INFINUM

Command Line as an IDE

MARKO IVANEK

IDEs

IDE

- “An integrated development environment (IDE) is a software application that provides comprehensive facilities to computer programmers for software development.”

- An application that integrates powerful tools that make a developers job easier

DRAWBACKS

- Slow
- Monolythic
- Hardly portable
- (Sometimes) Single-platform
- (Sometimes) Single-language

The Unix Philosphy

“Do one thing, but do it well”

IDE FEATURES

- Text editor
- Project management
- Syntax checking
- Code completion
- Code navigation (File opening, jump to method definitions)
- Version control system integration
- Build and testing tools (rake tasks, tests)

THE BASE

THE BASE – TMUX

- Terminal multiplexer
- Scriptable
- Session–based
- Shareable

THE TEXT EDITOR

TEXT EDITOR

- (Neo)Vim
- Modal, extensible, light-weight, etc.
- Excellent integration with Tmux through plugins

PROJECT MANAGEMENT

VIM-PROJECT

- Vim plugin for managing projects
- Opens a list of available projects when starting Vim
- Requires manual definitions of projects in .vimrc
- Hasn’t been updated in 2 years

TMUXINATOR

- Gem for managing tmux projects
- Configure pane and window layout, define project root directory, execute custom commands while starting a project, etc.
- Define project configuration through YAML files

TMUX SESSIONS

- Sessions – attach and detach from sessions
- Tmux processes are cheap, so you can have dozens of them open at the same time
- tmux-sessionist – Convenient keybindings for session management
- tmux-resurrect – Store and restore session information, persist sessions between restarts

6 SOURCE CODE NAVIGATION

FILE TREES

- NERDTree – Plugin that offers a standard, familiar file tree implementation

- NERDTree isn't opened automatically in new tabs

- vim-nerdtree-tabs enables that functionality – NOT RECOMMENDED

FUZZY FILE FINDERS

- Vim plugins
- CtrlP is the most popular
- fzf offers a (Neo)Vim plugin as well

JUMP TO FILE UNDER CURSOR

- gf – open file path under cursor

- Looks in places defined by the ‘path’ variable and tries to append suffixes defined in ‘suffixesadd’

- Those can be defined locally in a buffer

- Usually configured automatically by language-specific file-type plugins

- vim-rails offers context-aware gf

JUMP TO METHOD DEFINITION

- Exuberant Ctags + Vim

- (Neo)Vim offers CTags integration out of the box
  - <C-]> - jump to definition of method under cursor

- vim-tags plugin - enables tag auto-generation on file save, supports Ruby's bundler

TAGBAR

- “A class outline viewer”
- Requires Exuberant Ctags to be installed
- Automatically creates tags for the current buffer in-memory

SYNTAX CHECKERS

SYNTAX CHECKERS

- Command-line syntax checkers or static analysis tools
- Vim plugin(s)

SYNTASTIC (VIM)

- Supports 98 languages out of the box
- Configurable (on save checks, on open checks, only manual checks) and extensible
- Runs syntax checkers synchronously

NEOMAKE (NEOVIM)

- Based on Syntastic
- Supports only 34 languages out of the box, but is also much younger than Syntastic
- Run checkers asynchronously

CODE COMPLETION

CODE COMPLETION

- (Neo)Vim support code completion out of the box
- Seperate completion mode – accessed w/ C-x
- Offers numerous completion modes
- General completion, whole line completion, tag completion, omni completion, spelling completion, dictionary completion, file name completion

GENERIC COMPLETION

- Uses multiple sources for populating possible matches
- By default, looks for matches in the current buffer, all loaded buffers and tags files (if available)
- You can change sources with :set complete (:h ‘complete’)
- Triggered with <C-n> and <C-p> in insert mode

TAG COMPLETION

- Searches through the tags file for possible matches
- Triggered with <C-x><C-]>

OMNI COMPLETION

- Implemented as a file-type plugin, so needs 'set nocompatible' and 'filetype plugin on' options to be set

- Vim supports around a dozen languages out of the box, support for more can be added through file-type plugins

- vim-ruby offers omni completion for Ruby

- Triggered with <C-x><C-o>

WHOLE LINE COMPLETION

- Completes the whole line
- Triggered with <C-x><C-l>

COMPLETION PLUGINS

- YouCompleteMe – a Vim completion engine that can also use external completion engines (CLang for C family of languages, Jedi for Python, etc.)
- Neocomplete
- Deoplete (NeoVim)

VERSION CONTROL SYSTEM INTEGRATION

COMMAND LINE – TMUX / VIM

- Use git from the command line in a seperate Tmux pane/ split

- Execute git commands directly from Vim by prepending ! in command line mode

FUGITIVE.VIM

- “I’m not going to lie to you; fugitive.vim may very well be the best Git wrapper of all time.” – Tim Pope, Fugitive author

- Browse whole history of a repository, browse whole history of a file, convenience wrappers around mv/rm/etc., stage changes, commit changes, resolve merge conflicts and much, much more

- http://vimcasts.org/blog/2011/05/the-fugitive-series/

RUNNING TESTS AND BUILDS

RUNNING FROM TMUX

- Run manually from another pane / window
- Boring...

RUNNING FROM VIM

- Since Tmux is scriptable, Vim can be extended to easily send commands to Tmux

- Run test, builds, compilations with a Vim shortcut or command

DISPATCH.VIM

- Send arbitary commands to async adapters (tmux, screen, iterm, headless, etc.)

- General-purpose plugin for sending any command to tmux
- Used as a base by various wrappers for language or build specific tasks / commands

VIM-TEST

- Plugin for running tests from Vim
- Supports various languages and testing frameworks
- Supports various strategies for running tests (Dispatch.vim, Neovim terminal emulator, Vimux, etc.)

08

DEBUGGING

DEBUGGING

- Command line tools (irb, pry, gdb, etc)
- Dispatch.vim, Vimux, etc.
- Easy acces to REPLs

INFINUM

Thank you!

Visit infinum.co or find us on social networks:

infinum.co infinumco infinumco infinum