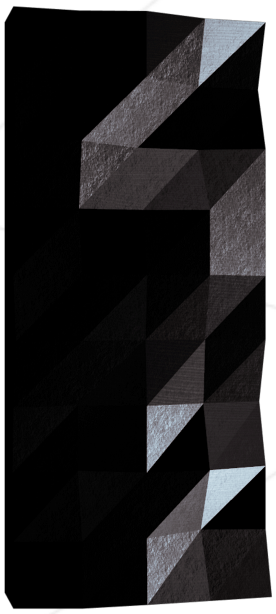


Mexican Standoff

Text editors - Advanced features

42 Staff pedago@42.fr

Summary: This rush is intended to make you see the possibilities (infinite ?) of your text editors.

Contents

Chapter I

Foreword

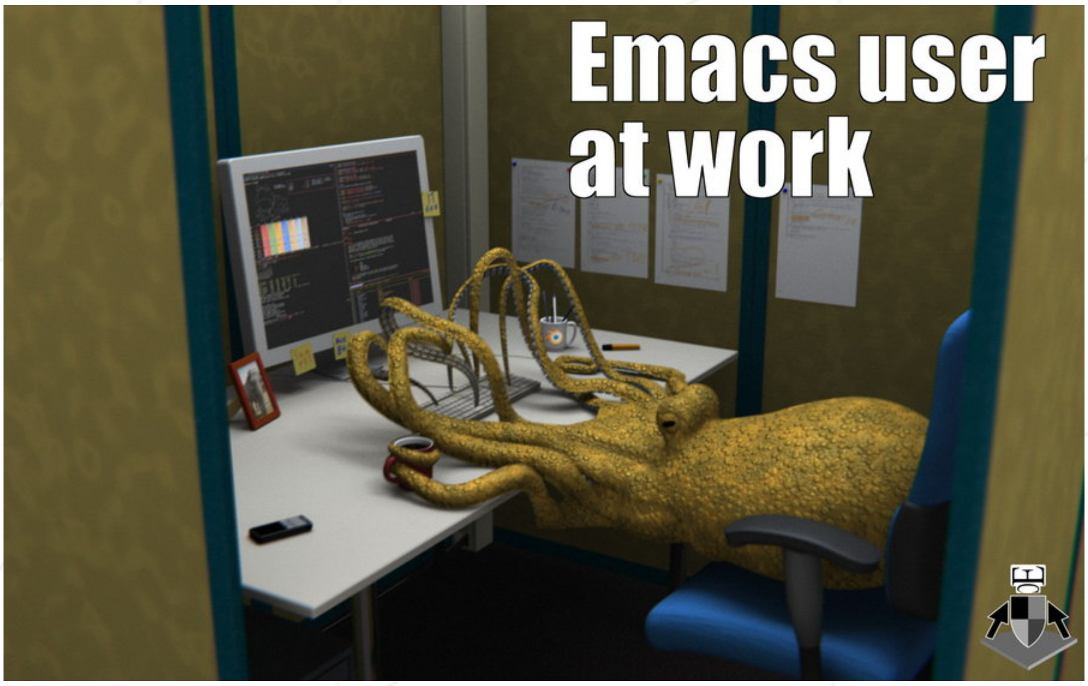
how to quit smoking

how to quit **smoking**

how to quit **facebook**

how to quit **sugar**

how to quit **vim**



(Source.)

Chapter II

Introduction

This rush is intended to make you glimpse the features (infinite ?) of the two mammoths of the text edition : `emacs` and `vim`.

Today, you will overcome old feuds and will use both! The opportunity to get rid of your prejudices (trade them for certainty is better), and on the other hand never again need to ask how we get out of this publisher ****** opened by mistake, shit.

Chapter III

Objectives

The main purpose of this rush is to encourage you to take ownership of your text editor, whatever it may be. Text editors are very complete, basic tools, but you will see that it is possible to customize them to make them stick to your needs.

Hard to really understand all the possibilities of a publisher until you have tried to configure it, learn its shortcuts, or program your own modes!

The goal is to start a process. It's up to you, the finished rush, to look for the text editor that really suits you (it can be `emacs`, it can be `vim`, it can be the Evil mode on `emacs`... it can also be completely something else), and make it your own.

Chapter IV

Instructions

The following three exercises have to be performed for `emacs` AND for `vim`. No more, no less.

IV.1 Configuration files

You will first tackle the configuration of your editors.

Write the files `.emacs` and `.vimrc` possessing at least the following configurations :

- A C code is automatically indented with tabs
- A pair (parenthesis or brace) is automatically closed when you know the opening element
- The position column of the cursor is displayed
- Two side-by-side spaces are highlighted
- A whitespace before a line break is highlighted
- Backup files (ending with ~) are archived in a specific folder inside the folders `~/.emacs.d` and `~/.vim`, respectively.

`You have the right to use plugins.`

IV.2 Keyboard shortcuts

To get the most out of your text editor, it is advisable to go through learning the keyboard shortcuts. Using the shortcuts will be laborious as long as the muscle memory has not taken over, but as soon as the automatisms are in place you will gain productivity!

You will find on the project page a file `normalize_me.c`.

You must find the smallest combination (keys and shortcuts) possible to put this file to the norm (as the norminette authentic). You have the right to use only the shortcuts of the basic configuration, and not to binder commands that make coffee (neither in rendering nor in defense).

You will save your combinations sequences in the files `normalize_emacs` and `normalize_vim`.

IV.3 Header 42

You will now put your hands in the programming itself, by recoding the generation of the header 42.

Header here :

```
/* ************************************************************************** */
/*                                                                            */
/*                                                        :::      ::::::::   */
/*   header.c                                           :+:      :+:    :+:   */
/*                                                    +:+ +:+         +:+     */
/*   By: laurie <laurie@staff.42.fr>                +#+  +:+       +#+        */
/*                                                +#+#+#+#+#+   +#+           */
/*   Created: 2015/12/02 14:40:24 by laurie            #+#    #+#             */
/*   Updated: 2015/12/02 14:40:24 by laurie           ###   ########.fr       */
/*                                                                            */
/* ************************************************************************** */
```

You must respect the following constraints :

- The email is retrieved in the MAIL variable of the environment.
- You have to binder the generation of the header on the combination <Ctrl-c h> on `emacs`, and <Ctrl-c Ctrl-h> on `vim`.
- Press <Ctrl-c h> must generate a header ONLY if a correctly formatted header is not already present.
- The header generation will only be tested on C files. You do not have to manage the file mode.
- The “Updated” line should only be updated if the buffer has been modified since the last backup. Backing up without making changes should not update this line.

- Nothing is forbidden. (In the case of cheating obviously, you will have to be able to explain your code in defense.)

`Start by reading an introduction to Elisp and Vimscript, you'll quickly get an idea of what you can use to realize the header.`

Chapter V

Bonus part

Bonuses will only be evaluated if all of the previous questions have been validated, for both text editors.

Here are some bonus ideas to make :

- Repeat previous exercises with another text editor
- Customize the header according to the mode
- Configure and use a syntax checker (type `flymake` for `emacs`)
- Write a minor mode that will highlight your standard mistakes

Chapter VI

Turn-in and peer-evaluation

- Your rendering will be corrected only by humans.
- The files `.emacs`, `.vimrc`, `normalize_emacs` and `normalize_vim` have imposed names, for others you are free to name them as you wish.
- Files related to `emacs` will be in an eponymous folder at the root of the rendering, likewise for files related to `vim`.
- You will have, in defense, comment and explain your code. Likewise, you need to know what the shortcuts do without having to use them.
- You must return to the root of your rendering repository a file named `author` containing the logins of the two members of the group followed by a '\n', such as :

```
$> cat -e ./author
login1$
login2$
$>
```