

My literate GNU Emacs config

Marius Peter

23 Jul, 2020 (Thu)

Contents

1	TODO First-time setup	4
1.1	File system paths	4
2	Early setup	5
2.1	Garbage collection	5
2.2	Emacs client	5
2.3	Custom file	5
2.4	Profiling — start	5
2.5	Customization shortcuts	5
2.6	Backups	6
2.7	Secrets	6
3	Global key bindings	6
3.1	Navigation	6
3.1.1	find-file	6
3.1.2	Exit behaviours	6
3.2	Mouse zoom	7
4	Packages	7
4.1	Package archives	7
4.2	use-package	7
4.3	TODO Convenient package update	7
4.4	ivy	8
4.4.1	counsel	8
4.4.2	swiper	8
4.5	TODO evil-mode	8
4.6	org-mode	8
4.6.1	Basic customization	9
4.6.2	Invisible edits	9
4.6.3	Agenda	9

4.6.4	Timestamps	9
4.6.5	TEX export	10
4.6.6	Publish	10
4.6.7	Export	11
4.7	undo-tree	11
4.8	dumb-jump	11
4.9	gnuplot	11
4.10	ledger	11
4.11	TODO Sidebar	12
4.12	which-key	12
4.13	company	12
4.14	powerline	12
4.15	yasnippet	12
4.16	flycheck	12
4.17	TODO flyspell	12
4.18	csv-mode	12
4.19	json-mode	13
4.20	magit	13
4.21	pdf-tools	13
4.22	dashboard	13
4.23	rainbow-mode	13
4.24	projectile	13
4.25	all-the-icons	14
4.26	Better parentheses	14
5	Cosmetics	14
5.1	Cursors	14
5.1.1	Default cursor	14
5.2	Faces	14
5.2.1	variable-pitch-mode	15
5.2.2	TODO Default font size	15
5.3	Initial and default frames	15
5.3.1	GNU/Linux	15
5.4	Theme	15
5.4.1	blendoit-light	15
5.4.2	blendoit-dark	16
6	Editing preferences	16
6.1	Clean up menus	16
6.2	Coding standards	16
6.3	Dividers	17
6.4	Tabs	17
6.5	auto-fill	17
6.6	Recent files	17

6.7	pop-up-frames	17
7	Late setup	17
7.1	Profiling — stop	17
7.2	Profiling — report	17
8	Conclusion	17

Abstract

GNU Emacs is most often used as a text editor. The utmost level of customisation is afforded by enabling the user to rewrite *any* part of the source code and observe the editor's modified behaviour in real time. Since its inception in 1984, GNU Emacs has grown to be much more than a full-featured, high-productivity text editor—new *modes* have been written to interact with hundreds of file formats, including .txt, .pdf, .jpg, .csv, and .zip just to name a few. This configuration file itself was written in *Org mode*, a collection of functions enabling the harmonious mixing of code and comments in view of publication: this is the endgame of *literate programming*.

1 TODO First-time setup

Spacemacs-like dialog for default settings.

```
;; Prompt enterprise or personal install. Create file in .emacs.d/ on Linux,
;; AppData/ on Windows. Ask user for details and preferred bindings.

; Check if .emacs.d exists

; If it does, warn user

; Copy init-bootstrap.el from USB to where operating systems expects init.el

;; (defun blendoit/first-time-setup-windows-nt ()
;;   "Execute the first-time setup on MS Windows.
;;   If no '.emacs.d/' config exists on local system, copy
;;   init-bootstrap.el to '~.emacs.d/'."
;;   (interactive)
;;   (find-file "~/emacs.d/blendoit/blendoit-init.org"))

;; (cond ((string-equal system-type "windows-nt") blendoit/first-time-setup-windows-nt)
;;       ((string-equal system-type "gnu/linux") blendoit/first-time-setup-linux))
```

1.1 File system paths

In this subsection, we tell Emacs about relevant paths to resources.

On my MS Windows machine, I add the path to Portable Git.¹

```
(if (string-equal system-type "windows-nt")
    (add-to-list 'exec-path "C:/Users/marius.peter/PortableGit/bin/"))
```

¹Download from <https://git-scm.com/download/win>

2 Early setup

2.1 Garbage collection

First, we increase the RAM threshold beyond which the garbage collector is activated.

```
(setq gc-cons-threshold 100000000)
```

2.2 Emacs client

Makes opening emacs faster for following instances.

```
; (setq initial-buffer-choice (lambda () (get-buffer "*dashboard*")))
```

2.3 Custom file

Load settings created automatically by GNU Emacs Custom. (For example, any clickable option/toggle is saved here.) Useful for fooling around with M-x customize-group <package>.

user-emacs-directory

```
(setq custom-file (concat user-emacs-directory "init-custom.el"))  
(load custom-file)
```

2.4 Profiling — start

We start the profiler now , and will interrupt it in section 7.1. We will then present profiling report in section 7.2.

```
; (profiler-start)
```

2.5 Customization shortcuts

We begin by defining a user shortcut to this very file:

```
(defun my/find-literate-config ()  
  "Jump to this very file."  
  (interactive)  
  (find-file my/literate-config))  
  
(global-set-key (kbd "C-c c") 'my/find-literate-config)
```

Now, different shortcuts for other customization actions:

```
(global-set-key (kbd "C-c v") 'customize-variable)  
(global-set-key (kbd "C-c f") 'customize-face)
```

2.6 Backups

Backups are so important that they should be described right after the shortcut to this file.

```
(setq backup-directory-alist '(("." . ,temporary-file-directory))
auto-save-file-name-transforms '(("." . ,temporary-file-directory t))
  backup-by-copying t      ; Don't delink hardlinks
  version-control t        ; Use version numbers on backups
  delete-old-versions t    ; Automatically delete excess backups
  kept-new-versions 20     ; how many of the newest versions to keep
  kept-old-versions 5      ; and how many of the old
)
```

2.7 Secrets

```
(setq user-full-name "Marius Peter"
      user-mail-address "blendoit@gmail.com")

(setq user-full-name "Marius Peter"
      user-mail-address "blendoit@gmail.com")
```

3 Global key bindings

The following bindings strive to further enhance CUA² mode.

3.1 Navigation

```
(global-set-key (kbd "C-'"') 'delete-other-windows)
(global-set-key (kbd "C-s") 'save-buffer)
(global-set-key (kbd "C-r") 'counsel-recentf)
```

3.1.1 find-file

Open file with C-o.

```
(global-set-key (kbd "C-o") 'find-file)
```

3.1.2 Exit behaviours

The following bindings lead to more natural exit behaviors.

```
(defun delete-window-or-previous-buffer ()
  "Delete window; if sole window, previous buffer."
  (interactive))
```

²Common User Access.

```
(if (> (length (window-list)) 1)
    (delete-window
     (previous-buffer)))

(global-set-key (kbd "C-w") 'delete-window-or-previous-buffer)
(global-set-key (kbd "C-q") 'save-buffers-kill-terminal)
```

3.2 Mouse zoom

The typical binding on both GNU/Linux and MS Windows is adequate here: C-= to zoom in, C-- to zoom out.

```
(global-set-key (kbd "C--") 'text-scale-decrease)
(global-set-key (kbd "C-=") 'text-scale-increase)
(global-set-key (kbd "C-+") 'text-scale-increase)
```

4 Packages

Packages are collections of .el files providing added functionality to Emacs.

4.1 Package archives

List of package archives.

```
(require 'package)
(add-to-list 'package-archives '("melpa" . "https://melpa.org/packages/") t)
(add-to-list 'package-archives '("org" . "https://orgmode.org/elpa/") t)
(package-initialize)
```

4.2 use-package

Ensure use-package is installed, as well as all packages described in this configuration file.

```
(unless (package-installed-p 'use-package)
  (package-refresh-contents)
  (package-install 'use-package)
  (eval-when-compile (require 'use-package)))
(setq use-package-always-ensure t)
(require 'use-package)
(require 'bind-key)
```

4.3 TODO Convenient package update

One-function rollup of upgradeable package tagging, download and lazy install.

4.4 ivy

Auto completion.

```
(use-package ivy
  :config
  (setq ivy-use-virtual-buffers t
        ivy-count-format "%d/%d "
        enable-recursive-minibuffers t))
(ivy-mode t)
```

4.4.1 counsel

Wonderful counsellor!

```
(use-package counsel
  :bind ("M-x" . counsel-M-x)
  :config (counsel-mode t))

(global-set-key (kbd "C-f") 'counsel-grep-or-swiper)
```

4.4.2 swiper

```
(use-package swiper
  :bind (("C-f" . counsel-grep-or-swiper)))
```

4.5 TODO evil-mode

Forgive me, for I have sinned.

```
(use-package evil)
; (setq evil-toggle-key "C-c d") ; devil...
; (evil-mode 1)
```

4.6 org-mode

Phew, I can finally introduce Org mode! I am so **excited**.

Org mode replaces a word processor, a presentation creator, and a spreadsheet editor. IMHO, the spreadsheet ability captures more than 80% use cases wherein one wishes to include a table in a text document destined for physical publication. (It is clear that Excel spreadsheets are *not* destined for physical publication—simply attempt to print an Excel spreadsheet with the default settings.) In my opinion, Org mode matches all *useful* features of the Microsoft Office suite 1-to-1.

What follows are customizations designed to make Org mode behave more like Microsoft Word. The end goal is, once again, to draw as many new users to Emacs as possible!

4.6.1 Basic customization

Org base directory is in user home on GNU/Linux, or in AppData in MS Windows.

```
(setq org-directory (concat user-emacs-directory "~/org"))
```

First, we hide markup symbols for **bold**, *italic*, underlined and ~~strikethrough~~ text, and ensure our document appears indented upon loading:³

For the time being, I will in fact display emphasis markers, because hiding them corrupts tables.

```
(setq org-hide-emphasis-markers nil)
(setq org-startup-indented t)
```

Then, we customize Org headings to emulate WYSIWYG⁴ behavior normally found in Word:

4.6.2 Invisible edits

```
(setq org-catch-invisible-edits t)
```

4.6.3 Agenda

The agenda displays a chronological list of headings across all agenda files for which the heading or body contain a matching `org-time-stamp`.⁵

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

(defun my/find-diary-file ()
  "Load 'org-agenda-diary-file'."
  (interactive)
  (find-file org-agenda-diary-file))

(global-set-key (kbd "C-c d") 'my/find-diary-file)
```

4.6.4 Timestamps

More literary timestamps are exported to \LaTeX using the following custom format:

```
(setq org-time-stamp-custom-formats
  '("%d %b, %Y (%a)" . "%d %b, %Y (%a), at %H:%M"))
```

³It *appears* indented, but the underlying plaintext file does not contain tab characters!

⁴What You See Is What You Get (input and output are identical), as opposed to What You See Is What You Mean (the input contains instructions that can modify the output).

⁵An `org-time-stamp` can be inserted with `C-c .` (period)

4.6.5 \LaTeX export

The following makes CLOSED items appear green in \LaTeX . Very stylish, much flair!

```
(setq org-latex-inactive-timestamp-format
  "\\textcolor{ForestGreen!60}{\\textit{%s}}")
```

4.6.6 Publish

In the following *alist* (association list), we describe the projects publishable via org-publish. We separate the publishing of .org files and attachments, because an online tutorial recommended we do so.

```
(require 'ox-publish)
(setq org-publish-project-alist
  '(
    ("Safran-VIP-html"
     :base-directory "~/org/WORK/Safran/programs/B787/VIP/doc/org/"
     :base-extension "org"
     :publishing-directory "~/org/WORK/Safran/programs/B787/VIP/doc/wiki/"
     :recursive t
     :publishing-function org-html-publish-to-html
     :auto-preamble t
     :auto-sitemap t
     :sitemap-title "" )

    ("Safran-VIP-static"
     :base-directory "~/org/WORK/Safran/programs/B787/VIP/doc/org/"
     :base-extension "css\\|js\\|png\\|jpg\\|gif\\|pdf\\|mp3\\|mp4\\|ogg\\|swf"
     :publishing-directory "~/org/WORK/Safran/programs/B787/VIP/doc/wiki/"
     :recursive t
     :publishing-function org-publish-attachment )

    ("Safran-VIP-all"
     :components ("Safran-VIP-html" "Safran-VIP-static"))

    ("Safran-MA700-html"
     :base-directory "~/org/WORK/Safran/programs/MA700/doc/org/"
     :base-extension "org"
     :publishing-directory "~/org/WORK/Safran/programs/MA700/doc/wiki/"
     :recursive t
     :publishing-function org-html-publish-to-html
     :auto-preamble t
     :auto-sitemap t
     :sitemap-title "" )
```

```

("Safran-MA700-static"
 :base-directory "~/org/WORK/Safran/programs/MA700/doc/org/"
 :base-extension "css\\|js\\|png\\|jpg\\|gif\\|pdf\\|mp3\\|mp4\\|ogg\\|swf"
 :publishing-directory "~/org/WORK/Safran/programs/MA700/doc/wiki/"
 :recursive t
 :publishing-function org-publish-attachment )

("Safran-MA700-all"
 :components ("Safran-MA700-html" "Safran-MA700-static"))))

(add-to-list 'org-latex-packages-alist '("table" "xcolor"
  t ("pdflatex"))))
(add-to-list 'org-latex-packages-alist '("AUTO" "babel"
  t ("pdflatex"))))
(add-to-list 'org-latex-packages-alist '("AUTO" "polyglossia"
  t ("xelatex" "lualatex"))))

```

4.6.7 Export

This creates a shorter binding for the most common Org export: Org \rightarrow \LaTeX \rightarrow PDF

```

(defun my/org-quick-export ()
  "Org export to PDF and open.
  This basically reimplements 'C-c C-e l o'."
  (interactive)
  (org-open-file (org-latex-export-to-pdf)))

(global-set-key (kbd "C-c e") 'my/org-quick-export)

```

4.7 undo-tree

```
(global-undo-tree-mode)
```

4.8 dumb-jump

```

(use-package dumb-jump)
(add-hook 'xref-backend-functions #'dumb-jump-xref-activate)

```

4.9 gnuplot

```
(use-package gnuplot)
```

4.10 ledger

```
(use-package ledger-mode)
```

```
:bind
  ("C-c r" . ledger-report)
  ("C-c C" . ledger-mode-clean-buffer))
```

4.11 TODO Sidebar

Get inspiration from `ibuffer-sidebar` and create a better sidebar.

```
;; (load-file)
```

4.12 which-key

```
(use-package which-key
:init
  (which-key-mode)
;; :config
;; (setq which-key-idle-delay 1000)
;; (setq which-key-idle-secondary-delay 0.05)
;; (setq which-key-show-early-on-C-h t)
)
```

4.13 company

```
; (add-hook 'after-init-hook 'global-company-mode)
```

4.14 powerline

```
(use-package powerline)
(powerline-default-theme)
```

4.15 yasnippet

```
(use-package yasnippet)
(use-package yasnippet-snippets)
(yas-global-mode 1)
```

4.16 flycheck

```
(use-package flycheck
:init (global-flycheck-mode))
```

4.17 TODO flyspell

4.18 csv-mode

```
(use-package csv-mode)
```

4.19 json-mode

Oí, Jason!

Not needed in 27.1?

```
; (use-package json-mode)
```

4.20 magit

```
(use-package magit
  :bind ("C-c g" . magit-status))
```

4.21 pdf-tools

```
(use-package pdf-tools)
;; (pdf-tools-install)
```

4.22 dashboard

We replace the standard welcome screen with our own.

```
(setq inhibit-startup-message t)
(use-package dashboard
  :config
  (dashboard-setup-startup-hook)
  (setq dashboard-startup-banner (concat user-emacs-directory "img/Safran_logo.svg"))
  (setq dashboard-items '((recents . 5)
                          (projects . 5)))
  (setq dashboard-banner-logo-title "A modern professional text editor."))
```

4.23 rainbow-mode

This highlights hexadecimal numbers which look like colors, in that same color.

```
(use-package rainbow-mode
  :ensure t
  :init
  (add-hook 'prog-mode-hook 'rainbow-mode))
```

4.24 projectile

This enables us to better manage our .git projects.

```
(use-package projectile
  :bind ("C-c p" . 'projectile-command-map)
  :init (projectile-mode 1)
  (setq projectile-completion-system 'ivy))
```

4.25 all-the-icons

```
(use-package all-the-icons)
```

4.26 Better parentheses

```
(use-package rainbow-delimiters
  :config (add-hook 'prog-mode-hook #'rainbow-delimiters-mode))
(electric-pair-mode)
```

5 Cosmetics

5.1 Cursors

In order to imitate other modern text editors, we resort to a blinking bar cursor. We choose red, the most captivating color, because the cursor is arguably the region on our screen:

1. most often looked at;
2. most often searched when lost.

5.1.1 Default cursor

In files containing only fixed-pitch fonts (i.e. files containing only code), the cursor becomes a high-visibility box.

In files containing a mix of variable-pitch and fixed-pitch fonts, the cursor is a more MS Word-like bar.

```
(setq-default cursor-type (quote box))
(setq-default mixed-pitch-variable-pitch-cursor (quote bar))
```

5.2 Faces

- default: Hack
 - Legible, modern monospace font
 - Strict, sharp, uncompromising
- fixed-pitch: Hack
- variable-pitch: Liberation Sans
 - Libre alternative to Arial
- org-block: Hermit
 - Slightly wider than Hack
 - More opinionated shapes
 - Very legible parentheses

5.2.1 variable-pitch-mode

We use variable-pitch-mode for appropriate modes.

```
(add-hook 'org-mode-hook 'variable-pitch-mode)
(add-hook 'info-mode-hook 'variable-pitch-mode)
```

5.2.2 TODO Default font size

Make default font size larger on displays of which the resolution is greater than 1920x1080.

5.3 Initial and default frames

We set the dimensions of the initial and default frames.

```
(add-to-list 'default-frame-alist '(width . 100))
(add-to-list 'default-frame-alist '(height . 32))

(add-to-list 'initial-frame-alist '(width . 100))
(add-to-list 'initial-frame-alist '(height . 32))
```

5.3.1 GNU/Linux

These settings affect the first and subsequent frames spawned by Emacs in GNU/Linux. Frame transparency increases when focus is lost.

```
(when (and (display-graphic-p) (string-equal system-type "gnu/linux"))
  (set-frame-parameter (selected-frame) 'alpha '(90 . 50))
  (add-to-list 'default-frame-alist '(alpha . (90 . 50))))
```

5.4 Theme

My custom themes.

```
(setq custom-theme-directory (concat user-emacs-directory "themes/"))
(load-theme 'blendoit-light)
; (load-theme 'blendoit-dark)
```

5.4.1 blendoit-light

A highly legible and unambiguous theme.

The default face is a black foreground on a white background, this matches with MS Word.

Red TODO; cursor; negative; danger

Green DONE; positive

Blue document structure

Purple org-mode elements

5.4.2 blendoit-dark

Same principle

6 Editing preferences

These customizations enhance editor usability.

A line of text is considered ‘filled’ when it reaches 79 characters in length.

```
(setq-default fill-column 79)
```

We replace the longer yes-or-no-p questions with more convenient y-or-n-p.

```
(defalias 'yes-or-no-p 'y-or-n-p)
```

Disable minibuffer scroll bar.

```
(set-window-scroll-bars (minibuffer-window) nil nil)
```

Save cursor location in visited buffer after closing it or Emacs.

```
(save-place-mode 1)
```

Saving any file in user-emacs-directory (by default on Emacs, ~/emacs.d) shall byte-recompile the entire .emacs/ directory, for increased speed.

```
;; (defun my/byte-compile-user-config ()  
;;   "Byte-compile dotfiles if current file is in 'user-emacs-directory'.  
;;   Also tangles 'my-literate-config'.  
;;   (interactive)  
;;   (if (string-equal buffer-file-name my/literate-config)  
;;       (org-babel-tangle))  
;;   (if (string-prefix-p user-emacs-directory default-directory)  
;;       (byte-recompile-directory (concat user-emacs-directory "blendoit/") 0)))  
  
;; (add-hook 'after-save-hook my/byte-compile-user-config)
```

6.1 Clean up menus

Originally, I wished to inhibit certain entries in the GUI menus. Not worth the effort at this time.

```
(setq menu-bar-mode t)
```

6.2 Coding standards

This is just a better default. Don't @ me.

```
(setq c-default-style "linux"  
      c-basic-offset 4)
```


6.3 Dividers

This ensures users can resize windows using the GUI.

```
(menu-bar-bottom-and-right-window-divider)
```

6.4 Tabs

6.5 auto-fill

Automatically break lines longer than `fill-column`.

```
(add-hook 'org-mode-hook 'turn-on-auto-fill)
```

6.6 Recent files

```
(recentf-mode 1)
(setq recentf-max-menu-items 25)
(setq recentf-max-saved-items 25)
(run-at-time nil (* 5 60) 'recentf-save-list)
```

6.7 pop-up-frames

```
; (setq pop-up-frames (quote graphic-only))
```

7 Late setup

7.1 Profiling — stop

```
;; (profiler-stop)
```

7.2 Profiling — report

```
;; (profiler-report)
```

8 Conclusion

In this configuration file, we described a series of customization steps taken to make Emacs more palatable to modern IDE users.