KOBE UNIVERSITY

Emacs と Emacs Lisp の初歩

2016.10.07

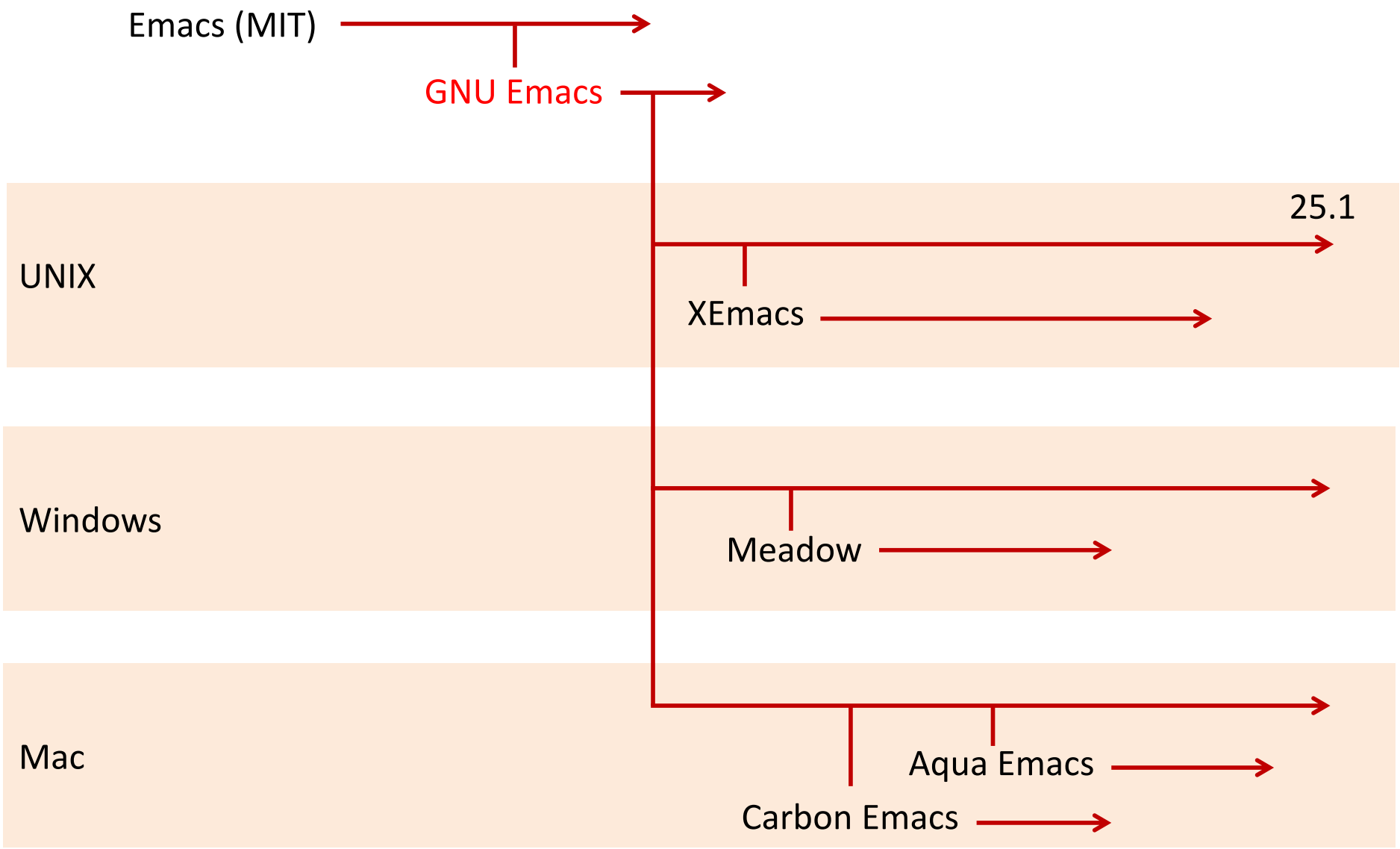
小川 和律

Emacs

Wikipedia jp より抜粋

- Emacs (イーマックス) およびその派生物は、その拡張性を特徴としたテキストエディタのファミリーである。Emacs の中で最も広く使われている派生物は GNU Emacs であるが、そのマニュアルには Emacs を「the extensible, customizable, self-documenting, real-time display editor」(拡張およびカスタマイズが可能で、自己文書化を行い、リアルタイム表示を行うエディタ) であると説明されている。最初の Emacs 開発は 1970 年代中盤に開始され、2016 年現在現在も続いている。

Emacs (日本語環境) の派生



KOBE UNIVERSITY

Emacs の構造

Emacs の特徴

- 拡張・カスタマイズ可能
- 自己文書化
- リアルタイム表示

- 編集テキストに色がつく
- キーバインド
- …

多機能で、拡張カスタマイズ性が高い。

ユーザ・愛好家が多く、多くの資産があり、移植・対応が早い。

“ ユーザーの中には Emacs 内部からテキスト編集だけでなく
ほとんど全ての作業を行うことができることに気づいた者もいる ”

KOBE UNIVERSITY

Emacs キーバインド

C-　コントロールを押しながら
M-　メタを押しながら (esc を押して離してから)

特にカーソル移動と基本編集に関するものは業界スタンダードの一つになっている。

- C-f (右)、C-b (左)、C-p (上)、C-n (下)
- C-a (行頭)、C-e (行末)
- C-d (文字削除)
- C-k (行末までカット)
- C-y (ヤンク (ペースト))
- C-v (ページ送り)

C-@ (C-space) でマークすると、その場所から現在のカーソルまでが選択領域になる。

- C-w (選択領域をカット)、M-w (選択領域をコピー)
- C-x r k (選択矩形領域をカット)、C-x r y (矩形領域をヤンク)
- C-x r t (矩形領域に挿入)

すべてのキーバインドは特定の関数 (コマンド) に結びついている。
コマンドの呼び出しは「M-x コマンド名」(タブ補完が効く)。

当然ですが、キーバインドはカスタマイズできる。

KOBE UNIVERSITY

Emacs のユーザ設定

- 従来は ~/.emacs に書き込む方式だった。
- 現在は ~/.emacs.d/ に Emacs 関連ファイルをまとめて
  ~/.emacs.d/init.el に基本設定を書き込む方式が標準。
- ユーザ設定は Emacs Lisp で記述する。
  基本は、既存の変数に値を代入していくだけ。
- Emacs 立ち上がり時に init.el が読まれて、
  上から順番に実行される。
- *.elc は *.el をバイトコンパイルしたもの。
  近年はあまりメリットがないが、elc を準備する方も多い。

Emacs Lisp

- 括弧を使うポーランド記法言語 LISP から派生した Emacs 用言語。

中置記法	ポーランド記法
`1 + 1`	`+ 1 1`

C 言語算術式	C 言語関数	Emacs Lisp
`x = 1 + 1`	`x = plus(1, 1)`	`(setq x (+ 1 1))`

- シンボルと値

`(setq x (+ 1 1))`　　変数 x について、シンボルは「x」。その値は「2」。

- set と setq

`(setq x 1)` = `(set 'x 1)`　　クオートはシンボルを指すための記号。
クオートをつけないと評価
(中の値を参照) される。

直前のS式は C-x-e (eval-last-sexp) で評価、C-j (eval-print-last-sexp) で評価 & print できる。

リスト

- Emacs Lisp のリスト

  (apple orange lemon banana)　　実装上はS式と同等。

- リストのための基本関数

  (car ‘(apple orange lemon))　　→ apple

  (cdr ‘(apple orange lemon))　　→ (orange lemon)

  (cons ’banana ’(apple orange lemon))　→ (banana apple orange lemon)

制御構文

- (if (条件) (真) (偽))
- (while (条件) (内容))
- dolist
- dotimes
- ....

一般的な制御構文は概ね実装されている。

外部ファイル呼び出しとメジャーモード

- Emacs には状況や編集するテキストの種類ごとにメジャーモードがあり、
  通常は1ないし複数の外部ファイルに記載されている。

- `(load gnuplot-mode)`
  この行で強制的に外部ファイルを読んで全て評価する。
  load-path で定義されているディレクトリに対して、
  gnuplot-mode.elc, gnuplot-mode.el の順に探す。

- `(require 'gnuplot-mode)`
  load と似ているが、同じファイルは2度目は読まない。
  呼ばれるファイル側で provide を書いておく必要があり。

- `(autoload 'gnuplot-mode "gnuplot-mode" nil t)`
  関数が実行された時に、初めて外部ファイルを探して読む。
  2度目は読まない。

KOBE UNIVERSITY

メジャーモードを作る

- メジャーモードに何を期待するか？

  文法に従って文字に色をつけたい。
  → モードファイル内で正規表現で定義して font-lock に登録する。

  モード固有の関数を定義したい。
  → モードファイル内で defun。

  特定のキーバインドで特定の動作をさせたい。
  → 定義した関数を define-key して、use-local-map に登録。

  モード固有のメニューを表示させたい。
  → easy-menu という簡単メニュー作成パッケージがある。

- 自分が作りたいモードと “似たような” モードのファイルを探してきて
  中身を改造すると、楽に作れる。

- 作ったファイルを load-path がかかっているディレクトリに放り込んで、
  init.el に autoload を書いておく。
  ~/.emacs.d/site-litp/ 以下に置いて load-path を張っておくと良い。

KOBE UNIVERSITY

最近 (?) 流行りの色々なモード

- auto-complete
  プログラム作成中に、変数名や関数をタブ補完できる。

- anything
  統合インターフェイス。なんというか、いろいろ便利。

- flycheck
  プログラム作成中に、常時文法エラーを表示する。

- wdired
  dired 画面からパーミッションやファイル名を直接編集できるキワモノ。

- moccur-edit
  grep の結果を表示して、それを直接編集できる。

- color-theme
  モード横断的に色セットを設定できる。