Spacemacs

•••

Vim? Emacs? ¿Porque no los dos?

Attendance

https://tinyurl.com/spacemacsextratation

¿Porque no los dos?

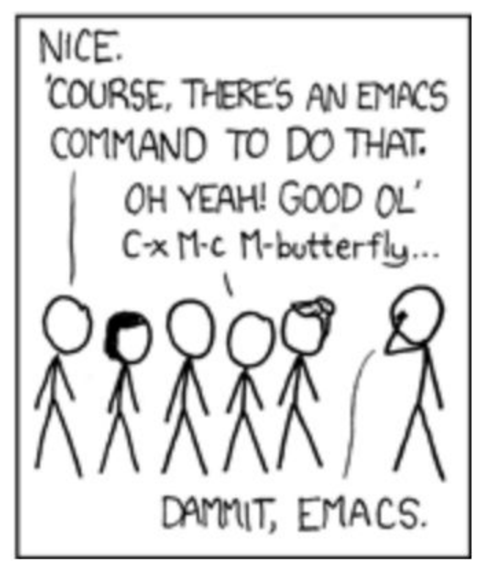
Emacs Background

History of Emacs

- Originally grew out of the ITS operating system in 1976 at MIT
- Modern Emacs dates from 1984 with the GNU project to reimplement UNIX
- Like Vi, it predates modern GUIs and uses different terminology from both Vi and modern applications

Emacs Philosophy

- Endlessly customizable
- Non-modal editing
  - No normal, insert, or visual modes
  - All keybindings work the same, all the time
- All-encompassing
  - Vim is intended as a small tool as part of a workflow involving Bash and shell commands
  - Emacs wants you to stay in Emacs
    - The continuation of the Symbolics Lisp Machine


NICE.
'COURSE, THERE'S AN EMACS COMMAND TO DO THAT.
OH YEAH! GOOD OL' C-x M-c M-butterfly...
DAMMIT, EMACS.

Emacs and Vim as languages, and as platforms

- Think of Vim and Emacs keybindings as languages for manipulating text
    - Many text editors implement Vim and Emacs keybindings as an extension
    - Vim and Emacs have extensions that implement each other
- Also think of them as platforms for building editors
    - How configurable an editor is can be as important as the default editing experience

Emacs is not (originally) a Unix program

- The original developers of Emacs were Lisp and AI hackers
- The original developers of Unix were C programmers
  - Actually, C and UNIX were quite literally written for each other
- This is one reason why the Emacs/Vi tension exists

Emacs as a platform

- Emacs is a platform for building text editors
  - It isn’t very useful without customization
- There exist “starter packs” similar to our GPI .vimrc to simplify / complicate editing

Emacs Lisp

```
  ;; Eshell C-l support
  (defun eshell-clear-buffer ()
    "Clear terminal"
    (interactive)
    (let ((inhibit-read-only t))
      (erase-buffer)
      (eshell-send-input)))
  (add-hook 'eshell-mode-hook
            '(lambda()
               (local-set-key (kbd
"C-l") 'eshell-clear-buffer)))
```

- Configuration language for Emacs
- Somewhat archaic
  - Dynamic scope
  - Weakly typed
  - Parentheses!!!
- Powerful enough to write very useful extensions

Dot Emacs (.emacs)

- Emacs equivalent of the .vimrc
  - Single file holding configuration information written in Emacs Lisp
  - .emacs.d directory often used to hold multiple files

Demo: .emacs

Emacs modes

- Emacs is a non-modal editor
  - Contrast with Vim, which has Normal, Insert, and Visual modes
- But Emacs has modal editing at a **higher level**
- Which keys do which things depends on
  - What type of file you are editing
  - Arbitrary “hooks” to change keybindings in response to events
  - Manually changing the modes with M-x

Major and minor modes

- Each “buffer” is associated with exactly one major mode and some list of minor modes
- Modes can “inherit” from other modes
    - For example, prog-mode is the parent of python-mode and c++-mode
- Each major mode redefines tons of keybindings in ways that are useful for the particular file you are editing
- Minor modes are less intrusive, and define smaller sets of commands for editing tasks that are useful in many different file types
    - Like spellcheck through flyspell-mode

Vim emulation using evil-mode

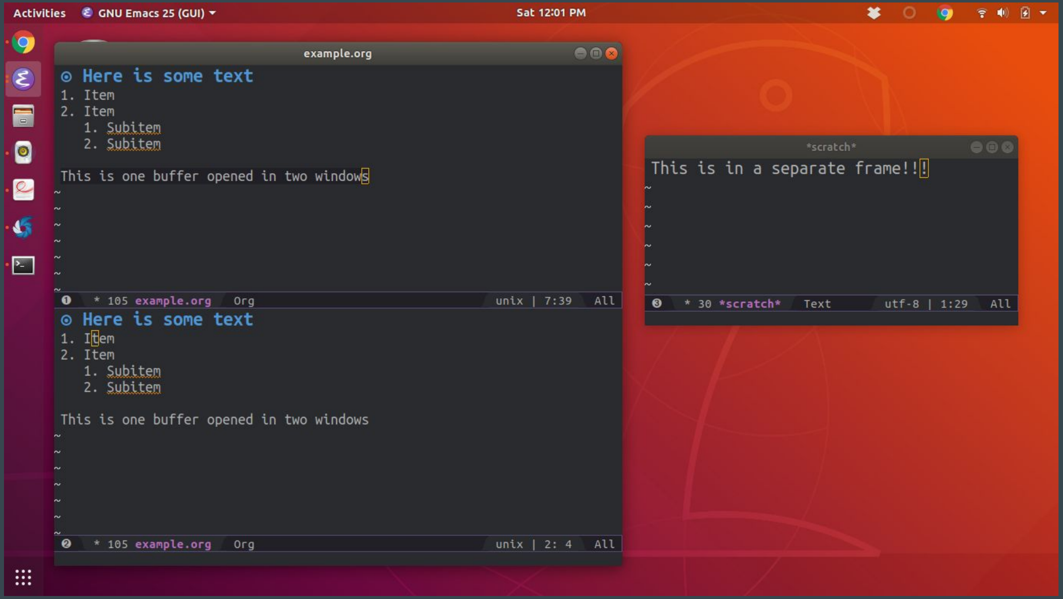
- Evil-mode changes the Emacs editing language to the vim editing language
  - All of the keybindings you are familiar with from class will work
  - Emacs M-x commands are still available

Getting help in Emacs

- C-h i
    - Open Emacs manual
- C-h b
    - List all bindings in scope
- C-h c
    - Describe what command is bound to a key

Emacs terminology for GUIs is kind of weird

- Windowing commands
- Buffers vs Windows vs Frames
- Split view on a single buffer



```
⊙ Here is some text
1. Item
2. Item
   1. Subitem
   2. Subitem

This is one buffer opened in two windows
~
~
~
~
~
~
~
❶  * 105 example.org   Org                    unix | 7:39   All
⊙ Here is some text
1. Item
2. Item
   1. Subitem
   2. Subitem

This is one buffer opened in two windows
~
~
~
~
~
~
~
❷  * 105 example.org   Org                    unix | 2: 4   All
```

```
This is in a separate frame!!
~
~
~
~
~
~
❸  * 30 *scratch*   Text        utf-8 | 1:29   All
```

Demo: windows

Spacemacs

Spacemacs

http://spacemacs.org

Four pillars:

1. Mnemonic
2. Discoverable
3. Consistent
4. Crowd-Configured

A community-driven Emacs distribution

The best editor is neither Emacs nor Vim, it's Emacs *and* Vim!

Demo: Spacemacs in the browser

1. Mnemonic

- Remember that Vim is a language. It’s helpful to subvocalize, for example:
  - `di(` as the phrase “delete inside parentheses”
  - `“qyw` as the phrase “register ‘q’ yank word”
- With Spacemacs, this language is heavily extended from core editing commands to basically everything
  - `SPC ff` becomes “<base command key> file find”
  - `SPC gs` becomes “<base command key> git status”

2. Discoverable

- Remember that Emacs is “self-documenting”, with universal keybindings to discover exactly the command you need to run at the time you need it
- Spacemacs improves discoverability by showing submenus of available keybindings as you type
- Also has fuzzy file / buffer / everything search with Helm mode

The most important command

<SPC> <SPC>

3. Consistent

- All layers (collections of packages) that ship with Spacemacs are documented and conform to conventions
- Spacemacs makes the choice that backwards compatibility and stability are secondary to having a consistent editor
  - When you update, things will break
  - They can only really get away with this because everything is so discoverable...

4. Crowd-Configured

- When was the last time a normal person wrote an Eclipse or Visual Studio plugin?
- Spacemacs is completely open source and developed transparently on Github
- Emacs Lisp and great documentation lowers the barrier for entry to contribute to the project

Installation

- Spacemacs turns your .emacs.d directory into a git repository
- Just run the following command and start Emacs:
    - git clone https://github.com/syl20bnr/spacemacs ~/.emacs.d
- Note: the Emacs on Andrew is too old
- Note: Bash on Windows should work, but without extra configuration Emacs will only work in terminal

.spacemacs

- This is where user configuration is stored
- Sets which layers are installed
- Custom Lisp code typically goes in the dotspacemacs/user-config function

Spacemacs Layers

- Spacemacs organizes extensions into layers by topic
- Can also install emacs extensions manually from the ELPA or MELPA repositories
    - Or just copy the Emacs Lisp files into .emacs.d

Emacs packages

tramp-mode

- Edit files over ssh
- Very useful for editing files on Andrew
- Open file dialog (SPC-f-f in Spacemacs) and use the following syntax
  - ssh:andrew:private/gpi-labs

Demo

shell-mode

- Open bash inside of Emacs
- M-x shell

Demo

magit-mode

- Git client for Emacs

Demo

org-mode

- Outline and todo list
  - Great for note-taking
  - Can write latex inline

Demo

avy-mode

- Like the t and f commands from Vim, but more general
- Simplify movement across entire files

Demo

helm

- File browsing and completion framework
  - Find commands and files

Demo

docview-mode

- View pdfs and images inside of emacs
- Works in combination with tramp-mode to view writeup files on Andrew

Demo

flycheck-mode

- Syntax-checking for several different programming languages
- IDE-like discovery of syntax errors before compilation

Language-specific modes

- Most languages have their own spacemacs layer providing an editing mode specific to that language