Wprowadzenie do Gnu Emacsa

Witold Paluszyński
Katedra Cybernetyki i Robotyki
Politechnika Wrocławska
http://www.kcir.pwr.edu.pl/~witold/

2013

Czym jest Gnu Emacs?

Gnu Emacs jest programem do edycji dokumentów tekstowych.
Czym wyróżnia się spośród innych tego typu programów?

- Jest oryginalnie programem terminalowym, i może być używany w trybie **tekstowym, lub okienkowym**. Ma to znaczenie w pracy w połączeniach sieciowych, i w warunkach, gdy system okienkowy z jakichś powodów nie pozwala na poprawne odpalenie edytora okienkowego.
- Jest **bardzo stabilny**.
- Ma wbudowaną i potwierdzoną **odporność na utracenie pliku**. Trzeba nieźle się namęczyć, żeby utracić plik edytowany Emacsem. Emacs rozpoznaje i zapobiega problemom, o których nawet nie wie użytkownik.
- Emacs stara się **nie utracić również niezapisanych jeszcze danych**, nawet jeśli program zostanie zakończony w niekontrolowany sposób.
- Emacs jest **wysoce konfigurowalny**; daleko, daleko bardziej niż jakikolwiek inny edytor tekstowy. Można poznawać tę konfigurację po kawałku, stając się coraz większym ekspertem (i uzależnionym od) Emacsa.

- Ta konfigurowalność, w połączeniu z dobrym wykorzystaniem klawiszy funkcyjnych (skrótów klawiaturowych), pozwala osiągnąć dużą **szybkość pracy i wysoką produktywność**, szczególnie w pracy programistycznej, ale również pisaniu tekstów.
- Emacs posiada **tryby kontekstowe dla wielu języków programowania** i innych formatów (np. LaTeX'a, HTML'a, itp). Niektóre dość rozbudowane, istotnie wspomagają tworzenie dokumentów.
- Emacs ma **wewnętrzny język programowania**, będący prostą wersją Lispu. Pozwala on definiować nowe funkcje edycyjne, które następnie można przypisać klawiszom, i używać jak zwykłych wbudowanych funkcji edycyjnych.
- Emacs stanowił wyznacznik potężnego, niezwykle funkcjonalnego edytora tekstu przez lata, kiedy alternatywne programy nie istniały. Nawet dziś, gdy istnieją całkiem udane i rozwinięte edytory w środowisku Unix/Linux, **szereg niezwykle przydatnych funkcji Emacsa jest w nich niedostępnych**.

Ogólnie, Gnu Emacs jest ciekawym programem, który ma długą historię rozwoju.[1] Związane z nią **są zalety i wady**. Warto znać jedne i drugie.

[1] Oryginalna wersja Emacsa nie była napisana w C, który dopiero równolegle powstawał, tylko w języku do edycji plików TECO. Miał już jednak interfejs podobny do dzisiejszego (choć skromniejszy), i możliwość pisania rozszerzeń w TECO. Pomimo iż TECO był językiem wysokiego poziomu, mówiono, że program w nim napisany stosunkowo najbardziej przypomina kod binarny.

Czym jeszcze jest Gnu Emacs?

Emacs jest tworem szalonych programistów (oryginalnie jednego z największych z nich, Richarda M. Stallmana, jednego z inicjatorów ruchu wolnego oprogramowania), i w miarę jak okazywał się niezwykle przydatnym narzędziem, był rozbudowywany o podsystemy służące różnym celom. Na przykład:

- wbudowane, oparte na shellu i GNU debuggerze, środowisko uruchamiania programów,
- wbudowany browser systemu plików (DIRED),
- wbudowany interfejs poczty elektronicznej, news-ów, i feed-ów RSS,
- wbudowane interfejsy do wielu programów uniksowych, które można uruchamiać w oknach Emacsa,
- wbudowane gry jak tetris, albo wersja programu Eliza.

Te wbudowane podsystemy powodują, że czasami określa się Emacsa mianem mini systemu operacyjnego.

Jednak trzeba pamiętać, że głównym przeznaczeniem, i źródłem przydatności Emacsa jest **głównie jego niezwykła sprawność edycji tekstów**.

Czym nie jest Gnu Emacs?

Przy całej swej przydatności, Emacs ma pewne cechy szczególne, które przynajmniej **na początku utrudniają pracę z nim**.

- Oryginalna filozofia Emacsa zakładała operowanie głównie skrótami klawiaturowymi. Jakkolwiek od zawsze mogły być one indywidualnie definiowane, w latach 80-tych XX-wieku nie istniały powszechne dzisiaj konwencje takie jak ctrl-O (open file), ctrl-S (save file), albo ctrl-C (copy), itp. Użytkownicy Emacsa zawsze używali jego charakterystycznych skrótów.

  Niestety, są one **niekompatybilne z konwencjami Windows-owymi**. I niestety, nikt dziś nie namówi zagorzałych, doświadczonych uniksowców, żeby przedefiniowali domyślne ustawienia na zgodne ze światem Windowsa. (Każdy może jednak dość łatwo zrobić to sam na własny użytek.)

  Emacsa można używać za pomocą strzałek, jednak nie daje to ani wydajnej pracy ani nie pozwala nabywać wprawy. Aby używać Emacsa na poważnie, **trzeba nauczyć się pracować z jego konwencjami klawiszowymi**. Ma to jednak dodatkowe zalety, ponieważ niektóre programy wykorzystują emacsowe konwencje klawiaturowe (np. bash).

- Co więcej, Emacs od początku wprowadził swoją konwencję bufora kasowania, który jest również **niekompatybilny z mechanizmem Copy/Cut/Paste** współczesnych systemów. Powoduje to, że wersja okienkowa Emacsa nie jest w pełni zintegrowana z resztą oprogramowania systemu okienkowego.

  Trwają próby integracji Emacsa, ale inherentne niekompatybilności powodują, że próby te dotąd kończyły się porażką.

  W efekcie, **współpraca Emacsa z systemem okienkowym** (np. operacje myszą), również **ma charakter indywidualny, trochę odmienny** od innych programów.

Powyższe uwarunkowania powodują, że **Emacs nie dla każdego** okaże się tak użytecznym programem jakim w rzeczywistości jest.

Na szczęście nie jest to problem. Istnieje dziś w świecie Uniksa/Linuksa **wiele edytorów do wyboru**, z reguły bardziej przystępnych do użycia. Nie ma powodu, żeby każdy nie znalazł sobie programu, który najlepiej mu odpowiada.

Jeśli jednak ktoś zaczyna **poważniejszą pracę w środowisku Unix/Linux**, i chciałby poznać prawdziwą moc jego narzędzi (w tym również moc uzależniającą), to **powinien dać Emacsowi szansę**, i postarać się go poznać.

Konwencje Gnu Emacsa

Po wystartowaniu, Emacs prezentuje okienko tekstowe z kursorem, którym można poruszać się i modyfikować tekst.

Ważna jest koncepcja wykorzystania klawiatury. W dodatku do zwykłych klawiszy, Emacs wykorzystuje dwa klawisze-modyfikatory: Control i Meta. Daje to dużą liczbę funkcji, które można wywołać jednym przyciśnięciem klawiszy, ale wymaga dwóch albo trzech palców (a często dwóch rąk, np. spróbuj Control-p). W dokumentacji Emacsa używa się skrótów typu C-a, M-f, albo nawet C-M-w symbolizujących klawisze Control-a, Meta-f, oraz Control-Meta-w.

Większość klawiatur na świecie nie ma klawisza Meta. W jego miejsce na ogół wykorzystywany jest jakiś inny klawisz modyfikator, na przykład klawisz Alt (lewy) na klawiaturze PC-towej, albo klawisz $\diamond$ na klawiaturze Sunowskiej.

Nawet gdy dana klawiatura posiada klawisz Meta, albo jego zamiennik, przy połączeniu sieciowym mogą one być inaczej interpretowane na zdalnym systemie. Dlatego w Emacsie od początku przyjęto konwencję, że znaki z Meta mogą być symulowane dwuznakowymi sekwencjami ze znakiem ESCAPE. Np. zamiast Meta-f można nacisnąć ESCAPE a potem f.

Gnu Emacs — poruszanie się w buforze

Emacs ma wiele poleceń służących do przemieszczania kursora.

Podstawowe, znak-po-znaku, lub wiersz-po-wierszu:
C-f (*forward*), C-b (*backward*), C-p (*previous*/góra), C-n (*next*/dół)

Po wyrazach:
M-f (*forward*), M-b (*backward*)

Początek i koniec wiersza:
C-a, C-e

Początek i koniec zdania:
M-a, M-e

Następny, poprzedni ekran:
C-v, M-v

Początek, koniec bufora (pliku):
M-<, M->

Jak widać, jest pewna logika w tych konwencjach, może nie od razu przekonująca, ale trzeba się z nią zgodzić. Co jednak ważniejsze, funkcje te **są szybkie**, można nabrać dużej wprawy w poruszaniu się po buforze.

Gnu Emacs — przeszukiwanie tekstu

W Gnu Emacsie istnieją funkcje przeszukiwania jednorazowego, ale o wiele lepiej od razu spróbować, niezwykle przydatnej i łatwej w użyciu funkcji przeszukiwania inkrementalnego. Najlepiej zrozumieć i opanować te operacje praktycznie.

C-s przeszukuje wprzód a C-r w tył.

Przeszukiwany tekst wprowadza się w minibuforze (obszar pod linijką statusową) i kursor przeskakuje do pierwszego miejsca znalezionego tekstu od razu po wprowadzeniu pierwszych znaków, a w miarę wpisywania większej części poszukiwanego wzorca, przesuwa się do właściwego miejsca.

W trakcie wprowadzania wzorca (np. po dwóch pierwszych literach), można nacisnąć C-s ponownie, co znajduje następne wystąpienie wzorca w tekście. Wtedy można kontynuować wprowadzanie dalszej części wzorca, jeśli znalezione wystąpienie nie jest właściwe. Można też kontynuować naciskanie C-s, jeśli znaleziony został prawidłowy wzorzec, ale nie to wystąpienie o które chodziło.

Można naciskać zamiennie C-s i C-r aby zmienić kierunek przeszukiwania. Można również naciskać klawisz kasowania znaku wstecz (w zależności od tego czy jest to BACKSPACE czy DEL), co powoduje cofanie się do poprzednich wystąpień, lub skracanie poszukiwanego wzorca.

Aby zakończyć przeszukiwanie najłatwiej nacisnąć Enter (czasami zapisywany RET). W tym przypadku kursor pozostaje w znalezionym miejscu. Naciśnięcie jakiegoś innego klawisza funkcyjnego również kończy przeszukiwanie, ale jednocześnie uruchamia związaną z tym klawiszem funkcję. Można również nacisnąć standardowy klawisz przerwania Emacsa C-g, co powoduje wyjście z trybu przeszukiwania i powrót kursora do punktu startu.

Można wznowić poprzednie przeszukiwanie naciskając C-s C-s.

Domyślnie, przeszukiwanie jest *case-independent*. Aby zmienić je na *case sensitive* należy ustawić odpowiednią flagę (patrz dalej).

Przeszukiwanie inkrementalne ma wiele dodatkowych opcji, przyspieszających jego użycie. Na przykład: naciśnięcie C-w w trybie przeszukiwania, powoduje przeszukiwanie wyrazu, przy którym stoi kursor. W trakcie pracy z Gnu Emacsem można nabierać wprawy w jego użyciu używając coraz większej liczby takich opcji.

Gnu Emacs — zastępowanie stringów

W wielu edytorach tekstowych zastępowanie jest operacją spokrewnioną z przeszukiwaniem, i wywołuje się je z jednego formularza. W Emacsie również można zainicjować zastępowanie stringa z ostatniego przeszukiwania, jednak najłatwiej jest wywołać polecenie zastępowania interakcyjnego M-% (query-replace).

Istnieje szereg dodatkowych poleceń, ułatwiających wprowadzanie wzorców do zastępowania, odwoływanie się do poprzednich operacji przeszukiwania i zastępowania, itp. Podobnie jak się to ma z innymi operacjami edycyjnymi, gdy mamy do czynienia z jakąś poważniejszą pracą edytorską, warto przejrzeć dostępne polecenia Emacsa, i wykorzystać te z nich, które daną pracę mogą zasadniczo ułatwić. Do prostych operacji zwykle jednak wystarcza powyższe (plus operacje z wyrażeniami regularnymi, o których poniżej).

Gnu Emacs — przeszukiwanie i zastępowanie na regex

Zarówno przeszukiwanie inkrementalne, jak i zastępowanie interakcyjne mają ciekawy wariant wykorzystujący wyrażenia regularne, gdzie wzorzec do przeszukiwania lub zastępowania jest wyrażeniem regularnym. Wyrażenia regularne ogólnie zajmują znaczące miejsce w większości narzędzi systemu Unix/Linux.

Przeszukiwanie inkrementalne wyrażeń regularnych: C-M-s lub C-u C-s
Interakcyjne zastępowanie wyrażeń regularnych: C-M-%

Same wyrażenia regularne nie będą tu omawiane. Warto tylko wspomnieć o jednym szczególnym elemencie, przydatnym przy pracy programistycznej. Chodzi o przeszukiwanie i zastępowanie **ograniczone do wyrazów**. Na przykład, chcąc zmienić nazwę zmiennej x (np. na x1) w dużym programie, zwykłe zastępowanie może być kłopotliwe, ze względu na występowanie znaku „x" w innych wyrazach. Podanie wyrażenia regularnego \<x\> ogranicza zastępowanie do wystąpień x jako pełnego wyrazu, i zwykle rozwiązuje problem.

Gnu Emacs — region i zaznaczanie

Dla umożliwienia pewnych operacji blokowych Emacs stosuje pojęcie **regionu**, którym jest obszar pomiędzy pozycją kursora, a ustawionym **znakiem** (*mark*).

Znak ustawia się poleceniem C-@ lub C-SPACE co Emacs potwierdza krótkim komunikatem w minibuforze. Przydatne jest też polecenie C-x C-x zamieniające pozycję kursora ze znakiem.

Szereg poleceń Emacsa **automatycznie ustawia znak w pozycji kursora**, np.: start przeszukiwania, skok na początek lub koniec bufora, odzyskanie skasowanego bloku tekstu, albo wstawienie większego bloku tekstu (np. pliku) do bufora, i inne. Operacje te powodują następnie przeniesienie kursora „gdzieś", z zapamiętaniem początkowej pozycji. Wtedy możliwość powrotu przez C-x C-x jest często przydatna.

Warto wypróbować operacje z zaznaczaniem na poleceniach kasowania:

Kasowanie wyrazu wstecz: M-BACKSPACE, lub wprzód: M-d
Kasowanie do końca wiersza: C-k (zwykle dwa razy)
Kasowanie do końca zdania: M-k
Kasowanie regionu (zaznaczenia): C-w
Odzyskiwanie skasowanego (kumulacyjnie) tekstu: C-y

Gnu Emacs — tutorial

Emacs posiada wbudowany tutorial (samouczek) będący plikiem tekstowym, który można czytać, i jednocześnie wykonywać zawarte w nim polecenia. Wykonanie całego tutoriala trwa krótko, ale daje dobre wprowadzenie do podstawowych konwencji i operacji Emacsa.

Ten tutorial istniał od wczesnych wersji Emacsa (TECO), i jego koncepcja wykorzystuje standardowy rozmiar terminala tekstowego (80x24). Warto czytać i wykonywać tutorial w oknie o takim dokładnie rozmiarze, ponieważ w większym okienku treść i polecenia tutoriala mogą nie „trafiać" w to co widzi użytkownik. Tutorial jest tylko dobrze napisanym plikiem tekstowym.

Gnu Emacs — operacje na plikach

Otwarcie pliku w nowym buforze: C-x C-f
Otwarcie pliku w bieżącym buforze: C-x C-v

Zapisanie edytowanego pliku: C-x C-s
Zapisanie pliku pod nadaną nazwą (*Save As*): C-x C-w

Wywołanie tych operacji powoduje wczytanie nazwy pliku w minibuforze, w czasie którego można naciskać klawisz TAB w celu dopełnienia nazwy pliku. (Ponowne naciśnięcie TAB przy dopełnieniu niejednoznacznym powoduje automatyczne otwarcie nowego okna z pełną listą plików.)

Otwarcie pliku nie jest operacją niezbędną, bo często wykonywane jest domyślnie przy starcie, gdy plik do otwarcia jest argumentem wywołania.

Podobnie zapis zmodyfikowanego bufora na pliku — jedna z najważniejszych operacji — jest i tak wywoływane przy zwykłym wyjściu z programu. Emacs nie pozwoli zakończyć programu bez zapisania bufora, albo jawnego, podwójnego potwierdzenia przez użytkownika pełnym słowem (`"yes"`), że chce porzucić zmodyfikowany bufor. W przeciwnym wypadku Emacs sam zapisuje dane z bufora w specjalnym pliku o nazwie #...#.

Gnu Emacs — kończenie lub zawieszanie pracy

Zakończenie pracy i wyjście z programu: C-x C-c (save-buffers-kill-emacs)

Trochę spokrewnioną operacją jest zawieszenie Emacsa: C-z

Ta operacja jest niezwykle użyteczna w trybie tekstowym (program pozostaje zawieszony, a użytkownik odzyskuje kontrolę nad shellem). Jednak w trybie okienkowym jest bezużyteczna, a wręcz zdradziecka, ponieważ powoduje tylko **zminimalizowanie okienka**. W szczególności nie powoduje ona uporczywych prób zapisania pliku, normalnie towarzyszących zamykaniu programu.

Początkujący użytkownicy często nie zauważają tego, sądząc, że program się zakończył, i następnie uruchamiają nowe instancje Emacsa. Poza niepotrzebnym obciążeniem systemu, prowadzi to do sytuacji, w której istnieje wiele instancji Emacsa z otwartym tym samym plikiem w różnych stadiach edycji.

Końcowe wylogowanie sesji i zabicie wszystkich Emacsów mogłoby doprowadzić do równoczesnego nadpisywania plików, i autentycznej utraty danych, gdyby nie fakt, że początkujący użytkownicy na ogół nie mają żadnych cennych danych w swoich plikach ...

Gnu Emacs — operacje na buforach

Normalnie Emacs wczytuje plik do bufora pamięciowego, wykonuje edycję w tym buforze, a na końcu zapisuje zmodyfikowaną zawartość bufora na pliku. (A także okresowo w czasie pracy sam ją zapisuje w pliku tymczasowym.)

Często przydatne jest jednoczesne otwarcie kilku plików w oddzielnych buforach, i poruszanie się między nimi:

Otwarcie pliku w nowym buforze: C-x C-f
Otwarcie pliku w bieżącym buforze: C-x C-v
Wyświetlenie listy buforów: C-x C-b (powoduje otwarcie nowego okna)
Zamknięcie bieżącego bufora: C-x k (z próbą zapisania, jeśli zmodyfikowany)
Przełączenie się do innego wcześniej otwartego bufora: C-x b

Przełączanie do innego bufora wymaga podania jego nazwy, która normalnie jest identyczna z nazwą pliku (z dodanym numerem wersji, jeśli otwartych jest jednocześnie wiele plików z identycznymi nazwami). W trakcie pisania nazwy bufora można użyć dopełniania (TAB), co w przypadku niejednoznaczności może prowadzić do otwarcia nowego okna z listą buforów.

Gnu Emacs — operacje na oknach

Oknem w terminologii Emacsa jest obszar na ekranie, w którym wyświetlana jest zawartość bufora. Okna Emacsa są wyświetlane w kontrolowanym przezeń ekranie terminala tekstowego, lub w okienku systemu okienkowym.

Początkowo zwykle Emacs wyświetla jedno okno (jednak wywołany od razu z kilkoma nazwami plików w argumentach, od razu otwiera je w oddzielnych buforach, i wyświetla w oddzielnych oknach).

Dodatkowo, wywoływanie poleceń pomocy, i dopełnianie poleceń i nazw plików i buforów, powodują automatyczne otwarcie nowego okna, co zawsze dzieje się przez podział okna istniejącego.

Dlatego warto znać podstawowe polecenia kontrolujące zachowanie okien:

Utworzenie nowego okna przez podział bieżącego: C-x 2 (można powtarzać)
Utworzenie nowego okna przez podział bieżącego: C-x 3 (podział horyzontalny)
Zamknięcie wszystkich okien z wyjątkiem bieżącego: C-x 1
Zwiększenie rozmiaru bieżącego okna (kosztem sąsiedniego): C-x ^

Gnu Emacs — polecenia rozszerzone

Wszystkie skróty klawiszowe w Gnu Emacsie wykonują określone polecenia, które posiadają nazwy. Na przykład, przesunięcie kursora o jeden znak do przodu jest realizowane poleceniem Forward-char.

Emacs posiada bardzo wiele poleceń, z których tylko część jest przypisanych do konkretnych klawiszy. Natomiast każde polecenie można wywołać z nazwy, przez polecenie M-x (execute-extended-command). Pisząc nazwę polecenia, realizowane jest dopełnianie przez TAB.

Nie całkiem związane z poleceniami rozszerzonymi, ale ogólne, jest polecenie C-g (keyboard-quit), powodujące przerwanie wykonywania bieżącego polecenia, i powrót do pętli głównej.

Gnu Emacs — argumenty polecenia

Poleceniom Emacsa można zadać argumenty. Do zadawania argumentów z klawiatury służy polecenie C-u (universal-argument).

Przykłady poleceń, którym warto spróbować zadawanie argumentu liczbowego:

C-n
C-v (ciekawe)
M-f
C-k
C-x ^

Gnu Emacs — czytanie dokumentacji

Dokumentacja Emacsa potrzebna jest nie tylko początkującym użytkownikom. Niektóre polecenia (oraz opcje i inne mechanizmy) są niezwykle przydatne, ale jeśli nie używa się ich na co dzień, to przed użyciem trzeba przypomnieć sobie ich dokładny opis. Na szczęście jest to możliwe wewnątrz samego Emacsa.

Dokumentacja polecenia: C-h f (describe-function)
Dokumentacja trybu podstawowego: C-h m (describe-mode)
Dokumentacja zmiennej: C-h v (describe-variable)
Wyświetlenie listy elementów zawierających
    słowo lub wyrażenie regularne: C-h d (apropos-documentation)

Jako dokumentację polecenia Emacsa uzyskujemy opis wbudowanej funkcji Lispu, realizującej dane polecenie. Jedyna różnica jest w ewentualnych argumentach, które w trybie interakcyjnym zadaje się poleceniem C-u (universal-argument). Dokumentacja zwykle zawiera wskazówki na temat zadawania najważniejszych argumentów.

Funkcje Lispu można też wywoływać jawnie (dokładniej: obliczać S-wyrażenia) poleceniem: M-ESC : (eval-expression).

Gnu Emacs — Undo

C-/ — specyfika działania tego mechanizmu

Gnu Emacs — podstawy konfiguracji

Głównym plikiem konfigurującym pracę Emacsa dla danego użytkownika jest plik `.emacs` w katalogu domowym. Zawiera on wyrażenia Lispu, które Emacs interpretuje (oblicza) w trakcie startu. Wyrażenia mogą ustawiać wartości zmiennych, przypisywać komendy do klawiszy, definiować i wywoływać funkcje, wyświetlać komunikaty, itp. Należy jednak uważać — błędne wyrażenie w pliku startowym może uniemożliwić Emacsowi wystartowanie, albo jakąkolwiek poprawną pracę. Zwykle nie ma problemu w skutecznym uruchomieniu Emacsa, wystarczy zmienić nazwę feralnego pliku `.emacs` na inną. Jednak rozwiązanie problemu, tzn. zlokalizowanie błędnego wyrażenia, i jego poprawienie, w przypadku rozbudowanych konfiguracji często nie jest już takie proste.

```
;;; proba zapisania pliku bez koncowego znaku NEWLINE wywoluje dialog
(setq mode-require-final-newline 'ask)
;;; przedefiniowanie skrotow klawiaturowych
(global-set-key "\C-w" 'backward-kill-word)     ;; CTRL-W
(global-set-key "\C-x\C-k" 'kill-region)        ;; CTRL-x CTRL-k
;;; pliki .pl wchodza w tryb Prolog a nie Perl
(setq auto-mode-alist
      (append  '(("\\.pl" . prolog-mode))
               auto-mode-alist))
```

Gnu Emacs — tryby podstawowe

Gnu Emacs posiada pojęcie podstawowego trybu pracy (*main mode*), i przypisuje tryb pracy każdemu otwieranemu buforowi. Tryb pracy odpowiada zwykle edytowanej w buforze treści, np. tryb Text, tryb C, tryb XML, tryb TeX. Tryby pracy posiadają indywidualne ustawienia, właściwe dla danego typu treści, oraz często definiują polecenia specyficzne dla danego trybu.

Emacs zwykle sam ustawia tryb podstawowy każdego bufora na podstawie rozszerzenia nazwy pliku w nim otwieranego (gdy nie zna danego rozszerzenia to otwiera bufor w trybie Fundamental). Jednak czasem trzeba jawnie ustawić tryb podstawowy bufora. Można to zrobić dzięki temu, że każdy tryb podstawowy posiada polecenie Emacsa, ustanawiające ten tryb dla bufora.

Np., ustawienie trybu Perl-Mode dla bieżącego bufora: M-x Perl-Mode RET

Nazwa trybu podstawowego jest zawsze wyświetlana w linijce statusowej (w nawiasach okrągłych), a opis bieżącego trybu można przeczytać poleceniem C-_ m (describe-mode).

Gnu Emacs — tryb C-Mode

Emacs posiada dość nowy i dobrze oprogramowany tryb główny zwany CC Mode obsługujący rodzinę języków o składni podobnej do ANSI C, obejmującą również C++, Javę, i inne. Wariant dla każdego języka wywołuje się zgodnie z konwencją, poleceniami: c-mode, java-mode, itp.

Podstawą jego działania jest definicja składni konkretnego języka, dzięki któremu działają następujące mechanizmy:

- podświetlanie (kolorowanie) składni,
- formatowanie (wcięcia),
- pokazywanie pasujących nawiasów,
- tworzenie komentarzy,
- polecenia przeskoku do początku i końca bieżącego wyrażenia,
- znajdowanie po nazwie konstrukcji głównego poziomu,
- wiele specjalizowanych poleceń i innych mechanizmów.

Oprogramowanie CC Mode jest bardzo rozbudowane i posiada własny podręcznik użytkownika.

Gnu Emacs — tryby dodatkowe

Emacs posiada również tryby dodatkowe (*auxiliary modes*), wprowadzające pewne modyfikacje pracy trybu podstawowego. Bufor może mieć jeden lub więcej włączonych trybów dodatkowych, i można je zarówno włączać jak i wyłączać. Nazwy włączonych trybów dodatkowych mogą pojawiać się w linijce statusowej, jednak nie wszystkie się pojawiają.

Przykłady trybów dodatkowych:

tryb Auto-Fill powoduje automatyczne zawijanie (wstawianie twardych znaków końca wiersza) linijek dłuższych niż ustawiona wartość maksymalna,

tryb Line-Number powoduje wyświetlanie numeru wiersza w linijce statusowej.

Czy dany tryb dodatkowy jest domyślnie włączony czy nie, decyduje jego definicja, oraz ustawienia w pliku konfiguracyjnym.

Gnu Emacs — Abbrevs

Przykładem innego mechanizmu Emacsa, usprawniającego pisanie tekstów, i zrealizowanego w postaci trybu dodatkowego, jest mechanizm Abbrevs. Pozwala on na zdefiniowanie pewnej liczby skrótów, rozwijanych dynamicznie w czasie pisania. Nie chodzi jednak o polecenia klawiaturowe, będące znakami kontrolnymi, ale o bardziej normalne skróty, najczęściej dwu lub trzyliterowe.

Na przykład, pisząc jakieś opracowanie może zdarzyć się, że często będą w nim powtarzane frazy "neural networks" albo "common-sense reasoning". Pisanie ich wielokrotnie jest uciążliwe, a gdy takie długie, często powtarzane frazy pisze się bardzo szybko — łatwo przepalcować. Można zdefiniować skróty np. nn i cs aby były rozwijane do w/w fraz. Po wpisaniu liter nn i spacji, skrót jest automatycznie rozwijany, co znacznie przyspiesza pisanie.

Istnieje szereg poleceń pozwalających na definiowanie takich skrótów przed rozpoczęciem pracy (całej tabeli skrótów), oraz w trakcie pisania dokumentu. Frazy mogą być rozwijane inteligentnie, wprowadzając duże i małe litery zgodnie ze skrótem (np Nn). Istnieją również polecenia pomocnicze, wymuszające rozwijanie skrótów, albo ułatwiające pisanie fraz pochodnych (np. "subneural networks"). Przy kończeniu edycji Emacs pamięta o zapisaniu pliku z tabelą skrótów, jeśli w trakcie pracy zostały dodefiniowane jakieś nowe.

Gnu Emacs — keyboard macro

Keyboard macro (makrodefinicja klawiaturowa??) jest jednym z mocnych i przydatnych mechanizmów Emacsa. Często przychodzi na pomoc w sytuacji, gdy inne schematy edycji są niewystarczające.

Mówiąc najprościej, polega on na zapamiętaniu sekwencji naciśniętych klawiszy, i możliwości ich powtórzenia jako całości. W ten sposób można interakcyjnie opracować jakiś fragment pliku, a następnie szybko powtórzyć całą operację gdzie indziej, włącznie w powtórzeniem wielokrotnym, np. 50 razy.

Siła tego mechanizmu bierze się z niezwykłej prostoty jego użycia, a także z istnienia poleceń, wykonujących operacje edycyjne w bardzo ogólny sposób.

Zacznij zapamiętywanie: C-x (
Zakończ zapamiętywanie: C-x)
Wykonaj ostatnio zdefiniowane *k.macro*: C-x e (z arg. liczbowym – wielokrotnie)

Po zdefiniowaniu *keyboard macro* można nadać mu nazwę i wywoływać jako polecenie. Można także zapamiętać je na pliku, i wykorzystać przy innej edycji.

Gnu Emacs — praca ze znacznikami (TAGS)

Emacs posiada szereg mechanizmów wspomagających pracę z dużymi programami, składającymi się z wielu modułów źródłowych (np. wielu dziesiątek plików lub więcej), których utrzymanie wymaga często wykonania jednej operacji na wielu plikach. Jednym z takich mechanizmów są tablice „znaczników" (*tags tables*). W tym przypadku „znaczniki" są wystąpieniami istotnych elementów programów źródłowych, takich jak: definicje i wywołania funkcji, zmiennych, nazw klas i metod, itp. Tablica znaczników zawiera ich listę dla wszystkich plików, i musi być najpierw utworzona dla całego zestawu plików źródłowych, a w miarę prac nad programami musi być aktualizowana.

Tablicę znaczników dla wielu popularnych języków programowania można utworzyć programem etags, i pomiędzy wywołaniami Emacsa jest ona przechowywana domyślnie w pliku ETAGS w bieżącym katalogu.

Przydatne polecenia ze znacznikami:
Znajdź pierwszą definicję znacznika: M-. (find-tag)
Znajdź kolejną (alternatywną) definicję poprzedniego znacznika: C-u M-.
Wróć do poprzedniego znalezionego znacznika: C-u - M-.

Gnu Emacs — praca z plikami TAGS

Mechanizm TAGS jest przydatny nie tylko ze względu na operacje na samych znacznikach, które obejmują tylko globalne definicje, takie jak: funkcje, typedef, pakiety, klasy i metody. Często potrzebne są proste modyfikacje na zwykłym tekście, które trzeba przeprowadzić na wszystkich plikach źródłowych aplikacji. W takich przypadkach uciążliwe jest kolejne otwieranie wszystkich plików i powtarzanie tych samych operacji wiele razy. Emacs posiada wersje poleceń wielokrotnych (przeszukiwania i zamiany) dla plików określonych przez TAGS.

Przydatne polecenia na plikach zawierających znaczniki:
Przeszukiwanie wyrażeń regularnych w plikach: M-x tags-search RET
Interakcyjna zamiana wyraż. regul. w plikach: M-x tags-query-replace RET
Powtórzenie poprz. polec. przesz./zast. na plikach: M-, (tags-loop-continue)

Powyższe polecenia wykonują zwykłe operacje przeszukiwania i zastępowania, ale **wykonują je na wszystkich plikach należących do pakietu** określonego przez plik TAGS, automatycznie otwierając i przechodząc do kolejnych plików.

Gnu Emacs — sprawdzanie pisowni programem ispell

Gnu Emacs posiada interfejs do wywoływania wielu zewnętrznych programów wykonujących różne operacje na treści wpisywanej w buforach Emacsa. Jednym z bardzo przydatnych jest interfejs wywoływania programu `ispell` do sprawdzania poprawności słownikowej tekstów, z interakcyjnym poprawianiem.

W tym celu, na komputerze musi być zainstalowany jeden z programów do sprawdzania pisowni: `ispell` lub `aspell`. Musi być również zainstalowany słownik języka, którego pisownię chcemy sprawdzać, w standardowej lokalizacji znanej programowi. Dla upewnienia się, można wywołać dany program ręcznie. Pozwala on zlokalizować błędy pisowni, oraz wprowadzać poprawki w plikach. Jednak wygoda tej operacji jest dużo mniejsza niż za pośrednictwem Emacsa.

Podstawowe polecenia Emacsa do sprawdzania/poprawiania tekstów:

Ust.słownika języka polskiego: M-x ispell-change-dictionary RET polish RET
Sprawdzanie i poprawianie tekstu: M-x ispell-buffer RET

Polecenia ispell... dotyczą jednego z powyższych programów, w zależności od tego, który jest zainstalowany na komputerze.

Gnu Emacs — uwagi końcowe

Gnu Emacs jest dobrym narzędziem intensywnie używanym w środowisku Unix/Linux. Jedną z przyczyn jest, że edytory systemowe Uniksa (ed/ex/vi) nie nadają się do intensywnej pracy programistycznej.

Jednak Gnu Emacs **nie jest stałym elementem systemu Unix** i nie można **traktować go jako jedynego narzędzia**, odpowiedniego do wszystkich prac.

1. Żadna instalacja Uniksa, ani wiele współczesnych pakietów instalacyjnych Linuksa, nie zawiera Gnu Emacsa. Jest on programem, który trzeba doinstalować (często jest zawarty w rozszerzonym pakiecie instalacyjnym). W poważnej pracy trzeba liczyć się z tym, że w danej instalacji nie będzie Gnu Emacsa, może nie być dostępu do Internetu, a nawet gdyby był, to nie w każdych warunkach można rozpoczynać pracę od instalacji Gnu Emacsa.

2. W celu zwiększenia bezpieczeństwa pracy, Gnu Emacs tworzy szereg plików pomocniczych w katalogu z edytowanym plikiem. W szczególności zapisuje plik „zapasowy" (ze znakiem tyldy). Jest to bardzo bezpieczne i pożądane dla większości prac, ale nie jest właściwe np. w pracach administracyjnych, gdzie szereg okoliczności może uniemożliwić tworzenie dodatkowych plików (np. brak uprawnień do zapisu w katalogu, brak miejsca na dysku, itp.).

Wniosek: do pewnych prac **konieczne jest czasami użycie edytora vi**, który jest elementem każdej instalacji Uniksa/Linuksa, i edytuje pliki „w miejscu". Szczególnie, vi jest lepszym narzędziem do prac administracyjnych. Trochę paradoksalnie, jego mniejsze możliwości i niska wygoda pracy skłaniają administratorów systemu do wykonywania krótkich edycji, i szybkiego kończenia pracy, co jest ogólnie właściwe i pożądane.

Jednocześnie, ponieważ trudno jest dobrze znać dwa różne edytory, niektórzy programiści wolą używać wyłącznie vi, pomimo iż jest bezwzględnie gorszym narzędziem od Gnu Emacsa. Wyrazem tego jest rozbudowana wersja Gnu edytora vi — vim.