

Emacs Configuration

Nguyễn Đức Hiếu

Ngày 17 tháng 5 năm 2017

The whole thing is driven by the following `.emacs`

```
;; init.el for this setup. Must use Emacs 24
(org-babel-load-file
 (expand-file-name "init.org"
  "/media/Logical_Drive/Emacs/Settings"))
```

Path configurations

```
(require 'package)
(add-to-list 'package-archives '("melpa" . "http://melpa.org/packages/"))

(package-initialize)
```

Appearance

```
;; Initialize Emacs full screen
(add-to-list 'initial-frame-alist '(fullscreen . maximized))

;; Cursor type
(setq-default cursor-type 'bar)

;; Default font
(require 'unicode-fonts)
```

```

(unicode-fonts-setup)
(set-frame-font "DejaVu Sans Mono 10" nil t)

;; Set themes
(add-hook 'after-init-hook (lambda () (load-theme 'gruvbox)))

;; Global font-lock mode
(setq global-font-lock-mode t)

;; fontify code in code blocks
(setq org-src-fontify-natively t)
;; (set-face-attribute 'org-block nil :foreground "white")
(set-face-attribute 'org-block nil
  :foreground "#ffffff")

;; Enable line number and column number
(setq column-number-mode t)

;; Disable tool bar, menu bar, and scroll bar
(tool-bar-mode -1)
(scroll-bar-mode -1)

;; Paren mode
(show-paren-mode t)
(setq show-paren-delay 0)

```

Editing

```

;; Delete marked region when input
(delete-selection-mode 1)

```

```

;; "Yes or no"? Too much writing
(defalias 'yes-or-no-p 'y-or-n-p)

;; Auto close bracket insertion.
(electric-pair-mode 1)
(setq electric-pair-pairs '(
  (?\" . ?\")
  (?\(. . ?\))
  (?\{ . ?\})
) )

(when (fboundp 'electric-indent-mode) (electric-indent-mode -1))

;; Code completion
(require 'company)
(require 'company-files)
(add-hook 'after-init-hook 'global-company-mode)

(setq company-selection-wrap-around t
      company-tooltip-align-annotations t
      company-idle-delay 0.36
      company-minimum-prefix-length 2
      company-tooltip-limit 10)

;; Define function: fill character to 80
(defun fill-to-end (char)
  (interactive "cFill Character:")
  (save-excursion
    (end-of-line)
    (while (< (current-column) 80)
      (insert-char char))))

```

Miscelanous

```
;; Auto-revert mode
(global-auto-revert-mode 1)
(setq auto-revert-interval 0.5)

;; Backup stored in /tmp
(setq backup-directory-alist
      '((".*" . ,temporary-file-directory)))
(setq auto-save-file-name-transforms
      '((".*" ,temporary-file-directory t)))

;; Delete old backup
(message "Deleting old backup files...")
(let ((week (* 60 60 24 7))
      (current (float-time (current-time))))
  (dolist (file (directory-files temporary-file-directory t))
    (when (and (backup-file-name-p file)
               (> (- current (float-time (fifth (file-attributes file))))
                   week))
      (message "%s" file)
      (delete-file file))))

;; Startup
(find-file "/media/Logical_Drive/Emacs/Settings/init.org")
```

Helm config

```
(require 'helm)
(require 'helm-config)
```

```

;; The default "C-x c" is quite close to "C-x C-c", which quits Emacs.
;; Changed to "C-c h". Note: We must set "C-c h" globally, because we
;; cannot change 'helm-command-prefix-key' once 'helm-config' is loaded.
(global-set-key (kbd "C-c h") 'helm-command-prefix)
(global-unset-key (kbd "C-x c"))

(define-key helm-map (kbd "<tab>") 'helm-execute-persistent-action) ; rebind tab to r
(define-key helm-map (kbd "C-i") 'helm-execute-persistent-action) ; make TAB work i
(define-key helm-map (kbd "C-z") 'helm-select-action) ; list actions us
(global-set-key (kbd "C-x C-f") 'helm-find-files)

(when (executable-find "curl")
  (setq helm-google-suggest-use-curl-p t))

(setq helm-split-window-in-side-p t ; open helm buffer inside current window
      helm-move-to-line-cycle-in-source t ; move to end or beginning of source wh
      helm-ff-search-library-in-sexp t ; search for library in 'require' and '
      helm-scroll-amount 8 ; scroll 8 lines other window using M-<
      helm-ff-file-name-history-use-recentf t
      helm-echo-input-in-header-line t)

(defun spacemacs//helm-hide-minibuffer-maybe ()
  "Hide minibuffer in Helm session if we use the header line as input field."
  (when (with-helm-buffer helm-echo-input-in-header-line)
    (let ((ov (make-overlay (point-min) (point-max) nil nil t)))
      (overlay-put ov 'window (selected-window))
      (overlay-put ov 'face
        (let ((bg-color (face-background 'default nil)))
          '(:background ,bg-color :foreground ,bg-color)))
      (setq-local cursor-type nil))))

(add-hook 'helm-minibuffer-set-up-hook
  'spacemacs//helm-hide-minibuffer-maybe)

(setq helm-autoresize-max-height 0)
(setq helm-autoresize-min-height 30)
(helm-autoresize-mode 1)

```

```
(helm-mode 1)
```

```
(global-set-key (kbd "C-x b") 'helm-buffers-list)
(global-set-key (kbd "M-x") 'helm-M-x)
(setq helm-M-x-fuzzy-match t)
```

ESS

```
(require 'ess-site)
(require 'ess-rutils)

(add-hook 'inferior-ess-mode-hook (font-lock-mode))
;; Indentation style
(setq ess-default-style 'RStudio)

;; Remap "<-" key to M-- instead of smart bind to "_"
(ess-toggle-underscore nil)
(define-key ess-mode-map (kbd "M--") 'ess-smart-S-assign)

;; Hot key C-S-m for pipe operator in ESS
(defun then_R_operator ()
  "R - %>% operator or 'then' pipe operator"
  (interactive)
  (just-one-space 1)
  (insert "%>%")
  (reindent-then-newline-and-indent))
(define-key ess-mode-map (kbd "C-S-m") 'then_R_operator)
(define-key inferior-ess-mode-map (kbd "C-S-m") 'then_R_operator)

;; Truncate long lines
(add-hook 'special-mode-hook (lambda () (setq truncate-lines t)))
(add-hook 'inferior-ess-mode-hook (lambda () (setq truncate-lines t)))

;; Describe object
```

```

(setq ess-R-describe-object-at-point-commands
  '(("str(%s)")
    ("print(%s)")
    ("summary(%s, maxsum = 20)")))

(defun ess-rshiny ()
  "Compile R markdown (.Rmd). Should work for any output type."
  (interactive)
  ;; Check if attached R-session
  (condition-case nil
    (ess-get-process)
    (error
     (ess-switch-process)))
  (let* ((rmd-buf (current-buffer)))
    (save-excursion
      (let* ((sprocess (ess-get-process ess-current-process-name))
             (sbuffer (process-buffer sprocess))
             (buf-coding (symbol-name buffer-file-coding-system))
             (R-cmd
              (format "library(rmarkdown); rmarkdown::run(\"%s\")"
                      buffer-file-name)))
        (message "Running shiny on %s" buffer-file-name)
        (ess-execute R-cmd 'buffer nil nil)
        (switch-to-buffer rmd-buf)
        (ess-show-buffer (buffer-name sbuffer) nil))))))

(define-key ess-mode-map "\M-nr" 'ess-rshiny)

```

Elpy

```

(elpy-enable)
(setq elpy-rpc-python-command "python3")
(setq python-shell-interpreter "python3")

(require 'pymacs)

```

AUCTeX

```
(setq TeX-auto-save t)
(setq TeX-parse-self t)
;; Enable query for master file
(setq-default TeX-master nil)
(setq TeX-save-query nil)
(setq TeX-PDF-mode t)
(setq font-latex-fontify-sectioning 'color)
(setq font-latex-fontify-script nil)

;; Word-wrap
(add-hook 'TeX-mode-hook (lambda () (setq-default word-wrap t)))
```

Polymode

```
(require 'polymode)
(require 'poly-R)
(require 'poly-markdown)
(add-to-list 'auto-mode-alist '("\\.md" . poly-markdown-mode))
(add-to-list 'auto-mode-alist '("\\.Snw$" . poly-noweb+r-mode))
(add-to-list 'auto-mode-alist '("\\.Rnw$" . poly-noweb+r-mode))
;; (add-to-list 'auto-mode-alist '("\\.Rmd$" . poly-markdown+r-mode))
(add-to-list 'auto-mode-alist '("\\.Rmd$" . poly-markdown-mode))
(add-to-list 'auto-mode-alist '("\\.rapport$" . poly-rapport-mode))
(add-to-list 'auto-mode-alist '("\\.Rhtml$" . poly-html+r-mode))
(add-to-list 'auto-mode-alist '("\\.Rbrew$" . poly-brew+r-mode))
(add-to-list 'auto-mode-alist '("\\.Rcpp$" . poly-r+c++-mode))
(add-to-list 'auto-mode-alist '("\\.cppR$" . poly-c++r-mode))
```

Pdf-tools

```
(pdf-tools-install)
(setq pdf-view-display-size "fit-page")
(setq auto-revert-interval 0)
```



```
(setq ess-pdf-viewer-pref "emacsclient")  
(setq TeX-view-program-selection '((output-pdf "PDF Tools")))
```

Magit

```
;; Currently magit cause some error when auto revert mode is on  
(setq magit-auto-revert-mode nil)
```

Draft

Settings in this section are not yet organized but are being used