# Blendoit’s literate GNU Emacs config

Marius Peter

*21 Jul, 2020 (Tue)*

## Contents

## List of Figures

## List of Tables

**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 Preliminary setup

### 1.1 Garbage collection

Increase the garbage collection limit.

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

### 1.2 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>.

```
(setq custom-file "~/.emacs.d/blendoit/custom/custom-file.el")
(load custom-file)
```

### 1.3 Customization shortcuts

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

```
(defun find-init-blendoit ()
"Jump to this very file."
(interactive)
(find-file "~/.emacs.d/blendoit/init-blendoit.org"))
(global-set-key (kbd "C-c c") 'find-init-blendoit)
```

Now, different shortcuts for other customization actions:

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

### 1.4 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
    )
```

### 1.5 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")
```

## 2 Bindings

### 2.1 Navigation

```
(global-set-key (kbd "C-s") 'save-buffer)
(global-set-key (kbd "C-b") 'ibuffer-sidebar-toggle-sidebar)
(global-set-key (kbd "C-w") 'kill-buffer-and-window)
(global-set-key (kbd "C-q") 'save-buffers-kill-terminal)
(global-set-key (kbd "C-o") 'menu-find-file-existing)
(global-set-key (kbd "C-r") 'counsel-recentf)
```

### 2.2 `Org`

Quick access to important Org mode functionalities.

#### 2.2.1 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`.[1]

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

#### 2.2.2 Export

This creates a shorter binding for the most common Org export: Org to LaTeX.

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

```
(defun blendoit-org-quick-export ()
  "Org export to PDF and open.
  This basically reimplements ‘C-c C-e l o’."
  (interactive)
    (org-latex-export-to-pdf)
    (org-open-file
      (concat (substring buffer-file-truename 0 -3) "pdf")))

(global-set-key (kbd "C-c e") 'blendoit-org-quick-export)
```

### 2.3 Mouse zoom

Zoom in/out of selected buffer using Alt + mouse wheel.

```
(global-set-key [M-mouse-4] 'text-scale-increase)
(global-set-key [M-mouse-5] 'text-scale-decrease)
```

## 3 Packages

List of package archives.

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

Ensure use-package is installed.

```
(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)
```

### 3.1 Org

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

```
(setq org-hide-emphasis-markers t)
(setq org-startup-indented t)
(setq org-directory "~/org")

(font-lock-add-keywords 'org-mode
'(("^ *\\([-]\\) "
   (0 (prog1 () (compose-region (match-beginning 1) (match-end 1) "•"))))))
```

```
(use-package org-bullets
  :config
  (add-hook 'org-mode-hook (lambda () (org-bullets-mode 1))))

(let* ((variable-tuple
(cond ((x-list-fonts "Liberation Sans") '(:font "Liberation Sans"))
      ((x-family-fonts "Sans Serif")    '(:family "Sans Serif"))
      (nil (warn "Cannot find a Sans Serif Font.  Install Source Sans Pro."))))
       (base-font-color     (face-foreground 'default nil 'default))
       (headline           `(:inherit default :weight bold)))

  (custom-theme-set-faces
   'user
   `(org-level-8 ((t (,@headline ,@variable-tuple))))
   `(org-level-7 ((t (,@headline ,@variable-tuple))))
   `(org-level-6 ((t (,@headline ,@variable-tuple))))
   `(org-level-5 ((t (,@headline ,@variable-tuple))))
   `(org-level-4 ((t (,@headline ,@variable-tuple :height 1.1))))
   `(org-level-3 ((t (,@headline ,@variable-tuple :height 1.25))))
   `(org-level-2 ((t (,@headline ,@variable-tuple :height 1.5))))
   `(org-level-1 ((t (,@headline ,@variable-tuple :height 1.75))))
   `(org-document-title ((t (,@headline ,@variable-tuple :height 2.0 :underline nil))))))

(custom-theme-set-faces
 'user
 '(variable-pitch ((t (:family "Liberation Sans"))))
 '(fixed-pitch ((t ( :family "Hack")))))
```

```
(x-list-fonts "Hermit")
```

- Beautiful bullet lists in Org mode!

#### 3.1.1 Org Agenda

#### 3.1.2 Org publish

```
(require 'ox-publish)
(setq org-publish-project-alist
'(
  ("Safran-VIP-html"
   :base-directory "~/org/WORK/Safran/programs/VIP/doc/org/"
   :base-extension "org"
   :publishing-directory "~/org/WORK/Safran/programs/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/VIP/doc/org/"
   :base-extension "css\\|js\\|png\\|jpg\\|gif\\|pdf\\|mp3\\|mp4\\|ogg\\|swf"
   :publishing-directory "~/org/WORK/Safran/programs/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")))
```

## 3.2 `gnuplot`

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

## 3.3 `company`

## 3.4 Ledger

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

## 3.5 ibuffer-sidebar

```
(use-package ibuffer-sidebar)
```

## 3.6 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)
)
```

## 3.7 Ivy

Auto completion.

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

  (use-package ivy-hydra)
```

### 3.7.1 Counsel

Wonderful counsellor!

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

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

#### 3.7.2 Swiper

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

### 3.8 Ido

### 3.9 Company

```
(use-package company
 :config (add-hook 'after-init-hook 'global-company-mode))
```

### 3.10 Flycheck

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

### 3.11 Magit

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

### 3.12 PDF-tools

```
(use-package pdf-tools
 :config (pdf-loader-install))
```

## 4 Cosmetics

### 4.1 Faces & cursors

In order to imitate other modern text editors, we’ll resort to a blinking bar cursor.

#### 4.1.1 Default cursor

```
(setq-default cursor-type (quote bar))
```

#### 4.1.2 Default faces

Fixed-pitch and variable-pitch fonts will be used intelligently in all text modes.

```
(use-package mixed-pitch
  :hook
  ;; If you want it in all text modes:
  (org-mode . mixed-pitch-mode))
```

### 4.2 All the icons

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

### 4.3 Theme

```
;;  (use-package zenburn-theme
;;  :config
;;  (load-theme 'zenburn)
;; )
  (load-theme 'wombat)
```

### 4.4 Transparency

```
(add-to-list 'default-frame-alist '(alpha . (90 . 50)))
```

### 4.5 Scrollbars

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

## 5 Editing preferences

These customizations enhance editor usability.

```
(setq-default fill-column 79)
(defalias 'yes-or-no-p 'y-or-n-p)
```

### 5.1 Initial & subsequent frame size

### 5.2 Better parentheses

```
(use-package smartparens
  :config
  (add-hook 'prog-mode-hook 'smartparens-mode))

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