Emacs Config

will@wmedrano.dev

May 26, 2025

Contents

1 Packages **2**
- 1.1 Melpa . . . 2
- 1.2 Initialize . . . 2

2 Startup **2**

3 Backups & Autosaves **3**

4 Appearance & Feel **3**
- 4.1 Remove Clutter . . . 3
- 4.2 Lines . . . 3
- 4.3 Color Scheme . . . 4
- 4.4 Modeline . . . 4

5 Editor Completions **4**
- 5.1 Ivy . . . 4
- 5.2 Counsel . . . 4

6 Formatting **5**
- 6.1 Tabs . . . 5
- 6.2 Line Width . . . 5

7 Languages **6**
- 7.1 Rust . . . 6
- 7.2 Org Mode . . . 6

1 Packages

1.1 Melpa

Add Melpa to the package manager. Melpa contains many popular Emacs packages.

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

Counting the default package archives, the following package archives are available.

```
(cl-loop for package-archive in package-archives
         collect (list (car package-archive) (cdr package-archive)))
```

gnu	`https://elpa.gnu.org/packages/`
nongnu	`https://elpa.nongnu.org/nongnu/`
melpa	`https://melpa.org/packages/`

Package archives must be manually refreshed or fetched with `M-x package-refresh-contents`. All packages added to the `package-selected-packages` variable can be installed with `M-x package-install-selected-packages`. Packages can also be installed on a one-off basis interactively with `M-x package-install`.

In some cases, package installation may fail with "... not found". This likely means that the package archives point to an old (and non-existant) version of the package. The package definitions can be updated by running `M-x package-refresh-contents`. Packages may also be upgraded all at once with `M-x package-upgrade-all`.

1.2 Initialize

Initialize the package archive. This makes all previously installed packages available.

```
(package-initialize)
```

2 Startup

Disable the default Emacs startup screen. Instead, this displays just the opened file or the `*Scratch*` buffer if no file has been opened.

```
(setq-default inhibit-startup-screen t)
```

3 Backups & Autosaves

Backup and autosaves may litter the filesystem so we disable them. This is ok as my disk is reliable, I save often, and use version control.

```
(setq-default auto-save-interval 0
              create-lockfiles   nil
              make-backup-files  nil)
```

4 Appearance & Feel

4.1 Remove Clutter

Remove the menu bar and tool bar.

```
(menu-bar-mode -1)
(tool-bar-mode -1)
```

Disable the scroll bar. The functionality is ok sometimes, but it clashes with the theming.

```
(scroll-bar-mode -1)
```

4.2 Lines

Scroll conservatively values above 100 cause Emacs to scroll the minimum number of lines required to get the cursor in position. The default value of 0 causes Emacs to recenter the window.

```
(setq-default scroll-conservatively 101)
```

Display line numbers for text buffers. This can be toggled in an individual buffer with `M-x display-line-numbers-mode`.

```
(global-display-line-numbers-mode t)
```

Highlight the currently selected line. This can be toggled in an individual buffer with `M-x hl-line-mode`.

```
(global-hl-line-mode t)
```

4.3 Color Scheme

Use the `doom-dracula` theme from the Doom Themes package.

```
(add-to-list 'package-selected-packages 'doom-themes)
(load-theme 'doom-dracula t)
```

4.4 Modeline

Use Doom Modeline to display a nicer modeline. Mainly, it:

- Uses more icons.
- Displays a minimal amount of information while still keeping important information such as:
  - Syntax errors
  - Version control information

```
(add-to-list 'package-selected-packages 'doom-themes)
(doom-modeline-mode t)
```

5 Editor Completions

Editor completions refers to auto complete done within the editor context, as opposed to code. Editor completion is used to complete prompts for things such as selecting a file, buffer, or command.

5.1 Ivy

Editor completions are displayed using the Ivy package. This provides a huge improvement over the default built-in Emacs completion.

```
(add-to-list 'package-selected-packages 'ivy)
(ivy-mode t)
```

5.2 Counsel

Counsel provides functions that wrap ivy completion with some extra features. For example, `counsel-M-x` is an `M-x` replacement that also displays a keybinding if there is an active keybinding for the particular function.

```
(add-to-list 'package-selected-packages 'counsel)
(counsel-mode t)
```

Enabling `counsel-mode` makes the `counsel-mode-map` keymap active. This keymap defines several rebinds.

```
counsel-mode-map
```

keymap (24 keymap (98 . counsel-switch-buffer)) (remap keymap (bookmark-jump . counsel-book

However, it does not provide a rebind for `counsel-switch-buffer`. We make this our default (interactive) switch buffer command as it allows previewing the contents of a buffer before switching.

```
(define-key counsel-mode-map (kbd "C-x b") #'counsel-switch-buffer)
```

6 Formatting

6.1 Tabs

Emacs uses a combination of tabs and spaces when auto-indenting. This pleases neither the spaces nor tabs crowds. Tabs are disabled to prevent the mixed use, though opinionated languages will still find a way to use their correct default. For example, Go will still use tabs when indenting.

```
(setq-default indent-tabs-mode nil)
```

Use a default tab width of 4 spaces.

```
(setq-default tab-width 4)
```

6.2 Line Width

Set a target line width of 80. Contents of a "paragraph" may be made to follow the target line width through `M-x fill-paragraph` (default keybind `M-q`) or a highlighted region with `M-x fill-region`.

```
(setq-default fill-column 80)
```

Some languages have a different target line length.

```
(defun fill-column-100 ()
  (setq-local fill-column 100))

(add-hook 'rust-mode-hook #'fill-column-100)
```

7 Languages

7.1 Rust

```
(add-to-list 'package-selected-packages 'rust-mode)
```

7.2 Org Mode

Enable syntax highlighting for exported material.

```
(add-to-list 'package-selected-packages 'htmlize)
```

Enable previews while editing org document. Previews can be enabled with `M-x org-preview-html-mode`. Behind the scenes, this exports to HTML on save on displays the generated HTML in an `*eww*` buffer.

```
(add-to-list 'package-selected-packages 'org-preview-html)
```