Bogdan's EMACS Config

Bogdan Popa

April 30, 2014

Contents

1 This Document

The purpose of this document is mostly to help me remember all of my settings as well as to help newbies set up their environment and get started using my configuration.

2 Installation

2.1 Linux

This depends on the distribution. Most of them will have EMACS in their repositories.

```
sudo apt-get install emacs-snapshot-gtk
```

2.2 Mac OS X

Download and install EMACS for OS X from EMACS for Mac OS X.

2.3 Windows

Download and install EMACS for Windows from the GNU Repositories.

3 Setup

3.1 Unix

Clone the repo into your home directory and then fetch its submodules.

```
cd ~/
git clone https://github.com/Bogdanp/.emacs.d.git
cd .emacs.d
```

```
git submodule init
git submodule update
```

3.2 Windows

Windows makes things a bit harder. Add an environment variable called `HOME` and make it point to `C:/Users/Username/`. Clone the `windows` branch of the repository in that folder and fetch its submodules.

4 Bindings

The most important EMACS binding is `C-g`. Use it as an escape hatch for when you mess up your key combinations. VIM users: `C-g` is your `ESC` in EMACS.

4.1 General

`M-x` Bring up extended command mode.

`C-x C-c` Quit.

`C-x C-i` Bring up `imenu`.

`C-x C-e` Evaluate last sexp.

4.2 Getting Help

`C-h f` Describe function.

`C-h v` Describe variable.

`C-h k` Describe key combination.

4.3 Moving Around

`C-s` Search forward.

`C-p` Goto prev line.

`C-n` Goto next line.

`C-a` Goto beginning of line.

`C-e` Goto end of line.

`C-b` Goto prev character.

`C-f` Goto next character.

`M-b` Goto prev word.

`M-f` Goto next word.

`M-<` Goto beginning of buffer.

`M->` Goto end of buffer.

`C-w` Kill word backward.

`M-d` Kill word.

4.4 Buffer Management

`C-x C-s` Save.

`C-x C-w` Save as.

`C-x C-b` List buffers.

`C-x b` Switch buffers.

`C-x f` Open.

4.5 Window Management

These suck compared to the EVIL alternatives but they (mostly `C-x o`) will come in handy when dealing with `<E>` buffers.

`C-x 0` Delete current window.

`C-x 1` Delete other windows.

`C-x 2` Split window vertically.

`C-x 3` Split window horizontally.

`C-x o` Goto other window.

`C-c M-a` toggle between the current window state and a fullscreen terminal.

4.6 Popups

`q` Will close most popups, make sure you are in EMACS mode before you use it (you're in EMACS mode if there's an `<E>` in your status bar).

5 Backups

`undo-tree-mode` is enabled globally. All backups - including `undo-tree` files - are saved in `temporary-file-directory`.

6 dired+

You can bring up dired by pressing `C-j` on a folder whilst in `ido-find-file` (`C-x C-f`). You can also bring it up with `C-x d`.

6.1 Bindings

`n` Goto next line.

`p` Goto prev line.

`+` Create directory.

`x` Execute changes.

`D` Delete file immediately.

`d` Mark file for deletion.

`g` Refresh buffer.

`m` Mark file.

`u` Unmark file.

`R` Rename file(s).

`U` Unmark all files.

7 EVIL

7.1 Differences From Standard EVIL

My EVIL mode fork makes some changes to the way registers are handled. Most importantly, while you are inside an EVIL mode buffer you can only access the clipboard through the `+` register and nothing else, this prevents EMACS from dirtying the clipboard whenever you cut things.

The copy-on-motion bug is fixed.

`C-w` now works in the minibuffer.

7.2 Common Bindings

You can drop into EMACS mode for a single command using `\`.

`C-z` Toggle between EVIL and EMACS mode.

7.3 Normal Mode Bindings

`SPC` Bring up ace jump mode.

`S-SPC` Bring up ace char jump mode.

`C-w f` Toggle window fullscreen.

7.4 Visual Mode Bindings

The `C-a`, `C-e`, `C-p`, `C-n` bindings are available in visual mode.

7.5 Insert Mode Bindings

The `C-a`, `C-e`, `C-p`, `C-n` bindings are available in insert mode.

`C-w` Kill word backward.

`C-r` Yank from register.

8 Magit

Magit is fucking great. You can find its official manual here.

8.1 Generic Bindings

`n` Goto next object.

`p` Goto prev object.

`TAB` Expand/collapse object.

`RET` Open object.

8.2 Status Bindings

Use `C-c m` to bring `magit-status` up.

`S` Stage everything.

`s` Stage object under point.

`u` Unstage object under point.

`b*` Branch operations.

`c*` Commit operations.

`f*` Fetch operations.

`F*` Pull operations.

`P*` Push operations.

8.3 Branch Bindings

Use `bv` inside `magit-status` to bring up the visual branch manager.

9 Org

9.1 EMACS Bindings

9.1.1 Agenda

`C-c .` Insert date.

`C-u C-c .` Insert date and time.

9.1.2 Buffer

`C-x n s` Narrow buffer.

`C-x n w` Widen buffer.

9.1.3 Document

`C-c C-e` Export document.

9.1.4 Headlines

`TAB` Expand/collapse headline.

`S-TAB` Expand/collapse everything.

`C-RET` Add new headline at the current level.

`M-RET` Add new headline/list item at the current level.

`M-→` Increase headline level.

`M-←` Decrease headline level.

`S-M-→` Increase headline level incl. children.

`S-M-←` Decrease headline level incl. children.

9.1.5 Movement

`C-c C-u` Goto parent headline.

`C-c C-b` Prev same-level headline.

`C-c C-f` Next same-level headline.

`C-c C-p` Prev headline.

`C-c C-n` Next headline.

9.1.6 Source Code

`C-c'` Open up the code in a source code block in a separate buffer for editing.

`<sTAB` Insert source code block. You need to be in insert or EMACS mode to use this one.

9.2 EVIL Bindings

9.2.1 Agenda

`, a` Bring up agenda (this one is globally-available).

9.2.2 Headlines

`, t` Toggle headline state.

`-` Cycle list bullet.

`<` Decrease headline level.

`>` Increase headline level.

9.2.3 Movement

`gu` Goto parent headline.

`gj` Goto next same-level headline.

`gk` Goto prev same-level headline.

10 Prodigy

Prodigy is bound to `C-c p`.

10.1 Services

Services should be defined in `config/init-prodigy.el`. The following is an example service:

```
1:  (prodigy-define-service
2:    :name "Python Server"
3:    :command "python"
4:    :args '("-m" "SimpleHTTPServer" "8093")
5:    :cwd "/path/to/my/project/"
6:    :tags '(example)
7:    :kill-process-buffer-on-stop t)
```

10.2 Bindings

`n` Next process.

`p` Prev process.

`s` Start process.

`S` Stop process.

`r` Restart process.

`$` View process log.

`F` Clear filters.

`f t` Filter by tag.

`f n` Filter by name.

`j d` Open dired for the current process.

`j m` Open magit for the current process.

11 Haskell

You will need a recent version of `ghc` and `cabal` as well as the following packages: `hlint`, `ghc-mod`, `ghci-ng`, `structured-haskell-mode`, `stylish-haskell`.

11.1 Structured Haskell Mode

See the Official Repository.

11.2 Bindings

`C-c C-l` Load the buffer into a REPL.

`C-c C-t` View type of object at point.

12 PHP

PHP is supported through `php-mode` and `web-mode`. Use `web-mode` for mixed HTML and PHP files and `php-mode` for pure PHP files. The default mode for PHP files is `web-mode`.

12.1 Flycheck

Install `phpcs` from PEAR and you should be good to go.

13 Python

13.1 Flycheck

Install `flake8` to use it as a backend for Flycheck.

13.1.1 Ignoring Certain Errors

Create a `.flake8rc` file in your `HOME` directory. For example:

```
1:  [flake8]
2:  ignore = E501,F403,E712
```

13.1.2 Bindings

`C-c !n` Goto next error.

`C-c !p` Goto prev error.

`C-c !l` List errors.

13.2 Jedi

Install `virtualenv` and `epc` and then run `M-x jedi:install-server`.

13.2.1 Bindings

`C-c .` Goto definition (and `C-c ,`).

`C-c ?` Show documentation of the object at point.

14 Scala

View the ENSIME manual at the Official Repository. Start up ENSIME in a Scala buffer with `M-x ensime RET`.

15 Scheme

15.1 Setup

Link `mzscheme` and `racket` so that they can be found in `PATH`.

15.2 Geiser mode

Start it up with `M-x run-geiser`.

15.2.1 Buffer Bindings

`C-c C-a` Switch to the REPL and enter the current module.

`C-c C-z` Switch between the buffer and the REPL.

`C-M-x` Eval definition around point.

`C-c M-e` Eval definition around point and switch to REPL.

`C-c C-x` Eval sexp before point.

`C-c C-r` Eval region (also `C-c M-r`).

`C-c C-b` Eval buffer (also `C-c M-b`).

`M-g n` Jump to next error.

`M-g p` Jump to prev error.

15.2.2 REPL Bindings

`C-c C-q` Kill Scheme process.

`C-c M-o` Clear REPL.