cursorul ca pe un dreptunghi mic negru.

Point = poziţia curentă într-un buffer Fiecare buffer are propriul lui punct. Caracterele apăsate sunt inserate în text în locul unde se află punctul. Diferenţa dintre cursor şi punct este că există un punct în fiecare buffer, dar unul singur dintre puncte este şi cursor, cel din buffer-ul curent.

Minibuffer = locul pentru dialoguri cu Emacs Unul dintre buffere este rezervat pentru dialogul cu Emacs; acolo Emacs raportează rezultatele comenzilor, afişează combinaţiile de taste care au fost apăsate parţial.

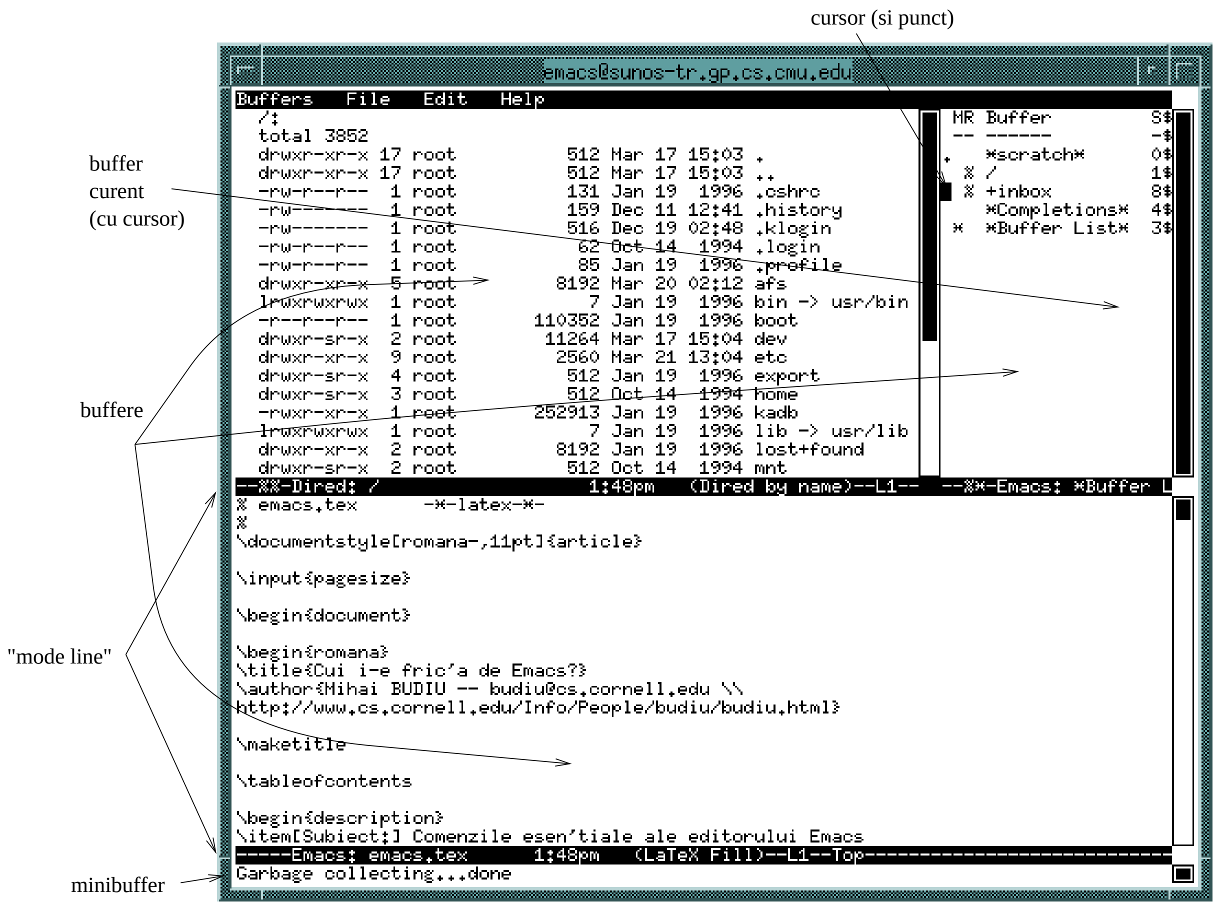

Figure 1: Emacs în acţiune.

Figura 1 arată un editor Emacs în acţiune. Sunt vizibile 3 ferestre, fiecare desigur cu un buffer. (Mai există încă un buffer care nu e vizibil).

Mode line = raportul stării curente Fiecare buffer are la bază o linie care de obicei este tipărită pe fond negru, care sumarizează lucrurile importante despre acel buffer. Printre altele linia arată ce fişier este prezent în buffer, dacă fişierul a fost modificat, "modul major şi cele minore active" în buffer (vedeţi ceva mai jos), poziţia cursorului (relativă şi absolută = linie/coloană). Utilizatorul poate influenţa ce informaţii să apară pe linia de mod, aşa că forma ei nu este întotdeauna aceeaşi. Nu o să-i consacrăm prea mult spaţiu în acest articol.

mark = o poziţie marcată în buffer Aşa cum fiecare buffer are un punct, fiecare poate avea şi o "marcă". Marca este o poziţie a punctului indicată de utilizator; practic atunci cînd utilizatorul vrea pentru un motiv oarecare să-şi amintească o anumită poziţie, pune punctul acolo şi apasă o combinaţie de taste. Locul acesta este marcat.

region = zona dintre punct şi marcă În fiecare buffer partea cuprinsă între marcă (unde a fost pusă ultima oară) şi punct se numeşte "regiune". Regiunea este extrem de utilă pentru că multe comenzi funcţionează pe zone de text: de pildă corectură de texte, aranjarea în pagină, capitalizarea, ştergerea, etc. Pe anumite terminale regiunea este înfăţişată cu altă culoare.

kill = a şterge text Emacs ne pune la dispoziţie multe comenzi pentru a şterge părţi de text. În terminologia sa ştergerea unei părţi de text ceva mai mari se numeşte “omorîre” (kill). Omorîrea este interesantă pentru că Emacs păstrează undeva în măruntaiele sale textele care au fost şterse, şi permite accesul la ele, aşa că ele nu sunt pierdute. Pentru a muta o bucată de text dintr-un loc într-altul ea este marcată cu o regiune, omorîtă şi apoi pusă la loc.

yank = a pune la loc un text omorît Acţiunea de a recupera un text omorît se numeşte “to yank”. Emacs încearcă să ţină minte cît se poate de multe texte omorîte (cît are memorie), în ceea ce se cheamă “inelul cu texte omorîte” (kill ring). E un inel pentru că la un moment dat texte omorîte vechi sunt “uitate” şi textele noi omorîte sunt scrise peste ele. Emacs permite extragerea (yank) tuturor textelor din inel.

undo = a des-face ultima acţiune Una din trăsăturile foarte plăcute ale lui Emacs este că ţine minte foarte multe din operaţiile făcute asupra textelor şi permite des-facerea lor. Dacă ai şters părţi mari de text, ai modificat altele, le-ai reordonat şi apoi te-ai răzgîndit, Emacs îţi permite să dai înapoi cît vrei.

mode (major/minor) = modul de lucru Emacs este foarte flexibil: în funcţie de tipul textelor pe care le editaţi permite efectuarea anumitor operaţii cu uşurinţă. De exemplu dacă scrieţi programe permite aranjarea automată în pagină, căutarea variabilelor şi funcţiilor în fişiere, compilarea, etc. Dacă scrieţi texte în limbi “naturale” oferă o suită de comenzi pentru prelucrat fraze, paragrafe, cuvinte. Aceeaşi comandă înseamnă un lucru diferit dacă se scriu programe sau texte.

Practic modul major arată ce comenzi oferă Emacs. Astfel, există suite de comenzi speciale pentru următoarele tipuri de texte (lista este ne-exhaustivă!):

- Limbi omeneşti;
- Limbaje de programare (C, Lisp, Pascal, Fortran, ML, Ada, Simula, asamblare, etc.);
- Scris şi citit poştă electronică;
- Manipulat fişiere şi directoare;
- Comenzi date calculatorului (shell);
- Lucru prin reţea (ftp, www, news);
- LaTeX, nroff;
- Compilare, depanare;
- Calendare şi agende;
- Felurite jocuri;
- etc, etc, etc.

Modurile minore oferă operaţiuni suplimentare şi coexistă cu modurile majore. De exemplu moduri minore oferă salvare automată a fişierelor, aranjare automată în pagină, posibilitatea de a abrevia anumite texte, etc.

fill = a aranja text Operaţia de a aranja textul în pagină în aşa fel încît sa nu aibă liniile prea lungi se numeşte la Emacs “fill” (umplere).

C-x, M-x = Control-x, Meta-x Emacs a fost proiectat să funcţioneze pe aproape orice fel de terminal. Din cauza asta, Emacs încearcă să se descurce fără să folosească taste cu săgeţi sau alte ciudăţenii care nu se găsesc peste tot (dacă le aveţi însă de obicei le puteţi folosi).

Pentru o implementa funcţionalitatea tastelor speciale Emacs foloseşte doar tastele ordinare prezente peste tot (litere, cifre, cîteva semne speciale) plus doar 2 taste speciale. Denumirile lor sunt **CONTROL** şi **META**. Ele nu trebuie să existe întocmai pe tastatura dumneavoastră. “Control” este prezentă pe cele mai multe claviaturi. Meta e mai rară, şi atunci de obicei tasta “Alt” îi ţine locul. Pentru uniformitate însă se foloseşte peste tot numele de “Meta”.

Control şi Meta se folosesc numai în combinaţii cu alte taste, ca şi shift-urile. Notaţia standard în Emacs pentru apăsarea simultană a lui Control **cu** x este `C-x`. Asta înseamnă că se apasă Control, se ţine apăsată şi se apasă apoi tasta x, după care se eliberează amîndouă.

Atenţie: `C-X` este altceva decît `C-x`: prima se apasă cu 3 taste!, Control, Shift şi X (e X mare), iar a doua numai cu Control şi x. Există şi combinaţii `C-M-x`: control, meta şi x simultan (nu le folosesc prea des).

Dacă chiar nu aveţi o tastă pentru Meta, se poate folosi tasta **ESCAPE** pentru acest scop, dar într-un mod diferit. Tasta ESCAPE se mai numeşte “metafy”, pentru că apăsată *separat* face următoarea tastă să se comporte ca şi cum ar fi apăsată cu Meta împreună. Cu alte cuvinte, următoarele acţiuni sunt echivalente:

Apăs Meta	Apăs Escape
Apăs x	Eliberez Escape
Eliberez Meta şi x	Apăs x
	Eliberez x

Anumite comenzi sunt formate din suite de taste. De exemplu comanda `C-x C-s` (care salvează fişierul din bufferul curent) înseamnă să se apese `C-x`, după care să se apese `C-s`. Comanda este complet diferită de `C-x s`. Dacă apăsaţi numai `C-x` Emacs va aştepta o urmare. Dacă ea nu vine suficient de repede, Emacs va scrie în minibuffer ce aţi apăsat: `C-x`.

2 Taste fundamentale

Voi prezenta foarte puţine taste, dar care practic folosesc cam 95% din timp. Voi încerca să arăt ca numele lor sunt foarte bine alese pentru a fi uşor de memorat. Ca dovadă (indirectă) că lucrurile stau aşa, comenzile Emacs pentru editat texte au devenit folosite de din ce in ce mai multe programe interactive: shell-uri (bash, csh), managere de fişiere (git), Netscape, etc. În plus, toate modurile majore încearcă să folosească tastele cam în acelaşi fel: de exemplu "P" să însemne mereu "înapoi" (Previous).

Iată deci regulile mnemonice:

Tasta	Mnemonică (în engleză)	Traducere
p	Previous	Anterior
n	Next	Următor
b	Back	Înapoi
f	Forward	Înainte
v	↓	În jos
e	End	Sfîrşit
a	A	Început
s	Search	Caută
r	Reverse Search	Caută invers
t	Transpose	Interschimbă
d	Delete	Şterge
k	Kill	Omoară
y	Yank	Adu text omorît

Fireşte, anumite comenzi nu au nume atît de sugestive, dar era de aşteptat. Iată ce cred că ar mai trebui ştiut:

- `C-x` este o comandă-prefix foarte folosită. Următoarea comandă indică ce trebuie să se întîmple. În general operaţiile pe fişiere şi pe ferestre încep cu `C-x`.
- Cam toate comenzile care încep cu `C-` acţionează pe unităţi de text (litere, rînduri). Toate comenzile corespunzătoare care încep cu `M-` acţionează pe unităţi de *limbă* (cuvinte, fraze).
- Pentru a opera eficient pe fraze şi paragrafe, Emacs consideră ca frazele se termină *cu două spaţii* după ultimul semn de punctuaţie. Dacă vă obişnuiţi să scrieţi în acest fel veţi profita de Emacs mult mai mult. De asemenea, paragrafele se termină cu un "newline" urmat de orice caracter spaţiu (spaţiu, tab sau alt newline, recomandabilă fiind ultima variantă, care înseamnă o linie liberă.
- Emacs vine cu un help foarte amănunţit despre toate comenzile lui. Acesta se invocă cu `C-h i`. Cînd nu ştiţi cum se face ceva, apelaţi la acest help. După ce aţi stăpînit comenzile de bază (din acest text), veţi avea nevoie relativ rar de help, ca să nu fie o corvoadă.
- Anumite comenzi pot fi precedate de un argument numeric, şi atunci se execută repetat. Argumente numerice se introduc folosind `C-u`. De exemplu, `C-f` înseamnă "înainte un caracter" (forward), iar `C-u 20 C-f` înseamnă "înainte 20 de caractere".
- Dacă aţi tastat o comandă parţial puteţi să o întrerupeţi oricînd apăsînd `C-g`. De asemenea, dacă aveţi o procesare care durează mult, puteţi să o întrerupeţi cu `C-g`.
- Orice comandă are un nume simbolic (un şir de caractere). Orice comandă se poate tasta folosind acest nume simbolic. Unele comenzi nici nu au tasta asociate, ci doar numele.

Pentru a tasta o comandă prin nume, se tastează `M-x nume-comanda ENTER`. Numele sunt destul de sugestiv alese pentru a fi uşor de reamintit. Astfel, în loc de `C-f` puteţi tasta `M-x forward-char`, care înseamnă acelaşi lucru, "înainte un caracter". Desigur, exemplul este idiot, dar ilustrativ. Numele comenzilor este afişat în minibuffer pe măsură ce-l tastaţi; dacă nu vă amintiţi un nume în întregime, ajunge să tastaţi primele litere şi să apăsaţi pe tasta `TAB`; Emacs vă va da indicaţii despre ce ar putea urma.

3 De ce Emacs

Ca să vă îndemn mai tare să folosiţi Emacs, iată o pagină de reclamă fără prea multe explicaţii. De-a lungul timpului veţi învăţa să folosiţi şi să apreciaţi aceste trăsături.

- Ştie să manipuleze dreptunghiuri de text (text delimitat între două linii şi coloane);
- Poate accepta orice comanda prin numele ei (cu `M-x`);
- Are interfaţă la shell (`M-x shell`);
- Are interfaţă pentru compilare (`M-x compile`) care permite corectarea erorilor cu uşurinţă (folosind după compilare `` C-x ` ``);
- Are interfaţă la debuggere (`M-x gdb`);
- Are interfaţă la manualele Unix (`M-x man`);
- Are interfaţă pentru corectarea de texte (`M-x ispell`);
- Are operaţii cu expresii regulate (`M-x replace-regexp`);
- Foloseşte automat sisteme de control a versiunii (RCS, SCCS, VC);
- Manipulează tabele de simboluri pentru programe cu multe fişiere (etags, `M-.`);
- Permite gruparea unor suite de taste sub un singur nume pentru procesări complicate ("keyboard macros");
- Permite scrierea de programe în lisp pentru efectuarea de operaţiuni asupra textelor;
- Are un help foarte sofisticat (`C-h`);
- Merge pe sute de calculatoare diferite arhitectural;
- Este Free Software: poate fi obţinut gratuit, distribuit fără restricţii, modificat, citit;
- Multe, multe alte avantaje.

4 Anexă

Iată pe o pagină, condensate toate lucrurile importante spuse de-a lungul articolului. Cred că dacă tipăriţi pagina asta şi o ţineţi la-ndemînă în timp ce-nvăţaţi Emacs, se poate dovedi foarte utilă.

Tabelul principal este în engleză, din cauza mnemonicilor şi e ordonat pentru a ilustra înrudirile dintre comenzi.

Essential Emacs keys

C-@	Set Mark (also C-SPC)	M-w	Copy region in kill ring
C-a	Beginning of line	C-x 0	Delete this window
M-a	Beginning of sentence	C-x 1	Keep only this window
C-b	Back one char	C-x 2	Split window vertically
M-b	Backward word	C-x 3	Split window horizontally
C-d	Delete one char	C-x b	Select buffer by name
M-d	Delete word	C-x C-b	List all buffers
DEL	Delete previous character	C-x C-c	Exit
M-DEL	Delete previous word	C-x C-f	Find a file and load it
C-e	End of line	C-x C-w	Write in file (save as)
M-e	End of sentence	C-x d	Directory editor (file manager)
C-f	Forward one char	C-x k	Kill this buffer
M-f	Forward word	C-x o	Move to other window
C-g	Abort current command	C-x s	Save all buffers
C-h i	Info help system	C-x u	Undo
C-k	Kill line	C-y	Yank last killed text
M-k	Kill sentence	M-y	Yank previous from kill ring
C-l	Refresh screen	C-z	Stop emacs
C-n	Next line	M-/	Fill word automatically
C-p	Previous line	M--	Same as C-u -1
C-r	Incremental search backward	M-c	Capitalize word
C-s	Incremental search forward	M-h	Make this paragraph region
C-SPC	Set mark here	M-l	Lowercase word
C-t	Transpose characters	M-q	Fill paragraph
M-t	Transpose words	M-SPC	Delete space arount point
C-u	Numeric argument follows	M-u	Uppercase word
C-v	Page down	M-x	Type command by name
M-v	Page up	M-<	Beginning of buffer
C-w	Kill region	M->	End of buffer

Moduri majore importante

text	Text (limbi normale)	gnus	Citit news
c	Limbajul C	tex	TeX
rmail	Citit mail	latex	LaTeX
mail	Scris mail	lisp	Lisp editting
dired	Manipulari fisiere	info	Help hipertext