Emacs: C Guide

Alexander Christensen

July 30, 2017

Abstract

Guide til opsætning af C udviklingsmiljø i Emacs, herunder indentering, kompilering, auto-completion, pakkekonfiguration, samt navigation.

1 Introduktion

Denne guide henvender sig primært til studerende på DIKU, som skal til at påbegynde deres andet studieår, og som derfor skal lære at programmere `C`. Der vil blive lagt væk på pakker og konfigurationer, der knytter sig specifikt til C (eller C++) programmering, og mange fundamentale elementer som fx navigation mellem filer, genvejstaster osv. vil ikke blive berørt.

2 Notation

I Emacs jargon betyder `"C-c"` at man trykker "Control c" på sit tastatur, `"M-c"` betyder "Alt c", og `"RET"` betyder "enter". En `buffer` er en åben instans af en fil, en `frame` er et programvindue, og en `frame` kan inddeles i flere `windows / vinduer`, som er pladsholdere for `buffere`. Cursoren kaldes for `point`.

3 Init-filen

Kernen omkring Emacs er den utroligt fleksible init-fil. Hvis Emacs åbnes direkte fra en 'frisk' installation ligner det noget, man vil holde sig langt væk fra! Det er meningen, at Emacs skal have en masse pakker installeret, og en rutineret init-fil kommer let op på flere tusind linjer (den kan deles op over flere filer!).

Så hvor finder man sin init-fil? Svaret afhænger af styresystemet. Flere detaljer kan ses på denne hjemmeside: `https://www.emacswiki.org/emacs/InitFile`.

nyttig info:

`"C-x C-e"` i slutningen af en linje eller blok med E-lisp kode vil evaluere denne linje eller blok.
`"M-x"` → `"eval-buffer"`→ `"<RET>"` vil læse hele init-filen igennem og aktivere (evaluere) alle skrevne konfigurationer. I visse tilfælde er det dog nødvendigt at lukke Emacs og starte det op igen for at se ændringerne - det sker heldigvis ikke særlig ofte.

Efter ændringerne i init-filen er evalueret, kan de aktiveres i eksempelvis en C buffer ved at skrive `"M-X"` → `"c-mode"` → `"<RET>"`.

4 Pakker

Pakker til Emacs kan installeres ved at trykke `"M-x"` → `"list-packages"`. Der kan søges efter pakker med `"C-s"` eller `"C-r"`, og hvis `point` er placeret i starten af linjen for en pakke, kan denne pakke markeres til installation ved at trykke `"i"`, eller markeres til sletning ved at trykke `"d"`. For at udføre alle markerede installationer/afinstallationer tryk `"x"`.

For at få mere udbytte af den indbyggede pakke-manager (læs: flere tilgængelige pakker), da kan der med fordel tilføjes **Melpa**-arkivet til init-filen, som følgende:

elisp - *init.el*

```elisp
;; Add and enable the MELPA package archive
(require 'package)
(add-to-list 'package-archives
             '("melpa" . "http://melpa.milkbox.net/packages/") t)

; activate all the packages (in particular autoloads)
(package-initialize)

; fetch the list of packages available
(unless package-archive-contents
  (package-refresh-contents))
```

Disse indstillinger bør stå allerøverst oppe i init-filen.

5 Customization

I Emacs er der ingen grænser for, hvad der kan ændres på. Et glimrende eksempel er hvis man skriver `"M-x"` → `"customize-face"`→ `"all faces"`→ `"<RET>"`: Så kommer der en lang liste frem over alle `faces` (dvs. farver, skrifttype/størrelser, osv), der kan konfigureres. I Emacs er der ikke noget, der ikke kan konfigureres, og det er ganske sjovt at 'lege med farverne'.

- I stedet for `"all faces"` kan man også vælge `"default"`: så kan man ændre baggrundsfarven i Emacs.

- Alle `customizations` man laver bliver automatisk gemt i init-filen når man trykker *'Save for future sessions'*.

6 Ielm

"Inferior Emacs Lisp Mode". Her kan man afprøve sproget **elisp**, som init-filen er skrevet i. Aktiveres ved at skrive `"M-x"` → `"ielm""<RET>"`. Det er en god øvelse, hvis man gerne vil lære at skrive sine egne udvidelser. (det får man brug for, hvis man har anvendt Emacs længe nok!)

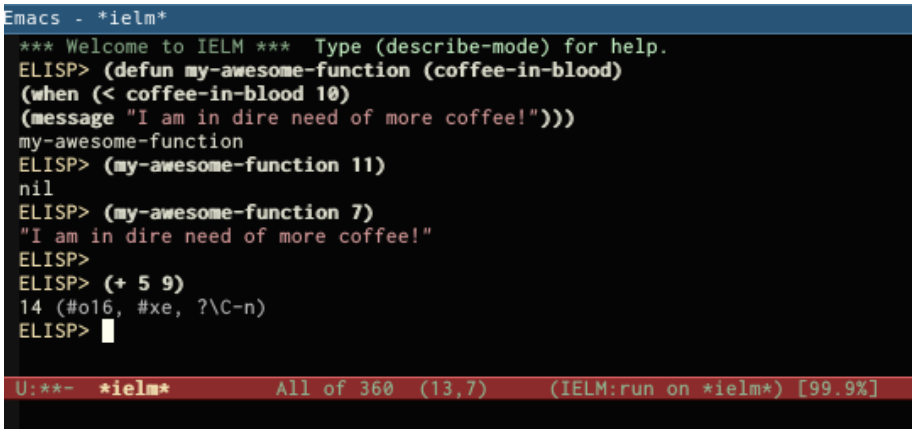

Figure 1: *Coffee scripting* med **Ielm**.

7 Indentering

I en aktiv C buffer, tast `"C-c C-o"`, og mini-bufferen vil vise den variabel, som definerer indenteringen på den pågældende linje. Tast `"RET"`, og du har mulighed for at ændre indenteringen (eller tast `"C-g"` for at afbryde).

Det er værd at bemærke, at denne indentering er afhængig af, om Emacs er indstillet til at indentere med tabs eller spaces! Til CompSys anbefales det at indentere med 2 spaces, og dette opnås ved at tilføje følende linjer til init-filen:

elisp - *init.el*

```
(setq indent-tabs-mode nil)
(setq-default tab-width 2)
```

eksempel:

For at kunne indentere smertefrit med 2 spaces, kunne følgende linjer tilføjes til init-filen:

elisp - *init.el*

```
(defun my-c-mode-hook ()
  (setq-default c-basic-offset 2
                tab-width 2
                indent-tabs-mode nil)
  (c-set-offset 'defun-block-intro 2)
  (c-set-offset 'statement-block-intro 2)
  (c-set-offset 'comment-intro 0)
  (c-set-offset 'func-decl-cont 0)
  )

(add-hook 'c-mode-hook 'my-c-mode-hook)
```

Hvis indentering 'driller', brug da metoden vist ovenfor: Undersøg, hvilken variabel, der definerer den pågældende linjes indentering, og tilføj rettelsen til init-filen.

8 Kompilering

Det kan være et tungt arbejde at kompilere en C fil, især hvis der skal linkes med flere biblioteker, defineres kompiler-version, optimeringsflag, output-fil, etc. Et hårrejsende eksempel på dette kunne være:

```
gcc -std=gnu11 -Wall -Werror -pedantic -o testfile testfile.c
```

Det ville være unødvendigt at skrive dette hver gang i terminalen (selvfølgelig findes der en `Makefile`!). En hurtig Emacs-løsning ville være følgende tilføjelse til init-filen:

elisp - *init.el*

```
(defun my-compile-function ()
  "Define custom compile settings relative to active ‘major-mode‘."
  (interactive)
  (defvar filename)
  (defvar comp-flags)
  (defvar output)

  (setq filename (file-name-nondirectory buffer-file-name))

  ;; c-mode
  (when (member major-mode '(c-mode))
    (setq comp-flags "-std=gnu11 -Wall -Werror -pedantic ")
    (setq output (concat "-o " (file-name-sans-extension filename) " "))
    (setq compile-command (concat "gcc " comp-flags output filename)))

  ;; ... other programming modes might be included here ...

  ;; COMPILE:
  (call-interactively 'compile)
  )

(global-set-key (kbd "<f5>") 'my-compile-function)
```

`"(interactive)"` *betyder i denne sammenhæng at funktionen vil optræde i menuen når man trykker* `"M-x"`.

Det anbefales, at læseren afsætter lidt tid til at forstå, hvordan denne kommando fungerer. Prøv at køre den!

9 Shell-Pop

Pakken `'shell-pop'` kan være smart hvis man vil hurtigt vil køre en terminalkommando inde fra den mappe, hvori den aktive buffer er hentet fra. Med et tryk på en knap (her er valgt `"<f2>"`) popper der en terminal frem neden under den aktive buffer, og med et tryk på samme knap forsvinder den igen.

elisp - *init.el*

```
(require 'shell-pop)
(global-set-key (kbd "<f2>") 'shell-pop)
```

Husk at installere pakken via Emacs' pakke-manager først.

10 Auto-completion

11 Øvrige referencer

http://tuhdo.github.io/c-ide.html#sec-2