Contents

;; Phil's custom emacs config

1 Tasks

1.1 DONE Configure backups to save the files in a central directory

1.2 TODO Set a short cut to open the config.org file on windows

1.3 DONE Magit Git interface

1.4 DONE Which key

1.5 DONE figure out easy buffer switching

1.6 DONE Window Switching

1.7 TODO Org Mode config

1.8 TODO org-ref

1.9 TODO deft?

1.10 TODO Set shortcuts for code blocks

1.11 TODO Elfeed

1.12 TODO Web mode

1.13 TODO Spelling

1.14 TODO Sync bookmarks

1.15 TODO Todo states

1.16 TODO Configure backups correctly on windows

1.17 TODO Python

1.18 TODO Latex

1.19 TODO Snippets

2 User Interface

```
; Highlights the current cursor line
(global-hl-line-mode t)

(global-set-key (kbd "C-x C-b") 'ibuffer)
```

```
(defvar runemacs/default-font-size 140)

(setq inhibit-startup-message t)

(scroll-bar-mode -1)        ; Disable visible scrollbar
(tool-bar-mode -1)          ; Disable the toolbar
(tooltip-mode -1)           ; Disable tooltips
(set-fringe-mode 10)        ; Give some breathing room
(menu-bar-mode -1)            ; Disable the menu bar

;; Set up the visible bell
(setq visible-bell t)

(set-face-attribute 'default nil :font "Menlo" :height runemacs/default-font-size)

;; Make ESC quit prompts
(global-set-key (kbd "<escape>") 'keyboard-escape-quit)

;; Initialize package sources
(require 'package)

(setq package-archives '(("melpa" . "https://melpa.org/packages/")
 ("org" . "https://orgmode.org/elpa/")
 ("elpa" . "https://elpa.gnu.org/packages/")))

(package-initialize)
(unless package-archive-contents
  (package-refresh-contents))

;; Initialize use-package on non-Linux platforms
(unless (package-installed-p 'use-package)
  (package-install 'use-package))

(require 'use-package)
(setq use-package-always-ensure t)

(column-number-mode)
;(display-line-numbers-mode 'relative)
(setq display-line-numbers 'relative)
```

```
;; Disable line numbers for some modes
(dolist (mode '(term-mode-hook
shell-mode-hook
treemacs-mode-hook
eshell-mode-hook))
  (add-hook mode (lambda () (display-line-numbers-mode 0))))


(use-package command-log-mode)

(use-package ivy
  :diminish
  :bind (("C-s" . swiper)
 :map ivy-minibuffer-map
 ("TAB" . ivy-alt-done)
 ("C-l" . ivy-alt-done)
 ("C-j" . ivy-next-line)
 ("C-k" . ivy-previous-line)
 :map ivy-switch-buffer-map
 ("C-k" . ivy-previous-line)
 ("C-l" . ivy-done)
 ("C-d" . ivy-switch-buffer-kill)
 :map ivy-reverse-i-search-map
 ("C-k" . ivy-previous-line)
 ("C-d" . ivy-reverse-i-search-kill))
  :config
  (ivy-mode 1))

;; NOTE: The first time you load your configuration on a new machine, you'll
;; need to run the following command interactively so that mode line icons
;; display correctly:
;;
;; M-x all-the-icons-install-fonts

(use-package all-the-icons)

(use-package doom-modeline
  :init (doom-modeline-mode 1)
  :custom ((doom-modeline-height 15)))
```

```
(use-package doom-themes
  :init (load-theme 'doom-palenight t))

(use-package rainbow-delimiters
  :hook (prog-mode . rainbow-delimiters-mode))

(use-package which-key
  :init (which-key-mode)
  :diminish which-key-mode
  :config
  (setq which-key-idle-delay 1))

(use-package ivy-rich
  :init
  (ivy-rich-mode 1))

(use-package counsel
  :bind (("M-x" . counsel-M-x)
 ("C-x b" . counsel-ibuffer)
 ("C-x C-f" . counsel-find-file)
 :map minibuffer-local-map
 ("C-r" . 'counsel-minibuffer-history)))

(use-package helpful
  :custom
  (counsel-describe-function-function #'helpful-callable)
  (counsel-describe-variable-function #'helpful-variable)
  :bind
  ([remap describe-function] . counsel-describe-function)
  ([remap describe-command] . helpful-command)
  ([remap describe-variable] . counsel-describe-variable)
  ([remap describe-key] . helpful-key))

(use-package general
  :config
  (general-create-definer rune/leader-keys
    :keymaps '(normal insert visual emacs)
    :prefix "SPC"
    :global-prefix "C-SPC")
```

```
  (rune/leader-keys
    "t"  '(:ignore t :which-key "toggles")
    "tt" '(counsel-load-theme :which-key "choose theme")))

(use-package evil
  :init
  (setq evil-want-integration t)
  (setq evil-want-keybinding nil)
  (setq evil-want-C-u-scroll t)
  (setq evil-want-C-i-jump nil)
  :config
  (evil-mode 1)
  (define-key evil-insert-state-map (kbd "C-g") 'evil-normal-state)
  (define-key evil-insert-state-map (kbd "C-h") 'evil-delete-backward-char-and-join)

  ;; Use visual line motions even outside of visual-line-mode buffers
  (evil-global-set-key 'motion "j" 'evil-next-visual-line)
  (evil-global-set-key 'motion "k" 'evil-previous-visual-line)

  (evil-set-initial-state 'messages-buffer-mode 'normal)
  (evil-set-initial-state 'dashboard-mode 'normal))

(use-package evil-collection
  :after evil
  :config
  (evil-collection-init))

(use-package hydra)

(defhydra hydra-text-scale (:timeout 4)
  "scale text"
  ("j" text-scale-increase "in")
  ("k" text-scale-decrease "out")
  ("f" nil "finished" :exit t))

(rune/leader-keys
  "ts" '(hydra-text-scale/body :which-key "scale text"))

(setq indo-enable-flex-matching t)
```

```
(setq ido-everywhere t)
(ido-mode 1)
```

3 Projectile

```
(use-package projectile
  :diminish projectile-mode
  :config (projectile-mode)
  :custom ((projectile-completion-system 'ivy))
  :bind-keymap
  ("C-c p" . projectile-command-map)
  :init
  ;; NOTE: Set this to the folder where you keep your Git repos!
  (when (file-directory-p "~/Dropbox/@Work")
    (setq projectile-project-search-path '("~/Dropbox/@Work")))
  (setq projectile-switch-project-action #'projectile-dired))

(use-package counsel-projectile
  :config (counsel-projectile-mode))
```

4 Magit

```
(use-package magit
  :custom
  (magit-display-buffer-function #'magit-display-buffer-same-window-except-diff-v1))

;; NOTE: Make sure to configure a GitHub token before using this package!
(use-package forge)
```

5 org mode

```
(require 'org)
;; set up org mobile mode for ipad
(setq org-directory "~/Dropbox/org")

(setq org-agenda-files (list "~/Dropbox/org/work.org"
```

```
 "~/Dropbox/org/personal.org"))

(setq org-mobile-inbox-for-pull "~/Dropbox/org/flagged.org")
(setq org-mobile-directory "~/Dropbox/Apps/MobileOrg")

(require 'org-tempo)

(global-visual-line-mode t)
(defun efs/org-mode-setup ()
  (org-indent-mode)
  (variable-pitch-mode 1)
  (visual-line-mode 1))

(use-package org-bullets
  :after org
  :hook (org-mode . org-bullets-mode)
  :custom
  (org-bullets-bullet-list '("" "" "" "" "" "" "")))

(defun efs/org-mode-visual-fill ()
  (setq visual-fill-column-width 100
visual-fill-column-center-text t)
  (visual-fill-column-mode 1))

(use-package visual-fill-column
  :hook (org-mode . efs/org-mode-visual-fill))
(defun efs/org-mode-setup ()
  (org-indent-mode)
  (variable-pitch-mode 1)
  (visual-line-mode 1))
```

6 Switch Window

```
    ;; easy window switchingo
(use-package switch-window
  :ensure t
  :bind
  ;; default C-x o is other-window
```

```
  ;; default C-x C-o is delete-blank-lines
  (("C-x o" . switch-window)
   ("C-x C-o" . switch-window))
  :config
  (setq switch-window-multiple-frames t)
  (setq switch-window-shortcut-style 'qwerty)
  ;; when Emacs is run as client, the first shortcut does not appear
  ;; "x" acts as a dummy; remove first entry if not running server
  (setq switch-window-qwerty-shortcuts '("x" "a" "s" "d" "f" "j" "k" "l" ";" "w" "e" "
  (setq switch-window-increase 3))
```

7 code blocks

```
    (setq org-src-fontify-natively t)

(org-babel-do-load-languages
 'org-babel-load-languages
 '((emacs-lisp . t)
   (python . t)))
```

8 Pdf tools

```
;;; Install epdfinfo via 'brew install pdf-tools --HEAD' and then install the
;;; pdf-tools elisp via the use-package below. To upgrade the epdfinfo
;;; server, just do 'brew upgrade pdf-tools' prior to upgrading to newest
;;; pdf-tools package using Emacs package system. If things get messed
;;; up, just do 'brew uninstall pdf-tools', wipe out the elpa
;;; pdf-tools package and reinstall both as at the start.

(use-package pdf-tools
    :ensure t
    :config
    (custom-set-variables
      '(pdf-tools-handle-upgrades nil)) ; Use brew upgrade pdf-tools instead.
   (setq pdf-info-epdfinfo-program "/usr/local/bin/epdfinfo"))
(pdf-tools-install)

(use-package org-pdftools
```

```
  :hook (org-mode . org-pdftools-setup-link))

(add-to-list 'org-file-apps '("\\.pdf\\'" . emacs))
```