Emacs User Config

Kevin Hong

September 5, 2016

Contents

1 Premble

1.1 Put this in User-config

```
;;(org-babel-load-file "~/Google Drive/_spacemacs/emacs_settings.org")
```

2 Essential

2.1 Basic Settings

2.1.1 Modes

```
(setq which-key-separator " ") ;; setting which-key
(setq which-key-max-description-length 20)
(require 'spaceline-config) ;; setting spaceline
```

```
(spaceline-spacemacs-theme)
(setq linum-format "%2d ")
(setq inhibit-splash-screen t
      inhibit-startup-message t
      inhibit-startup-echo-area-message t)
(add-hook 'org-mode-hook 'visual-line-mode)
(global-git-commit-mode t)
(global-company-mode)
(autopair-global-mode t)
(delete-selection-mode 1) ;; delete the selected region when entering text
(menu-bar-mode -1)
(tool-bar-mode -1)
(when (boundp 'scroll-bar-mode)
  (scroll-bar-mode -1))
(setq visual-line-fringe-indicators '(left-curly-arrow right-curly-arrow))
;; add golden ratio mode
;;(require 'golden-ratio)
;;(golden-ratio-mode 1)
(global-hungry-delete-mode)
;;(setq-default left-fringe-width nil)
;; (setq-default indent-tabs-mode nil)
(eval-after-load "vc" '(setq vc-handled-backends nil))
(setq vc-follow-symlinks t)
(setq large-file-warning-threshold nil)
(setq split-width-threshold nil)
(put 'narrow-to-region 'disabled nil)
(add-hook 'web-mode-hook 'rainbow-mode)  ;; hook rainbow-mode to the html mode as defau
(global-set-key "\C-cg" 'writegood-mode)
(setq-default git-enable-magit-svn-plugin t)
(global-auto-revert-mode t)
```

2.1.2 Aesthetics

```
(add-to-list 'default-frame-alist '(height . 50))
(add-to-list 'default-frame-alist '(width . 116))
(setq-default line-spacing 0.1)
(setq-default line-height 1.1)
(setq-default left-fringe-width  12)
(setq-default right-fringe-width 12)
(spacemacs//set-monospaced-font   "PragmataPro" "Hiragino Sans GB" 18 14);; set Chinese
```

```
(setq ns-use-srgb-colorspace nil) ;; turn off srgb color
;; (setq-default 'cursor-type 'hbar) ;; change cursor type
```

2.1.3 Helm find file anywhere

```
(use-package helm
  :ensure t
  :init
  (progn
    (require 'helm-config)
    ;; limit max number of matches displayed for speed
    (setq helm-candidate-number-limit 6)
    ;; ignore boring files like .o and .a
    (setq helm-ff-skip-boring-files t)
    ;; replace locate with spotlight on Mac
    (setq helm-locate-command "mdfind -name %s %s"))
  :bind (("C-x f" . helm-for-files)))
```

2.1.4 Scroll type

```
  (setq scroll-step            1
        scroll-conservatively  10000)
;; this sets scroll type, C+l will recenter the buffer.
```

2.1.5 Ask before closing

```
(defun ask-before-closing ()
  "Ask whether or not to close, and then close if y was pressed"
  (interactive)
  (if (y-or-n-p (format "Sure you want to exit Emacs, Kevin? "))
      (if (< emacs-major-version 24)
          (save-buffers-kill-terminal)
        (save-buffers-kill-emacs))
    (message "Kevin has canceled exiting Emacs.")))
(when window-system
  (global-set-key (kbd "C-x C-c") 'ask-before-closing))
```

2.1.6 Neotree

```
(global-set-key [f8] 'neotree-toggle)
(setq neo-smart-open t)
```

```
(setq neo-theme 'ascii)
(setq neo-create-file-auto-open t)
```

2.1.7 Ace window kbd

```
(global-set-key (kbd "M-p") 'ace-window)
(setq aw-keys '(?a ?s ?d ?f ?g ?h ?j ?k ?l))
```

2.1.8 Pandoc

```
(custom-set-variables
 '(markdown-command "~/anaconda/bin/pandoc"))
(add-hook 'markdown-mode-hook 'pandoc-mode)
(add-hook 'pandoc-mode-hook 'pandoc-load-default-settings)
```

2.1.9 Reveal in OSX finder kbd

```
(global-set-key (kbd "C-c z") 'reveal-in-osx-finder)
```

2.1.10 Fly check ignore

```
;;set up fly-check to ignore the E501 error
(setq-default flycheck-flake8-maximum-line-length 160)
```

2.1.11 Show-paren-mode settings

```
(show-paren-mode t)
(setq show-paren-delay 0)
(setq show-paren-style 'parenthesis)
(set-face-background 'show-paren-match (face-background 'default))
(set-face-foreground 'show-paren-match "maroon")
(set-face-attribute 'show-paren-match nil
                    :weight 'ultra-bold
                    :underline nil
                    :overline nil)
```

2.1.12 Undo Tree

```
(global-undo-tree-mode)
(global-set-key (kbd "M-/") 'undo-tree-visualize)
```

2.1.13 Multiple cursors

```
(global-set-key (kbd "C->") 'mc/mark-next-like-this)
(global-set-key (kbd "C-<") 'mc/mark-previous-like-this)
(global-set-key (kbd "C-c C->") 'mc/mark-all-like-this)
```

2.1.14 Delete current line

```
(global-set-key (kbd "M-9") 'kill-whole-line)
```

2.1.15 Flycheck for prose-lint

```
;;set up flycheck for proselint
(require 'flycheck)
(flycheck-define-checker proselint
  "A linter for prose."
  :command ("proselint" source-inplace)
  :error-patterns
  ((warning line-start (file-name) ":" line ":" column ": "
            (id (one-or-more (not (any " "))))
            (message (one-or-more not-newline)
                     (zero-or-more "\n" (any " ") (one-or-more not-newline)))
            line-end))
  :modes (text-mode markdown-mode gfm-mode org-mode))

(add-to-list 'flycheck-checkers 'proselint)
(add-hook 'markdown-mode-hook #'flycheck-mode)
(add-hook 'gfm-mode-hook #'flycheck-mode)
(add-hook 'text-mode-hook #'flycheck-mode)
(add-hook 'org-mode-hook #'flycheck-mode)
```

2.1.16 User Info

```
(setq user-full-name "Yili Hong"
      user-mail-address "yili.hong@outlook.com"
      calendar-location-name "Tempe, AZ")
```

2.1.17 Shackle Mode

```
(shackle-mode 1)
(setq shackle-rules '(("\\`\\*helm.*?\\*\\'" :regexp t :align t :ratio 0.4)))
(push '("*osx-dictionary*" :width 0.4 :position right) popwin:special-display-config)
```

2.1.18 Fancy git icon

```
(defadvice vc-mode-line (after strip-backend () activate)
  (when (stringp vc-mode)
    (let ((gitlogo (replace-regexp-in-string "^ Git." "  " vc-mode)))
      (setq vc-mode gitlogo))))
```

2.2 Dictionary

2.2.1 English

```
(setq osx-dictionary-dictionary-choice (list "English" "English Thesaurus"))
(global-set-key (kbd "C-c d") 'osx-dictionary-search-pointer)
(global-set-key (kbd "C-c i") 'osx-dictionary-search-input)
```

2.2.2 Youdao

```
(global-set-key (kbd "C-c Y") 'youdao-dictionary-search-at-point+)
(global-set-key (kbd "C-c y") 'youdao-dictionary-search)
(push '("*Youdao Dictionary*" :width 0.4 :position right) popwin:special-display-config
(setq youdao-dictionary-search-history-file "~/.emacs.d/.youdao")
(setq youdao-dictionary-use-chinese-word-segmentation t)
```

2.3 Diminish

```
(when (require 'diminish nil 'noerror)
  (require 'diminish)
  ;; Hide jiggle-mode lighter from mode line
  (diminish 'jiggle-mode)
  ;; Replace abbrev-mode lighter with "Abv"
  (diminish 'abbrev-mode "Abv")
  (diminish 'projectile-mode "p")
  (diminish 'holy-mode)
  (diminish 'company-mode "c")
  ;;(diminish 'autopair-mode "")
  (diminish 'autopair-mode "ap")
  (diminish 'which-key-mode "wk")
  ;;(diminish 'which-key-mode "")
  (diminish 'reftex-mode "ref")
  ;;(diminish 'reftex-mode "")
  (diminish 'visual-line-mode "")
  (diminish 'hungry-delete-mode)
```

```
  (diminish 'golden-ratio-mode)
  (diminish 'anzu-mode "")
  (diminish 'isearch-mode)
  (diminish 'magic-latex-buffer "")
  (diminish 'iimage-mode "")
  ;;(diminish 'flycheck-mode "")
  ;;(diminish 'python-mode "\f156")
  (eval-after-load "yasnippet"
    ;;'(diminish 'yas-minor-mode "")
    '(diminish 'yas-minor-mode "y")))
```

2.4 Search

2.4.1 Anzu

```
(global-anzu-mode +1)
(setq anzu-cons-mode-line-p nil) ;; avoid anzu info showing twice on spaceline
(set-face-attribute 'anzu-mode-line nil
                    :foreground "maroon" :weight 'bold)

(custom-set-variables
 '(anzu-mode-lighter "")
 '(anzu-deactivate-region t)
 '(anzu-search-threshold 1000)
 '(anzu-replace-threshold 50)
 '(anzu-replace-to-string-separator " => "))

(global-set-key [remap query-replace] 'anzu-query-replace)
(global-set-key [remap query-replace-regexp] 'anzu-query-replace-regexp)
```

2.4.2 Search Web

```
;;(setq w3m-user-agent "Mozilla/5.0 (Linux; U; Android 2.3.3; zh-tw; HTC_Pyramid Build/
 ;; awesome wikipedia search
 (defun wikipedia-search (search-term)
   "Search for SEARCH-TERM on wikipedia"
   (interactive
    (let ((term (if mark-active
                    (buffer-substring (region-beginning) (region-end))
                  (word-at-point))))
      (list
```

```
      (read-string
       (format "Wikipedia (%s):" term) nil nil term)))
   )
  (browse-url
   (concat
    "http://en.m.wikipedia.org/w/index.php?search="
    search-term
    ))
  )

;;when I want to enter the web address all by hand
(defun open-a-website (site)
  "Opens site in new w3m session with 'http://' appended"
  (interactive
   (list (read-string "Enter website address: http://" nil nil "scholar.google.com/cit
  (browse-url
   (concat "http://" site)))
```

2.5 Dired

```
;; (defvar ao/v-dired-omit t
;;   "If dired-omit-mode enabled by default. Don't setq me.")

;; (defun ao/dired-omit-switch ()
;;   "This function is a small enhancement for `dired-omit-mode', which will
;;  \"remember\" omit state across Dired buffers."
;;   (interactive)
;;   (if (eq ao/v-dired-omit t)
;;       (setq ao/v-dired-omit nil)
;;     (setq ao/v-dired-omit t))
;;   (ao/dired-omit-caller)
;;   (when (equal major-mode 'dired-mode)
;;     (revert-buffer)))

;; (defun ao/dired-omit-caller ()
;;   (if ao/v-dired-omit
;;       (setq dired-omit-mode t)
;;     (setq dired-omit-mode nil)))

;; (defun ao/dired-back-to-top()
```

```
;;   "Move to the first file."
;;   (interactive)
;;   (beginning-of-buffer)
;;   (dired-next-line 2))

;; (defun ao/dired-jump-to-bottom()
;;   "Move to last file."
;;   (interactive)
;;   (end-of-buffer)
;;   (dired-next-line -1))
```

3 Latex

3.1 Latex Path

```
(let ((my-path (expand-file-name "/usr/local/texlive/2015/bin/x86_64-darwin/")))
  (setenv "PATH" (concat my-path ":" (getenv "PATH")))
  (add-to-list 'exec-path my-path))
```

3.2 Basic Setting

```
(add-hook 'LaTeX-mode-hook 'turn-on-reftex)   ; with AUCTeX LaTeX mode
(autoload 'reftex-mode     "reftex" "RefTeX Minor Mode" t)
(autoload 'turn-on-reftex  "reftex" "RefTeX Minor Mode" nil)
(autoload 'reftex-citation "reftex-cite" "Make citation" nil)
(autoload 'reftex-index-phrase-mode "reftex-index" "Phrase mode" t)
(add-hook 'latex-mode-hook 'turn-on-reftex)   ; with Emacs latex mode
(setq reftex-enable-partial-scans t)
(setq reftex-save-parse-info t)
(setq reftex-use-multiple-selection-buffers t)
(setq reftex-plug-into-AUCTeX t)
```

3.3 Modify Auctex Behavior

```
(use-package auctex
  :ensure t
  :mode ("\\.tex\\'" . latex-mode)
  :commands (latex-mode LaTeX-mode plain-tex-mode)
  :init
  (progn
```

```
    ;;(add-hook 'LaTeX-mode-hook 'visual-line-mode)
    (add-hook 'LaTeX-mode-hook 'LaTeX-math-mode)
    (setq TeX-auto-save t
          TeX-parse-self t
          reftex-plug-into-AUCTeX t
          TeX-PDF-mode t))
  (add-hook 'LaTeX-mode-hook 'TeX-PDF-mode)
  (setq TeX-source-correlate-method 'synctex)
  (setq TeX-source-correlate-mode t)
  (eval-after-load "tex"
    '(add-to-list 'TeX-command-list '("xelatexmk" "latexmk -synctex=1 -shell-escape -xe
    )
  (add-hook 'TeX-mode-hook '(lambda () (setq TeX-command-default "xelatexmk"))))
```

3.4 PDF Viewer

```
(setq TeX-view-program-selection '((output-pdf "PDF Viewer")))
(setq TeX-view-program-list
      '(("PDF Viewer" "/Applications/Skim.app/Contents/SharedSupport/displayline -b %n
```

3.5 Bibtex

```
(setq bibtex-autokey-year-length 4
      bibtex-autokey-name-year-separator "-"
      bibtex-autokey-year-title-separator "-"
      bibtex-autokey-titleword-separator "-"
      bibtex-autokey-titlewords 2
      bibtex-autokey-titlewords-stretch 1
      bibtex-autokey-titleword-length 5)

(setq bibtex-completion-bibliography '("~/Google Drive/bibliography/references.bib"))
(setq reftex-default-bibliography
      '("~/Google Drive/bibliography/references.bib"))

(setq reftex-bibpath-environment-variables
      '("~/Google Drive/bibliography/"))

(setq reftex-default-bibliography '("~/Google Drive/bibliography/references.bib"))
(setq reftex-bibliography-commands '("bibliography" "nobibliography" "addbibresource"))
```

```
(setq reftex-default-bibliography
      (quote
       ("user.bib" "local.bib" "main.bib")))
```

3.5.1 Google Scholar

```
(setq gscholar-bibtex-default-source "Google Scholar")
(setq gscholar-bibtex-database-file "~/Google Drive/bibliography/references.bib")
```

3.6 Misc

```
(setq font-latex-match-reference-keywords
      '(("cite" "[{")
        ("cites" "[{}]")
        ("autocite" "[{")
        ("footcite" "[{")
        ("footcites" "[{")
        ("parencite" "[{")
        ("textcite" "[{")
        ("fullcite" "[{")
        ("citetitle" "[{")
        ("citetitles" "[{")
        ("headlessfullcite" "[{")))

(setq reftex-cite-prompt-optional-args t)
(setq reftex-cite-cleanup-optional-args t)
```

4 Org-Mode

4.1 Org-agenda

```
;; set key for agenda
(global-set-key (kbd "C-c a") 'org-agenda)

;;file to save todo items
(setq org-agenda-files '("~/todo.org"))

;;set priority range from A to C with default A
(setq org-highest-priority ?A)
(setq org-lowest-priority ?C)
```

```
(setq org-default-priority ?A)

;;set colours for priorities
(setq org-priority-faces '((?A . (:foreground "#FODFAF" :weight bold))
                           (?B . (:foreground "LightSteelBlue"))
                           (?C . (:foreground "OliveDrab"))))

;;capture todo items using C-c c t
(define-key global-map (kbd "C-c c") 'org-capture)
(setq org-capture-templates
      '(("t" "todo" entry (file+headline "~/todo.org" "Tasks")
         "* TODO [#A] %?\nDEADLINE: %(org-insert-time-stamp (org-read-date nil t \"+0d\

;;open agenda in current window
(setq org-agenda-window-setup (quote current-window))
;;warn me of any deadlines in next 7 days
(setq org-deadline-warning-days 7)
;;show me tasks scheduled or due in next fortnight
(setq org-agenda-span (quote fortnight))
;;don't show tasks as scheduled if they are already shown as a deadline
(setq org-agenda-skip-scheduled-if-deadline-is-shown t)
;;don't give awarning colour to tasks with impending deadlines
;;if they are scheduled to be done
(setq org-agenda-skip-deadline-prewarning-if-scheduled (quote pre-scheduled))
;;don't show tasks that are scheduled or have deadlines in the
;;normal todo list
(setq org-agenda-todo-ignore-deadlines (quote all))
(setq org-agenda-todo-ignore-scheduled (quote all))
;;sort tasks in order of when they are due and then by priority
(setq org-agenda-sorting-strategy
      (quote
       ((agenda deadline-up priority-down)
        (todo priority-down category-keep)
        (tags priority-down category-keep)
        (search category-keep))))
```

4.2 Org Aesthetics

```
(add-hook 'org-mode-hook
          (lambda ()
```

```
            (org-bullets-mode t)))
(setq org-bullets-bullet-list '("" "" "" "" ""))
(setq org-src-fontify-natively t)
(setq org-src-tab-acts-natively t)
(setq org-src-window-setup 'current-window)
;;(setq org-ellipsis "")
(setf org-todo-keyword-faces '(("PLANED" . (:foreground "white" :background "#FF8598"
                                ("TODO" . (:foreground "white" :background "#AEAEAE"  :
                                ("STARTED" . (:foreground "white" :background "#01B0F0"
                                ("DONE" . (:foreground "black" :background "#AEEE00" :bo
```

4.3 Org-latex

```
(require 'org-ref)
(require 'org-ref-pdf)
(require 'org-ref-url-utils)
(require 'helm-bibtex)
(require 'dash)
(require 'hydra)
(require 'key-chord)
(require 'parsebib)
(require 'async)

(add-to-list 'org-latex-default-packages-alist '("" "natbib" "") t)
(add-to-list 'org-latex-default-packages-alist
             '("linktocpage,pdfstartview=FitH,colorlinks,
linkcolor=blue,anchorcolor=blue,
citecolor=blue,filecolor=blue,menucolor=blue,urlcolor=blue"
               "hyperref" nil)
             t)
```

4.3.1 Orgmode-reftex

```
;; Make RefTeX work with Org-Mode
;; use 'C-c (' instead of 'C-c [' because the latter is already
;; defined in orgmode to the add-to-agenda command.
(defun org-mode-reftex-setup ()
  (load-library "reftex")
  (and (buffer-file-name)
       (file-exists-p (buffer-file-name))
```

```
      (reftex-parse-all))
  (define-key org-mode-map (kbd "C-c (") 'reftex-citation))

(add-hook 'org-mode-hook 'org-mode-reftex-setup)

(setq org-latex-pdf-process
    '("pdflatex -interaction nonstopmode -output-directory %o %f"
      "bibtex %b"
      "pdflatex -interaction nonstopmode -output-directory %o %f"
      "pdflatex -interaction nonstopmode -output-directory %o %f"))
```

4.4 Misc

4.4.1 Export Twitter Bootstrap

```
(setq org-publish-project-alist
      '(("org-notes"
         :base-directory "~/org/"
         :publishing-directory "~/public_html/"
         :publishing-function org-twbs-publish-to-html
         :with-sub-superscript nil
         )))
```

4.4.2 Org-publish-buffer

```
(defun my-org-publish-buffer ()
  (interactive)
  (save-buffer)
  (save-excursion (org-publish-current-file))
  (let* ((proj (org-publish-get-project-from-filename buffer-file-name))
         (proj-plist (cdr proj))
         (rel (file-relative-name buffer-file-name
                                  (plist-get proj-plist :base-directory)))
         (dest (plist-get proj-plist :publishing-directory)))
    (browse-url (concat "file://"
                        (file-name-as-directory (expand-file-name dest))
                        (file-name-sans-extension rel)
                        ".html"))))
```

4.4.3 Misc

```
(setq org-latex-default-packages-alist
       (-remove-item
        '("" "hyperref" nil)
        org-latex-default-packages-alist))
```

5 Misc

```
;; (use-package reftex
;;   :commands turn-on-reftex
;;   :init
;;   (setq reftex-cite-format
;;         '((?\C-m . "\\cite[]{%l}")
;;           (?t . "\\citet{%l}")
;;           (?p . "\\citep[]{%l}")
;;           (?a . "\\autocite{%l}")
;;           (?A . "\\textcite{%l}")
;;           (?P . "[@%l]")
;;           (?T . "@%l [p. ]")
;;           (?x . "[]{%l}")
;;           (?X . "{%l}")))
;;   (setq bibtex-autokey-titleword-length 0
;;         bibtex-autokey-titleword-separator ""
;;         bibtex-autokey-titlewords 0
;;         bibtex-autokey-year-length 4
;;         bibtex-autokey-year-title-separator "")
;;   (setq reftex-default-bibliography '("~/Google Drive/bibliography/references.bib"))
;;   (setq reftex-bibliography-commands '("bibliography" "nobibliography" "addbibresour
;;   (setq reftex-extra-bindings t)
;;   :config
;;   (add-hook 'LaTeX-mode-hook 'turn-on-reftex))

;;enable magic-latex-buffer
;; (require 'magic-latex-buffer)
;; (add-hook 'latex-mode-hook 'magic-latex-buffer)
;; (add-hook 'Latex-mode-hook 'magic-latex-buffer)

;;(setq spaceline-window-numbers-unicode t)
;;(setq spaceline-workspace-numbers-unicode t)
```

```
;; setting transparency
;;(global-set-key (kbd "C-M-)") 'transparency-increase)
;;(global-set-key (kbd "C-M-(") 'transparency-decrease)

;; RefTeX formats for biblatex (not natbib)
;; (setq reftex-cite-format
;;       '(
;;         (?\C-m . "\\cite[]{%l}")
;;         (?t . "\\textcite{%l}")
;;         (?a . "\\autocite[]{%l}")
;;         (?p . "\\parencite{%l}")
;;         (?f . "\\footcite[][]{%l}")
;;         (?F . "\\fullcite[]{%l}")
;;         (?x . "[]{%l}")
;;         (?X . "{%l}")
;;         ))

;;(setq bibtex-completion-pdf-open-function
;;      (lambda (fpath)
;;        (call-process "open" nil 0 nil "-a" "/Applications/Skim.app" fpath)))
```