Literate Doom Emacs Config

Jonathan Fung

Information

User Info

Some functionality uses this to identify you, e.g. GPG configuration, email clients, file templates and snippets.

```
(setq user-full-name  "Jonathan Fung"
      user-mail-address "jonathanfung2000@gmail.com")
```

Doom Info

Here are some additional functions/macros that could help you configure Doom:

- 'load!' for loading external *.el files relative to this one
- 'use-package' for configuring packages
- 'after!' for running code after a package has loaded
- 'add-load-path!' for adding directories to the 'load-path', relative to this file. Emacs searches the 'load-path' when you load packages with 'require' or 'use-package'.
- 'map!' for binding new keys

To get info about any of these functions/macros, move the cursor over the highlighted symbol at press 'K' (non-evil users must press 'C-c g k'). This will open documentation for it, including demos of how they are used.

You can also try 'gd' (or 'C-c g d') to jump to their definition and see how they are implemented.

Visual Interface

Fonts

Doom exposes five (optional) variables for controlling fonts in Doom. Here are the three important ones:

- 'doom-font'
- 'doom-variable-pitch-font'
- 'doom-big-font' – used for 'doom-big-font-mode'; use this for presentations or streaming.

They all accept either a font-spec, font string ("Input Mono-12"), or xlfd font string. You generally only need these two:

```
;; (setq doom-font (font-spec :family "Source Code Pro" :size 24))
;; (setq doom-big-font (font-spec :family "Source Code Pro" :size 36))
(setq doom-font (font-spec :family "JetBrains Mono" :weight 'light ':size 24))
(setq doom-big-font (font-spec :family "JetBrains Mono" :weight 'light :size 36))
(setq doom-variable-pitch-font (font-spec :family "Roboto" :size 24 :weight 'bold))
```

Theme

There are two ways to load a theme. Both assume the theme is installed and available. You can either set 'doom-theme' or manually load a theme with the 'load-theme' function.

Reference: `https://protesilaos.com/modus-themes`

```
(require 'modus-themes)                 ; common code
(require 'modus-operandi-theme)         ; light theme
(require 'modus-vivendi-theme)          ; dark theme

(load-theme 'modus-vivendi t)           ; Dark theme
(load-theme 'modus-operandi t)          ; Light theme

(global-set-key (kbd "<f5>") (lambda () (interactive) (modus-themes-toggle) (set-face-background 'mode-l

;; Set customization options to values of your choice
(setq modus-themes-slanted-constructs t
      modus-themes-bold-constructs t
      modus-themes-fringes 'intense ; {nil,'subtle,'intense}
      modus-themes-mode-line nil ; {nil,'3d,'moody}
      modus-themes-syntax nil ; Lots of options---continue reading the manual
      modus-themes-intense-hl-line t
      modus-themes-paren-match 'intense ; {nil,'subtle-bold,'intense,'intense-bold}
      modus-themes-links nil ; Lots of options---continue reading the manual
      modus-themes-no-mixed-fonts nil
      modus-themes-prompts 'subtle ; {nil,'subtle,'intense}
      modus-themes-completions 'opinionated ; {nil,'moderate,'opinionated}
      modus-themes-region 'bg-only-no-extend ; {nil,'no-extend,'bg-only,'bg-only-no-extend}
      modus-themes-diffs nil ; {nil,'desaturated,'fg-only,'bg-only}
      modus-themes-org-blocks 'grayscale ; {nil,'grayscale,'rainbow}
      modus-themes-headings ; Lots of options---continue reading the manual
      '((1 . rainbow-section)
        ;; (2 . rainbow-line-no-bold)
        ;; (3 . no-bold)
        (t . rainbow-line))
      modus-themes-variable-pitch-headings nil
      modus-themes-scale-headings nil
      modus-themes-scale-1 1.1
      modus-themes-scale-2 1.15
      modus-themes-scale-3 1.21
      modus-themes-scale-4 1.27
      modus-themes-scale-5 1.33)
```

```
;; Load the light theme (‘modus-operandi’)
; doesn’t seem to work, function is not defined
;(modus-themes-load-operandi)
```

TODO Display

```
;includes part of the file’s directory name at the beginning of the shared buffer name to make unique
(setq uniquify-buffer-name-style ’forward)
;; this may do the same thing as uniquify-buffer...
(setq ivy-rich-path-style ’abbrev)

; just editted these line 12/24
;; idk what these 2 lines do
;; (add-to-list ’default-frame-alist ’(font . "Source Code Pro-10"))
;; (set-face-attribute ’default t :font "Source Code Pro-10")
```

Modeline

```
; CAUTION
; This might be fatal, might turn off all keymaps
; (setq display-battery-mode t)

(setq display-time-mode t)
(setq display-time-default-load-average nil)
(setq line-number-mode nil)
(setq column-number-mode nil)
(set-face-background ’mode-line "default")

(setq doom-modeline-buffer-encoding nil)
(setq doom-modeline-buffer-file-name-style ’relative-from-project)
```

hl-line mode

```
(setq hl-line-mode nil)
(map! :n "SPC t h" #’hl-line-mode)

; meant to only have hl-line highlight on end of line
(defun my-hl-line-range-function () (cons (line-end-position) (line-beginning-position 2)))
;(setq hl-line-range-function #’my-hl-line-range-function)

; standard full-width
(defun my-hl-line-range ()
  "Used as value of ‘hl-line-range-function’."
  (cons (line-beginning-position) (line-end-position)))

(setq-default hl-line-range-function #’my-hl-line-range)
```

Usability

Navigation

```
; Bind Zooms??
```

```
(map! :n "C-_" #'er/contract-region
      :n "C-+" #'er/expand-region)

;; ; unbind J,K,M
(map! :map evil-normal-state-map "J" nil
      "K" nil)
(map! :map evil-motion-state-map "M" nil
      "K" nil)

;; ; rebind J,K for scrolling
(map! :n "J" #'evil-scroll-line-up)
(map! :n "K" #'evil-scroll-line-down)

;; ; bind M for contexual lookup
(map! :n "M" #'+lookup/documentation)

;; ;; Make evil-mode up/down operate in screen lines instead of actual lines
(define-key evil-motion-state-map "j" 'evil-next-visual-line)
(define-key evil-motion-state-map "k" 'evil-previous-visual-line)
;; ;; Also in visual mode
(define-key evil-visual-state-map "j" 'evil-next-visual-line)
(define-key evil-visual-state-map "k" 'evil-previous-visual-line)
```

Screenshots

```
(defun screenshot-svg ()
  "Save a screenshot of the current frame as an SVG image.
Saves to a temp file and puts the filename in the kill ring."
  (interactive)
  (let* ((filename (make-temp-file "Emacs" nil ".svg"))
         (data (x-export-frames nil 'svg)))
    (with-temp-file filename
      (insert data))
    (kill-new filename)
    (message filename)))
```

Custom Keybinds

```
;; Bind toggles
(global-set-key (kbd "<f2>") 'mixed-pitch-mode)
(global-set-key (kbd "<f3>") 'olivetti-mode)
(global-set-key (kbd "<f4>") 'toggle-rot13-mode)
(setq olivetti-body-width 90)
; ;; (global-set-key (kbd "U") 'undo-tree-redo)

; Unbind language input switcher
(map! :map global-map "C-\\" nil)
; Bind toggle for 80-char limit, buffer-wide
(map! :n "SPC t c" 'display-fill-column-indicator-mode)
(map! :n "C-\\" 'display-fill-column-indicator-mode)

;; ; currently do not use org-roam, need to delete
;; (setq org-roam-directory "~/emacs/org-roam")
```

```
;; (setq org-roam-index-file "index.org")
;(define-key org-roam-mode-map (kbd "C-c n l") #'org-roam)
;(define-key org-roam-mode-map (kbd "C-c n f") #'org-roam-find-file)
;(define-key org-roam-mode-map (kbd "C-c n j") #'org-roam-jump-to-index)
;(define-key org-roam-mode-map (kbd "C-c n b") #'org-roam-switch-to-buffer)
;(define-key org-roam-mode-map (kbd "C-c n g") #'org-roam-graph)
;(define-key org-mode-map (kbd "C-c n i") #'org-roam-insert)
;(require 'org-roam-protocol)
```

Org-Mode

Org

```
(setq org-directory "~/org/")
(setq display-line-numbers-type 'relative)

(setq org-ellipsis " ")
(setq org-startup-folded 'content)

(add-hook 'org-mode-hook (lambda () (org-superstar-mode 1)))
(setq org-superstar-headline-bullets-list
      '("" ("" ?) "" ""))

;; (add-hook 'org-mode-hook 'pandoc-mode)
;; (add-hook 'after-save-hook #'pandoc-convert-to-pdf)
```

Org Agenda + Super Agenda

```
(setq org-agenda-files '("~/org/Agenda.org"))
(setq org-tag-faces
      '(("Synth" . "gold2") ("Nano" . "lime green") ("Light" . "red2")
        ("Snr" . "medium orchid") ("Fail" . "dodger blue")))

(setq org-agenda-start-day "+0")

(setq org-super-agenda-date-format "%A, %e %b")
(setq org-super-agenda-header-separator ?-)
(org-super-agenda-mode)

(setq org-agenda-custom-commands
      '(("z" "Super View"
         (
          ;; (agenda "" ((org-super-agenda-groups
          ;;              '((:name "Today"
          ;;                       :time-grid t
          ;;                       :date today
          ;;                       :todo "TODAY"
          ;;                       :scheduled today
          ;;                       :order 1)))))
          (alltodo "" ((org-agenda-overriding-header (concat (make-string 20 ?\n) "Today is "(org-read-d
                       (org-super-agenda-groups
                        '(
```

```
                            (:name "Overdue"
                                   :deadline past
                                   :order 1)
                            (:name "Scheduled"
                                   :auto-planning t
                                   :order 0)
                            (:name "========\n Personal"
                                   :tag "Person"
                                   :order 10)
                            (:name "Email"
                                   :tag "Email"
                                   :order 15)
                            (:discard (:anything t)))))))))))

(defun org-super-view-agenda ()
  (interactive)
  (org-agenda nil "z"))
(map! :n "SPC o v" 'org-super-view-agenda)

; removes 'agenda' prefix coming from agenda.org
; also adds in effort level
; should be (todo   . " %i %-12:c") if using multiple files
(setq org-agenda-prefix-format
      '((agenda . " %i %-12:c%?-12t% s")
        (todo   . " [%e] ")
        (tags   . " %i %-12:c")
        (search . " %i %-12:c")))
```

Org Capture

```
(setq org-capture-templates
      '(("t" "Agenda TODO" entry (file "~/org/Agenda.org")
        "* TODO %?" :prepend t)
        ("e" "email" entry (file+headline "~/org/Agenda.org" "Emails")
         "* TODO Reply: %? \n - %a" :prepend t)
        ("d" "designboard" entry (file "~/org/designboard.org")
         "* %? \n- %t" :prepend t)
      ))

(map! :n "SPC z" 'org-capture)
```

Org Export - Latex

```
(setq org-latex-classes '(("article" "\\documentclass[11pt]{article}"
  ("\\section{%s}" . "\\section*{%s}")
  ("\\subsection{%s}" . "\\subsection*{%s}")
  ("\\subsubsection{%s}" . "\\subsubsection*{%s}")
  ("\\paragraph{%s}" . "\\paragraph*{%s}")
  ("\\subparagraph{%s}" . "\\subparagraph*{%s}"))
 ("report" "\\documentclass[11pt]{report}"
  ("\\part{%s}" . "\\part*{%s}")
  ("\\chapter{%s}" . "\\chapter*{%s}")
  ("\\section{%s}" . "\\section*{%s}")
```

```
  ("\\subsection{%s}" . "\\subsection*{%s}")
  ("\\subsubsection{%s}" . "\\subsubsection*{%s}"))
 ("book" "\\documentclass[11pt]{book}"
  ("\\part{%s}" . "\\part*{%s}")
  ("\\chapter{%s}" . "\\chapter*{%s}")
  ("\\section{%s}" . "\\section*{%s}")
  ("\\subsection{%s}" . "\\subsection*{%s}")
  ("\\subsubsection{%s}" . "\\subsubsection*{%s}"))
("notes"
 "\\documentclass[8pt]{article}
  \\usepackage[letterpaper, portrait, margin=1in]{geometry}
  \\usepackage[utf8]{inputenc}
  \\usepackage[T1]{fontenc}
  \\usepackage{amsmath}
  \\usepackage{amssymb}
  \\usepackage{hyperref}
  \\usepackage{enumitem}
  \\setitemize{itemsep=0.5pt}
  \\usepackage{lastpage}
  \\usepackage{fancyhdr}
  \\pagestyle{fancy}
  \\fancyhf{}
  \\usepackage{titling} % allows \thetitle \theauthor \thedate
  \\rhead{\\theauthor}
  \\lhead{\\thetitle}
  \\rfoot{\\thepage{} of \\pageref{LastPage}}
  \\linespread{1}
  \\setlength{\\parindent}{0pt}
  \\setlength{\\parskip}{0.5em plus 0.1em minus 0.2em}
  \\hypersetup{pdfborder=0 0 0}
  \\setcounter{secnumdepth}{0}"
  ("\\section{%s}" . "\\section*{%s}")
  ("\\subsection{%s}" . "\\subsection*{%s}")
  ("\\subsubsection{%s}" . "\\subsubsection*{%s}")
  ("\\paragraph{%s}" . "\\paragraph*{%s}"))))

(map! :n "SPC r r" #'org-latex-export-to-pdf)
```

Applications

Swiper - In-Buffer Fuzzy Finder

From r/emacs: By default if you have visual line mode on swiper scans every visual line, which can be really slow in large files. This forces swiper to revert back to searching only every actual line even if the user is using visual line mode

Note: seems to only find one occurrence in each file line, user needs to scan main buffer for thorough results.

```
(setq swiper-use-visual-line nil)
(setq swiper-use-visual-line-p (lambda (a) nil))
```

Dired - File Manager

```
(add-hook 'dired-mode-hook
```

```
      (lambda ()
        (dired-hide-details-mode)
        ))
; add this into above hook to default to sorting by edit time
; (dired-sort-toggle-or-edit)
```

Treemacs - Sidebar Directory Viewer

Bind external (zathura, etc.) opening for treemacs

```
(map! :n "SPC o o" #'treemacs-visit-node-in-external-application)
(map! :n "SPC o t" #'treemacs)
(setq treemacs-position 'right
      treemacs-width 25
      treemacs-indentation 1)
```

Notmuch - Email Client

```
;define function that syncs mbsync and refreshes notmuch
(defun sync-email ()
  "Lists the contents of the current directory."
  (interactive)
  (shell-command "mbsync -a && notmuch new"))

; bind notmuch-hello view
(map! :n "SPC o n" #'notmuch-hello)
; bind custom function to sync mbsync and notmuch
(map! :n "SPC r s" 'sync-email)

;; attempt to fix notmuch formatting
(setq notmuch-search-result-format
  '(("date" . "%12s ")
    ("count" . "%-6s ")
    ("authors" . "%-15s ")
    ("subject" . "%-10s ")
    ("tags" . "(%s)"))
)
(defun establish-notmuch ()
  (interactive)
(setq notmuch-saved-searches '((:name "Personal" :query "tag:inbox AND to:jonathanfung2000@gmail.com ANI
                               (:name "UCI" :query "tag:inbox AND to:fungjm@uci.edu AND date:nov_3_2020
                               (:name "Clean Inbox" :query "tag:inbox AND date:nov_3_2020..today")
                                   (:name "Flagged" :query "tag:inbox AND tag:flagged")
                               (:name "Inbox" :query "tag:inbox"))))

(map! :n "SPC r e" 'establish-notmuch)
```

Programming

LSP

```
; Rust
```

```
(setq lsp-rust-server "rust-analyzer")
(map! :n "SPC t u" #'lsp-ui-doc-mode)
```

Emacsclient

Workspace + emacsclient

Stops new emacsclient frames from creating new workspaces

```
(after! persp-mode
(setq persp-emacsclient-init-frame-behaviour-override "main"))
```

STRT Packages to Look At

STRT Transclusion

```
https://github.com/nobiot/org-transclusion
```

STRT Annotate

```
;(annotate-mode)
```

STRT Elgantt

```
;; enable elgantt - https://github.com/legalnonsense/elgantt/
;; (add-to-list 'load-path (concat user-emacs-directory "elgantt/")) ;; Or wherever it is located
;; (require 'elgantt)
```

HOLD Packages Not Used Right Now

HOLD header-line

```
;; (defun toggle-header-line-format ()
;;     "Toggle buffer-local var header-line-format as pseudo-top margin"
;;     (setq header-line-format (if (eq header-line-format nil) t nil))
;;     (interactive)
;;     (redraw-display))
;; (global-set-key (kbd "<f6>") 'toggle-header-line-format)
; use with set-face-font header-line
;(set-face-background 'header-line "white")
```

HOLD Pandoc

Bind pdf-export in pandoc

Note: Deprecated in favor of

```
;(map! :n "SPC r r" #'pandoc-convert-to-pdf)
```

HOLD Projectile

```
; unbind SPC p F
;(map! :map doom-leader-map "p F" nil)
; rebind SPC p F to search all projects' files
;(map! :n "SPC p F" #'projectile-find-file-in-known-projects)
```

HOLD mu4e

```
;; (add-to-list 'load-path "/usr/share/emacs/site-lisp/mu4e")
;; ;; Each path is relative to `+mu4e-mu4e-mail-path', which is ~/.mail by default
;; (set-email-account! "Personal"
;;   '((mu4e-sent-folder       . "/gmail/[Gmail].Sent Mail")
;;     ;(mu4e-drafts-folder     . "/gmail/Drafts")
;;     (mu4e-trash-folder      . "/gmail/[Gmail].Trash")
;;     (mu4e-refile-folder     . "/gmail/[Gmail].All Mail")
;;     (smtpmail-smtp-user     . "jonathanfung2000@gmail.com")
;;     ;; (mu4e-compose-signature . "---\nHenrik Lissner"))
;;   t))
;; (set-email-account! "UCI"
;;   '((mu4e-sent-folder       . "/uci/[Gmail].Sent Mail")
;;     ;(mu4e-drafts-folder     . "/gmail/Drafts")
;;     (mu4e-trash-folder      . "/uci/[Gmail].Trash")
;;     (mu4e-refile-folder     . "/uci/[Gmail].All Mail")
;;     (smtpmail-smtp-user     . "fungjm@uci.edu")
;;     ;; (mu4e-compose-signature . "---\nHenrik Lissner"))
;;   t))
```