Let's Play with Vanilla Vim

@hezbymuhammad
VimConf 2019
About Me
@hezbymuhammad

Indonesia Vim Community
(https://t.me/VimID)

Software Engineer at Quipper
Note

Purposes:

> Embrace awesomeness of Vim
> Vanilla Vim is also great for doing **productive** things
> Understand Vim

Level:

> **Beginner** users
> **Casual** users
Been 3 months since I use vanilla vim for coding Ruby...
Agenda

1. Vim Way
2. Useful Navigations
3. Netrw
4. File Search
5. Utilizing Tags
6. Searching
Always use HJKL whenever possible

ADM-3A full keyboard.
Modes

Normal mode is your best friend
<ul>
  <li>Eklat</li>
  <li>Kacang</li>
  <li>Pisang</li>
</ul>
<ul>
  <li>Coklat</li>
  <li>Kacang</li>
  <li>Pisang</li>
</ul>
<ul>
  <li>Apple</li>
  <li>Banana</li>
  <li>Orange</li>
</ul>
Useful Navigation
## Navigating

<table>
<thead>
<tr>
<th>Keyboard Shortcuts</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>f + &lt;char&gt;</td>
<td>Inline search next for <code>&lt;char&gt;</code> with pointer at <code>&lt;char&gt;</code></td>
</tr>
<tr>
<td>F + &lt;char&gt;</td>
<td>Inline search previous for <code>&lt;char&gt;</code> with pointer at <code>&lt;char&gt;</code></td>
</tr>
<tr>
<td>t + &lt;char&gt;</td>
<td>Inline search next for <code>&lt;char&gt;</code> with pointer before <code>&lt;char&gt;</code></td>
</tr>
<tr>
<td>T + &lt;char&gt;</td>
<td>Inline search previous for <code>&lt;char&gt;</code> with pointer before <code>&lt;char&gt;</code></td>
</tr>
<tr>
<td>ctrl + f</td>
<td>Scroll one page</td>
</tr>
<tr>
<td>ctrl + b</td>
<td>Scroll one page backward</td>
</tr>
<tr>
<td>H</td>
<td>Go to [line] from top window, default is first line</td>
</tr>
<tr>
<td>M</td>
<td>Go to middle line</td>
</tr>
<tr>
<td>L</td>
<td>Go to [line] from bottom window, default is last line</td>
</tr>
</tbody>
</table>
The quick brown fox jumps over the lazy dog. The quick brown fox jumps over the lazy dog. The quick brown fox jumps over the lazy dog. The quick brown fox jumps over the lazy dog.
The quick brown fox jumps over the lazy dog.
## Split Management

<table>
<thead>
<tr>
<th>Shortcut</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctrl + w _</td>
<td>Max height current split</td>
</tr>
<tr>
<td>ctrl + w</td>
<td>Max width current split</td>
</tr>
<tr>
<td>ctrl + w =</td>
<td>Normalize split</td>
</tr>
<tr>
<td>ctrl + w o</td>
<td>Close all split except current split</td>
</tr>
<tr>
<td>ctrl + w T</td>
<td>Convert split to tab</td>
</tr>
</tbody>
</table>
The quick brown fox jumps over the lazy dog.
What is it?

“NerdTree is a project drawer, netrw is a split explorer”

http://vimcasts.org/blog/2013/01/oil-and-vinegar-split-windows-and-project-drawer/
## Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>:Explore</td>
<td>Open netrw</td>
</tr>
<tr>
<td>:Vexplore</td>
<td>Open netrw in vertical split</td>
</tr>
<tr>
<td>:Sexplore</td>
<td>Open netrw in horizontal split</td>
</tr>
</tbody>
</table>
NERDTree’s Flavour

"NERDTree's flavour
let g:netrw_banner = 0
let g:netrw_liststyle = 3
let g:netrw_browse_split = 4
let g:netrw_altv = 1
let g:netrw_winsize = 15
Recursive Completer

:set path+=**
:set wildmode=longest:full,full
:set wildignore+=**/node_modules/**,**/public/**
VIM - Vi IMproved

version 8.1.2200
by Bram Moolenaar et al.
Vim is open source and freely distributable

Sponsor Vim development!
type :help sponsor<Enter> for information

type :q<Enter> to exit
type :help<Enter> or <F1> for on-line help
type :help version8<Enter> for version info
Utilizing Tags
Vim Complete Module

'complete' 'cpt' string (default: ".,w,b,u,t,i")
local to buffer
{not in Vi}

This option specifies how keyword completion ins-completion works
when CTRL-P or CTRL-N are used. It is also used for whole-line
completion i_CTRL-X_CTRL-L. It indicates the type of completion
and the places to scan. It is a comma separated list of flags:

.  scan the current buffer ('wrapscan' is ignored)
w  scan buffers from other windows
b  scan other loaded buffers that are in the buffer list
u  scan the unloaded buffers that are in the buffer list
U  scan the buffers that are not in the buffer list
k  scan the files given with the 'dictionary' option
kspell use the currently active spell checking spell
k{dict} scan the file {dict}. Several 'k' flags can be given,
patterns are valid too. For example:
:scat cpt=k/usr/dict/*,k-/spanish

s  scan the files given with the 'thesaurus' option
s[tar] scan the file [tar]. Several 's' flags can be given, patterns
are valid too.

i  scan current and included files
d  scan current and included files for defined name or macro
   i_CTRL-X_CTRL-D
]  tag completion
t  same as ']'
Ctags

brew install ctags
alias ctags="`brew --prefix`/bin/ctags"
# ruby example
ctags -R --languages=ruby --exclude=.git --exclude=log .

<table>
<thead>
<tr>
<th>Ctrl + ]</th>
<th>Open method definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctrl + w ]</td>
<td>Open method definition in horizontal split</td>
</tr>
<tr>
<td>ctrl + t</td>
<td>Back to method call</td>
</tr>
</tbody>
</table>
~/Code/play/frab  \ master  \vimvanilla

[Note: In Ruby not all definitions are supported by ctags out of the box]
Introduce Ag

:set grepprg=ag\ --nogroup\ --nocolor
:command -nargs=+ -complete=file -bar Ag silent! grep! <args>|cwindow|redraw!
nnoremap K :grep! "\b<C-R><C-W>\b"<CR>:cw<CR>
~/Code/play/vanillavim  cp  lorem.html  lorem-copy.html
Summary
Thank you