My Emacs Config File

Jordan Schupbach

August 24, 2018

Contents

1 Pre package load user defined options

```
;; Set fold path
(add-to-list 'load-path "~/.emacs.d/load-folder/")

;; Toggle menubar
(global-set-key [f9] 'toggle-menu-bar-mode-from-frame)

;; Remove tool-bar by default
(tool-bar-mode -1)

;; Load Theme
```

```
(load-theme 'tango-dark)

;; Vim style folding
;; (autoload 'folding-mode          "folding" "Folding mode" t)
;; (autoload 'turn-off-folding-mode "folding" "Folding mode" t)
;; (autoload 'turn-on-folding-mode  "folding" "Folding mode" t)

;; Load path to origami
(add-to-list 'load-path (expand-file-name "~/.emacs.d/elpa/origami-20180101.753/origam

;; (add-to-list 'load-path (expand-file-name "~/.emacs.d/icicles-el-files/icicles.el"))
;; (require 'icicles)
```

2 Load and Configure Packages

2.1 airline-themes

```
(use-package airline-themes)
(load-theme 'airline-dark t)
```

2.2 auto-complete

```
(use-package auto-complete)
```

2.3 elfeed

```
(use-package elfeed)
(global-set-key (kbd "C-x w") 'elfeed)
(use-package elfeed-org)
(use-package elfeed-goodies)
(elfeed-org)
(setq rmh-elfeed-org-files (list "~/.emacs.d/elfeed.org"))
(elfeed-goodies/setup)
```

2.4 ess

```
(use-package ess)
```

2.5 evil

```
(setq evil-want-integration nil)
```

```
(use-package evil)
(evil-mode 1)
```

2.6 evil-collection

```
(use-package evil-collection)
(when (require ‘evil-collection nil t)
  (evil-collection-init))
```

2.7 flycheck

```
(use-package flycheck)
(add-hook ’after-init-hook #’global-flycheck-mode)
```

2.8 origami

```
(use-package origami)
;;origami https://github.com/gregsexton/origami.el
;; (global-origami-mode 1)

;; (defun nin-origami-toggle-node ()
;;   (interactive)
;;   (if (equal major-mode ’org-mode)
;;       (org-cycle)
;;     (save-excursion ;; leave point where it is
;;       (goto-char (point-at-eol))             ;; then go to the end of line
;;       (origami-toggle-node (current-buffer) (point)))))                 ;; and try t

;; (add-hook ’prog-mode-hook
;;           (lambda ()
;;             (setq-local origami-fold-style ’triple-braces)
;;             (origami-mode)
;;             (origami-close-all-nodes (current-buffer))))
;; (evil-define-key ’normal prog-mode-map (kbd "TAB") ’nin-origami-toggle-node)
;;
;; (define-key evil-normal-state-map "za" ’origami-forward-toggle-node)
;; (define-key evil-normal-state-map "zR" ’origami-close-all-nodes)
;; (define-key evil-normal-state-map "zM" ’origami-open-all-nodes)
;; (define-key evil-normal-state-map "zr" ’origami-close-node-recursively)
;; (define-key evil-normal-state-map "zm" ’origami-open-node-recursively)
;; (define-key evil-normal-state-map "zo" ’origami-show-node)
;; (define-key evil-normal-state-map "zc" 'origami-close-node)
;; (define-key evil-normal-state-map "zj" 'origami-forward-fold)
;; (define-key evil-normal-state-map "zk" 'origami-previous-fold)
;; (define-key evil-visual-state-map "zf"
;;   '(lambda ()
;;      "create fold and add comment to it"
;;      (interactive)
;;      (setq start (region-beginning))
;;      (setq end (region-end))
;;      (deactivate-mark)
;;      (and (< end start)
;;           (setq start (prog1 end (setq end start))))
;;      (goto-char start)
;;      (beginning-of-line)
;;      (indent-according-to-mode)
;;      (insert comment-start)
;;      (setq start (point))
;;      (insert "Folding" " {{{")
;;      (newline-and-indent)
;;      (goto-char end)
;;      (end-of-line)
;;      (and (not (bolp))
;;           (eq 0 (forward-line))
;;           (eobp)
;;           (insert ?\n))
;;      (indent-according-to-mode)
;;      (if (equal comment-end "")
;;          (insert comment-start " }}}")
;;        (insert comment-end "}}}"))
;;      (newline-and-indent)
;;      (goto-char start)
;;      ))
```

2.9 ranger

```
(use-package ranger)
(ranger-override-dired-mode t)
```

3 Post package load user defined options

```
;; {{{ Define evil-mode mappings for vim-style folding
;; (define-key evil-normal-state-map "zz" 'folding-toggle-show-hide)
;; (define-key evil-normal-state-map "zR" 'folding-whole-buffer)
;; (define-key evil-normal-state-map "zM" 'folding-open-buffer)
;; (define-key evil-normal-state-map "zr" 'folding-hide-current-subtree)
;;(defe-key evil-normal-state-map "zm" 'folding-show-current-subtree)
;;(defi-key evil-normal-state-map "zo" 'folding-show-current-entry)
;;(definkey evil-normal-state-map "zc" 'folding-hide-current-entry)
;;(defineey evil-normal-state-map "zj" 'folding-next-visible-heading)
;;(define-y evil-normal-state-map "zk" 'folding-previous-visible-heading)
;;;; (definkey evil-normal-state-map "zf" 'folding-fold-region)
;;(define-keevil-normal-state-map "zf"
;;  '(lambda
;;     "createold, exit from shifting and add comment to it"
;;     (interacve)
;;     (folding-ld-region (region-beginning) (region-end))
;;     (folding-sft-out)
;;     (folding-tole-show-hide)
;;     (evil-appendine 1)
;;     (insert " FolngComment")
;;     (evil-normal-ste)
;;     (evil-backward-RD-begin)
;;))
;;
;;(defun bss/foing-t--org ()
;;  "selective folding toge by tab: skip org-mode"
;;  (interactive)
;;  (if (ual major-mode 'o-mode)
;;      (org-cycle)
;;    (foing-toggle-show-hid
;;    ))
;;(define-key evil-norl-state-p (kbd "<tab>") 'bss/folding-not-in-org)
;;
;;(add-hook 'python-mode-hook ama () (folding-mode)))
;; }}} Define evil-mode mappings for vim-style folding
```