

Contents

1	Tasks	2
1.1	DONE Configure backups to save the files in a central directory	2
1.2	TODO Set a short cut to open the config.org file on windows	2
1.3	TODO Get Beamer working	2
1.4	TODO Checkout what emacs for writers guy was talking about	2
1.5	DONE Magit Git interface	3
1.6	DONE Which key	3
1.7	DONE figure out easy buffer switching	3
1.8	DONE Window Switching	3
1.9	TODO Org Mode config	3
1.10	TODO org-ref	3
1.11	TODO deft?	3
1.12	DONE Set shortcuts for code blocks	3
1.13	TODO Elfeed	3
1.14	TODO Web mode	3
1.15	TODO Spelling	3
1.16	TODO Sync bookmarks	3
1.17	TODO Todo states	3
1.18	TODO Configure backups correctly on windows	3
1.19	TODO Python	3
1.20	TODO Latex	3
1.21	TODO Snippets	3
2	User Interface	3
3	Projectile	8
4	Magit	8
5	org mode	8
6	Switch Window	9
7	Code Blocks	10
8	Pdf tools	10
9	Elfeed	11

10 Beamer

11

;; 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 **TODO** Get Beamer working

1.4 **TODO** Checkout what emacs for writers guy was talking about

- short cut for last open file
- page up and down for moving through buffers
- listing recently open files (this was done through helm)
- See if you can find his config on github

- 1.5 DONE Magit Git interface
- 1.6 DONE Which key
- 1.7 DONE figure out easy buffer switching
- 1.8 DONE Window Switching
- 1.9 TODO Org Mode config
- 1.10 TODO org-ref
- 1.11 TODO deft?
- 1.12 DONE Set shortcuts for code blocks
- 1.13 TODO Elfeed
- 1.14 TODO Web mode
- 1.15 TODO Spelling
- 1.16 TODO Sync bookmarks
- 1.17 TODO Todo states
- 1.18 TODO Configure backups correctly on windows
- 1.19 TODO Python
- 1.20 TODO Latex
- 1.21 TODO Snippets

2 User Interface

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

; this will not require the yes to the reload.
(setq revert-without-query '(".pdf"))

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

; open buffers automatically vertically and not horizontal
(setq split-height-threshold nil)
(setq split-width-threshold 0)
```

```

; retina support for pdftools
(setq pdf-view-use-scaling t)

(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")


(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 switching
(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" "n"))
(setq switch-window-increase 3))

```

7 Code Blocks

```

;; This is needed as of Org 9.2
(require 'org-tempo)

(add-to-list 'org-structure-template-alist '("sh" . "src sh"))
(add-to-list 'org-structure-template-alist '("el" . "src emacs-lisp"))
(add-to-list 'org-structure-template-alist '("py" . "src python"))
(add-to-list 'org-structure-template-alist '("yaml" . "src yaml"))
(add-to-list 'org-structure-template-alist '("json" . "src json"))
(add-to-list 'org-structure-template-alist '("la" . "src latex"))

(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))

; open pdf in pdftools
(add-to-list 'org-file-apps '("\\.pdf\\'" . emacs))

```

9 Elfeed

```

(use-package elfeed
  :commands elfeed
  :config
  (setq elfeed-feeds
    '("https://nullprogram.com/feed/"
      "https://ambrevar.xyz/atom.xml"
      "https://valdyas.org/fading/feed/"
      "https://www.reddit.com/r/emacs/.rss")))

(global-set-key (kbd "C-x w") 'elfeed)

```

10 Beamer

```

(require 'ox-latex)
(add-to-list 'org-latex-classes
  '("beamer"

```

```
"\\documentclass\[presentation\\]{beamer}"  
("\\section{%s}" . "\\section*{%s}")  
("\\subsection{%s}" . "\\subsection*{%s}")  
("\\subsubsection{%s}" . "\\subsubsection*{%s}"))
```