Let’s Play with Vanilla Vim

@hezbymuhammad
VimConf 2019

shorturl.at/acJKM

About Me

@hezbymuhammad

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

Software Engineer at Quipper

Expand

Learning

Horizon

Note

Purposes:

> Embrace **awesomeness** of Vim

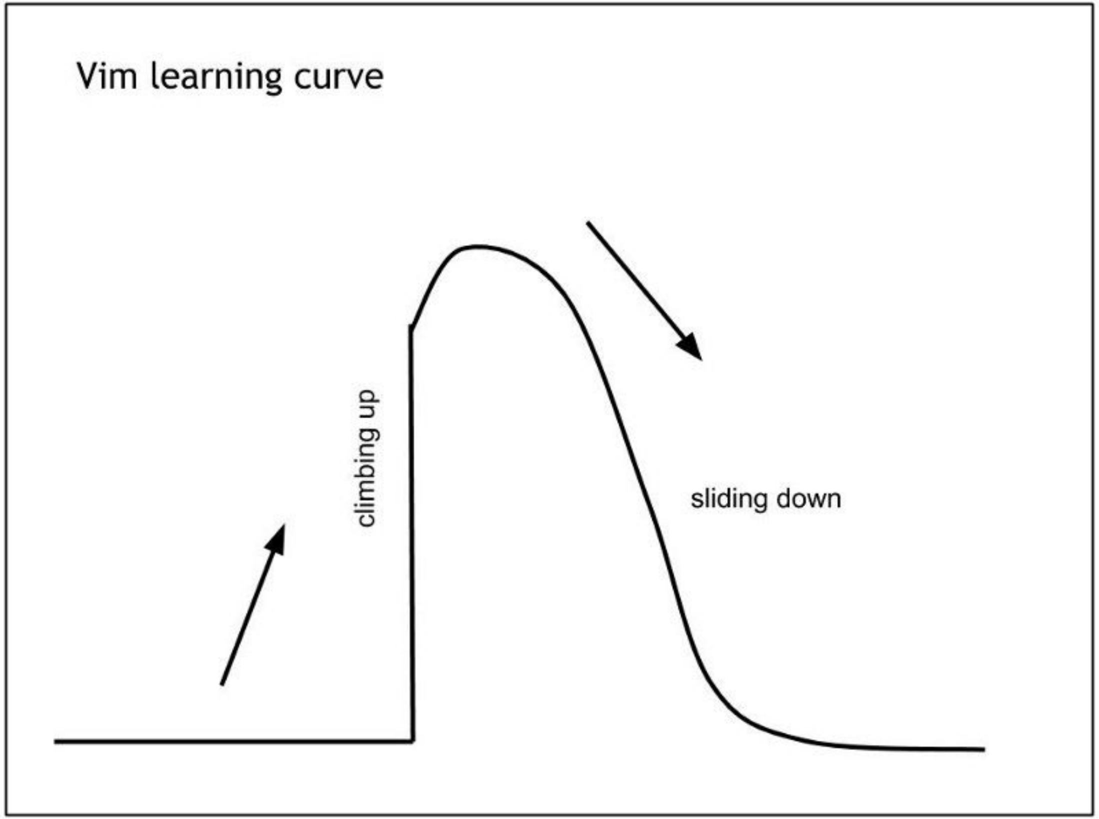
> Vanilla Vim is also great for doing **productive** things

> **Understand** Vim

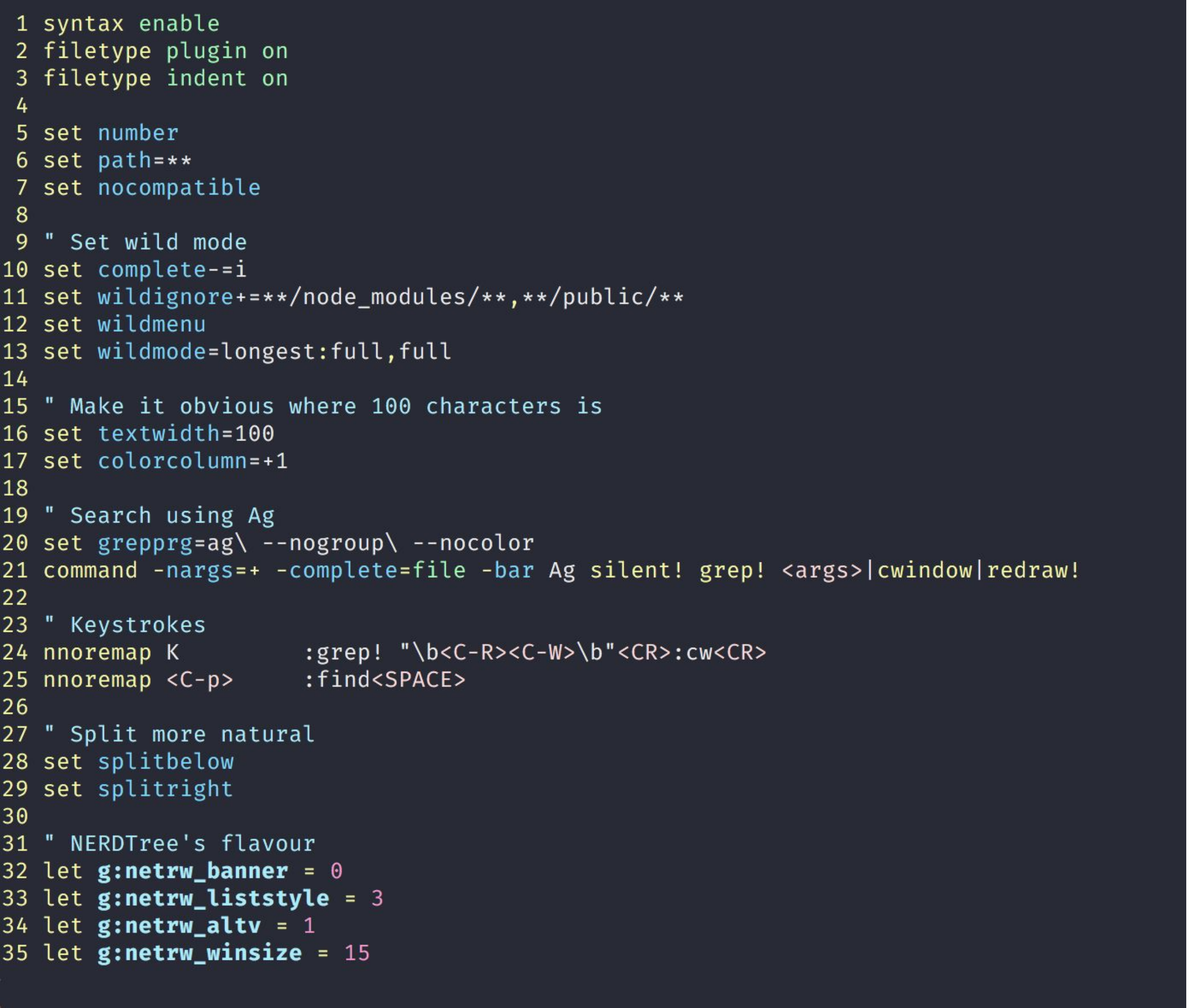
Level:

> **Beginner** users

> **Casual** users



Been 3 months since I use vanilla vim for coding Ruby...



```
syntax enable
filetype plugin on
filetype indent on

set number
set path=**
set nocompatible

" Set wild mode
set complete-=i
set wildignore+=**/node_modules/**,**/public/**
set wildmenu
set wildmode=longest:full,full

" Make it obvious where 100 characters is
set textwidth=100
set colorcolumn=+1

" Search using Ag
set grepprg=ag\ --nogroup\ --nocolor
command -nargs=+ -complete=file -bar Ag silent! grep! <args>|cwindow|redraw!

" Keystrokes
nnoremap K         :grep! "\b<C-R><C-W>\b"<CR>:cw<CR>
nnoremap <C-p>     :find<SPACE>

" Split more natural
set splitbelow
set splitright

" NERDTree's flavour
let g:netrw_banner = 0
let g:netrw_liststyle = 3
let g:netrw_altv = 1
let g:netrw_winsize = 15
```

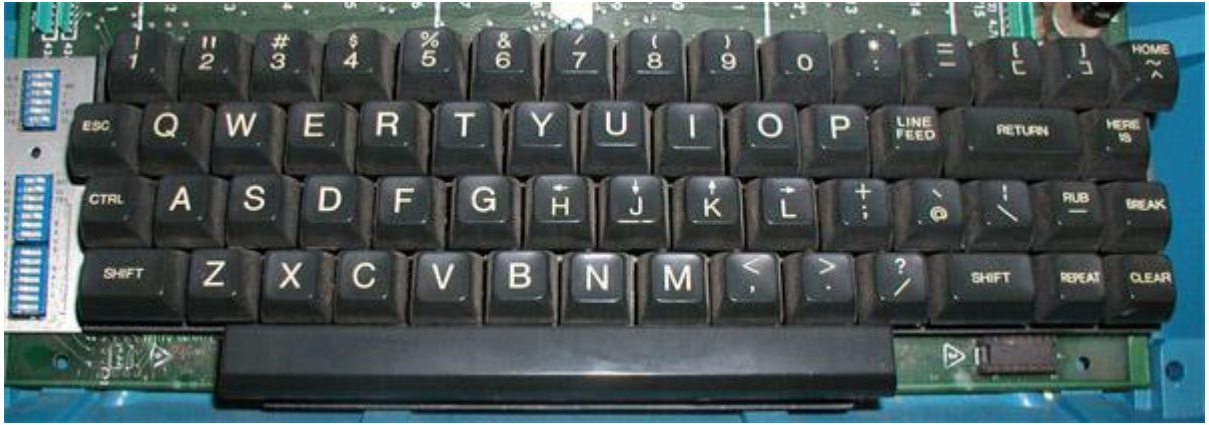
Agenda

1. Vim Way
2. Useful Navigations
3. Netrw
4. File Search
5. Utilizing Tags
6. Searching

Vim Way

HJKL

Always use HJKL whenever possible

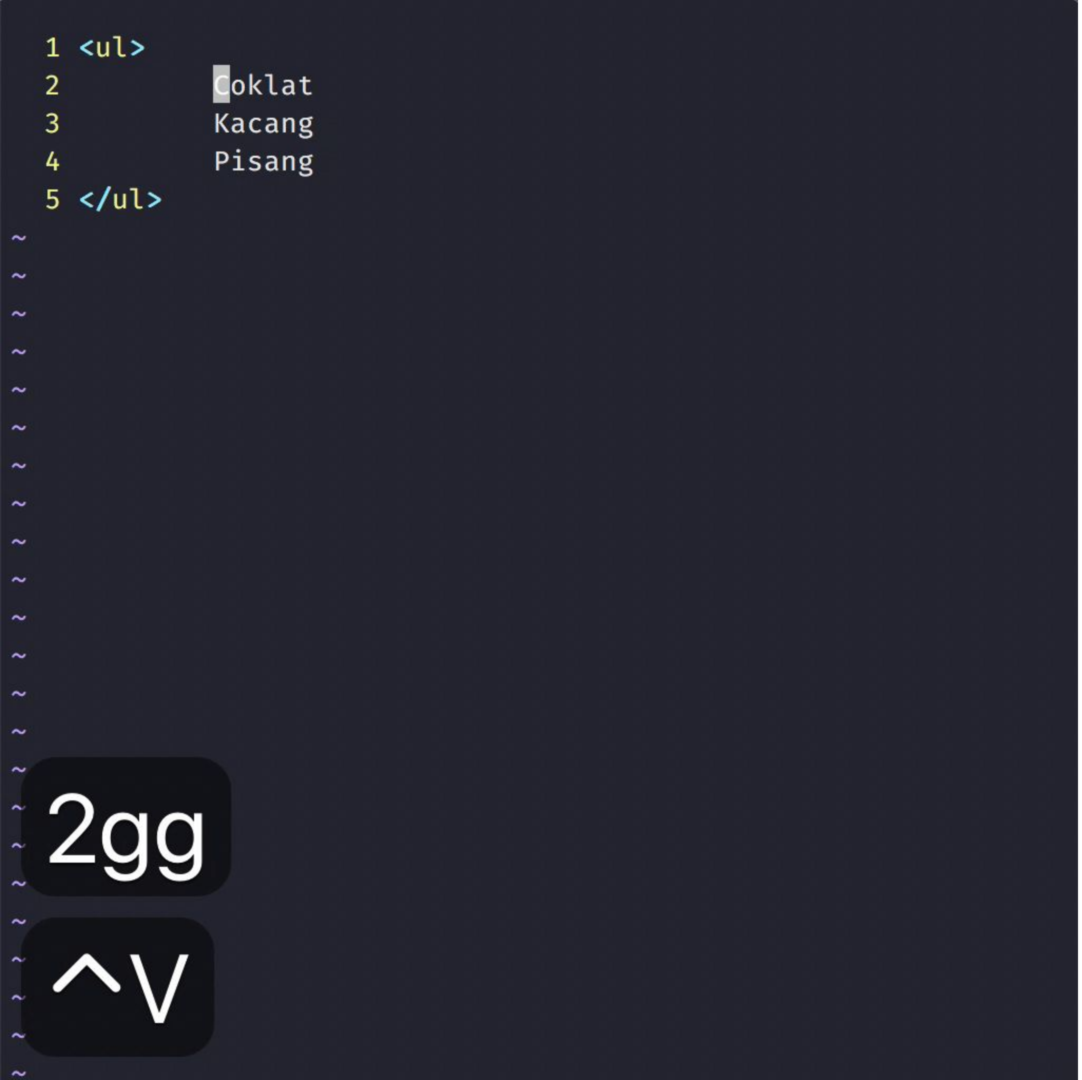


ESC Q W E R T Y U I O P LINE FEED RETURN HERE IS
CTRL A S D F G H J K L RUB BREAK
SHIFT Z X C V B N M SHIFT REPEAT CLEAR
HOME

ADM-3A full keyboard.

Modes

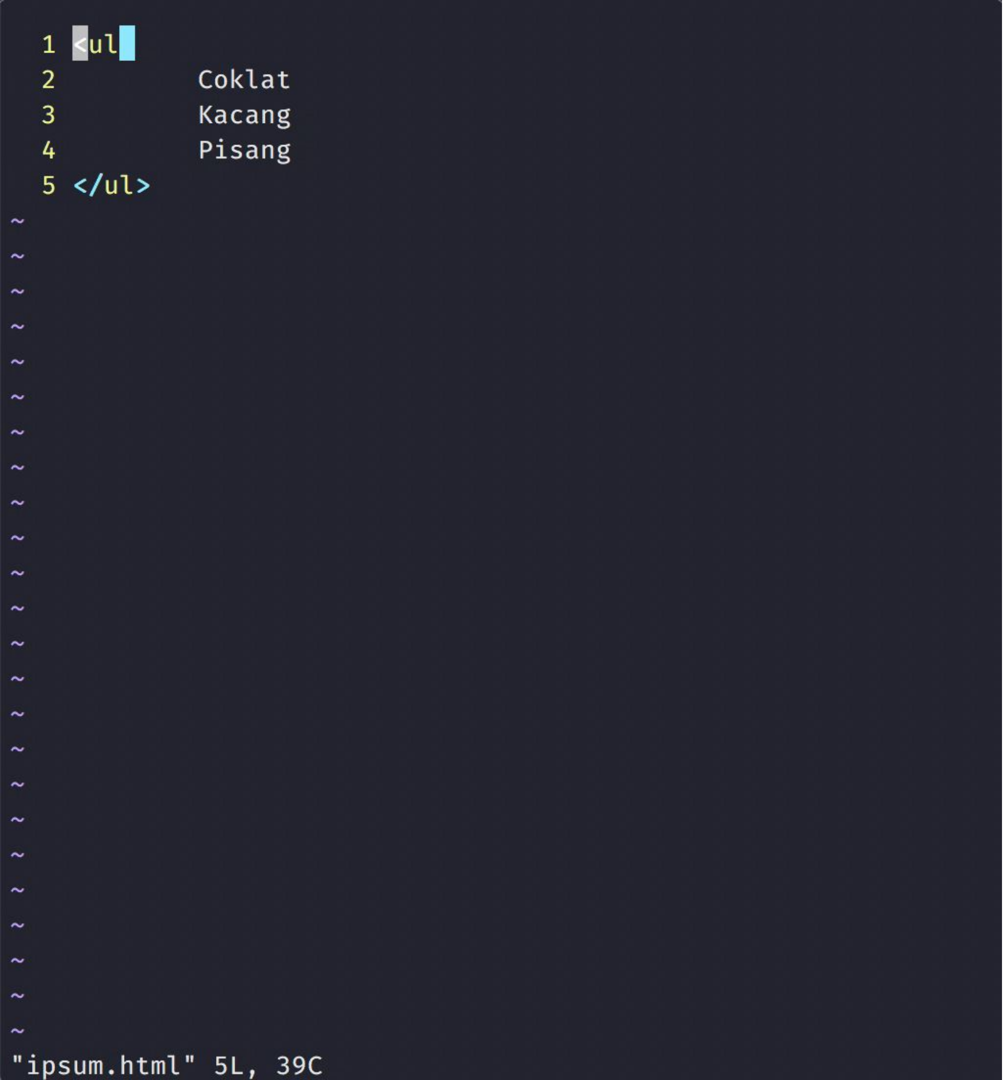
Normal mode is your best friend



```
<ul>
        Coklat
        Kacang
        Pisang
</ul>
```

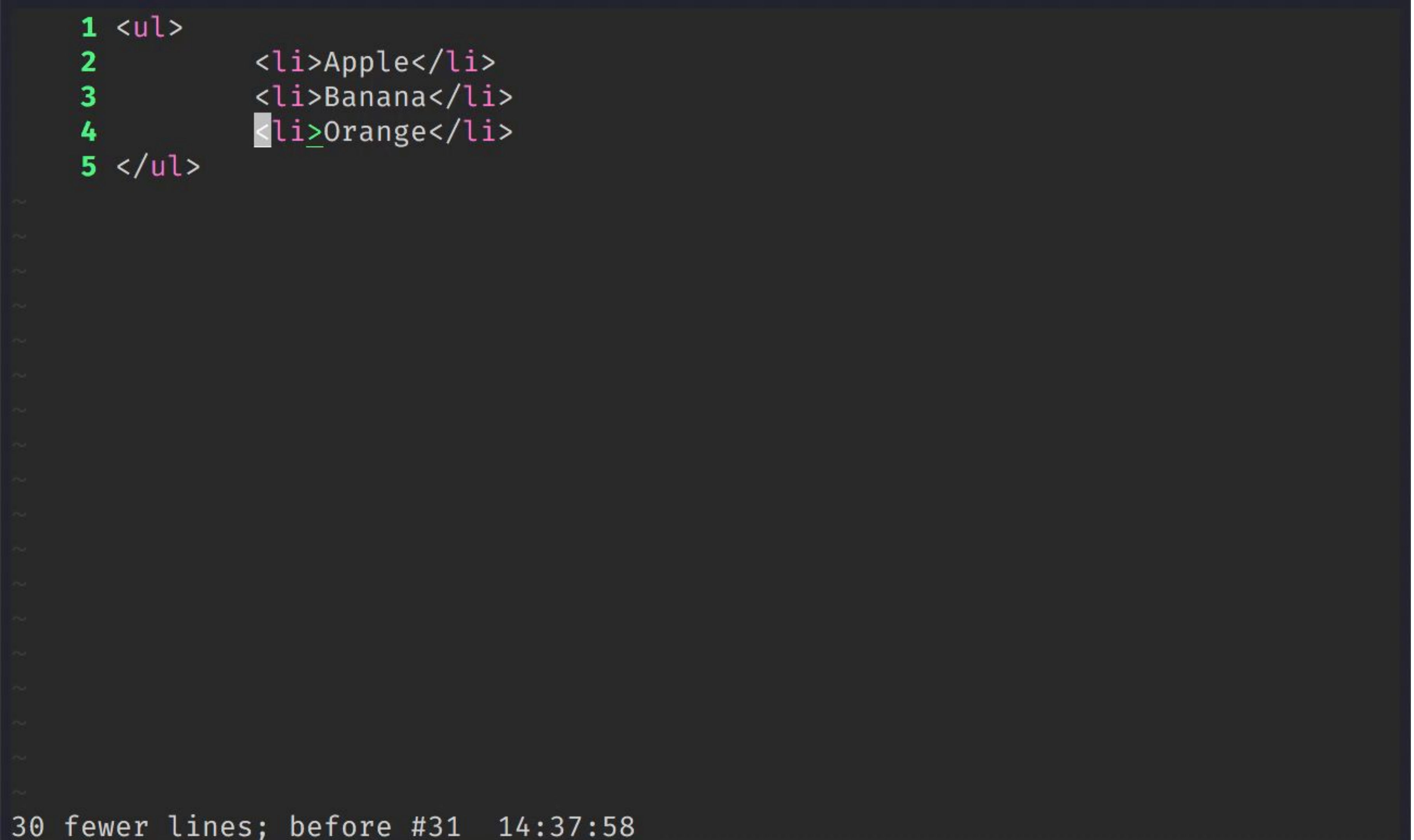
2gg

^V



```
<ul
        Coklat
        Kacang
        Pisang
</ul>
```

```
"ipsum.html" 5L, 39C
```



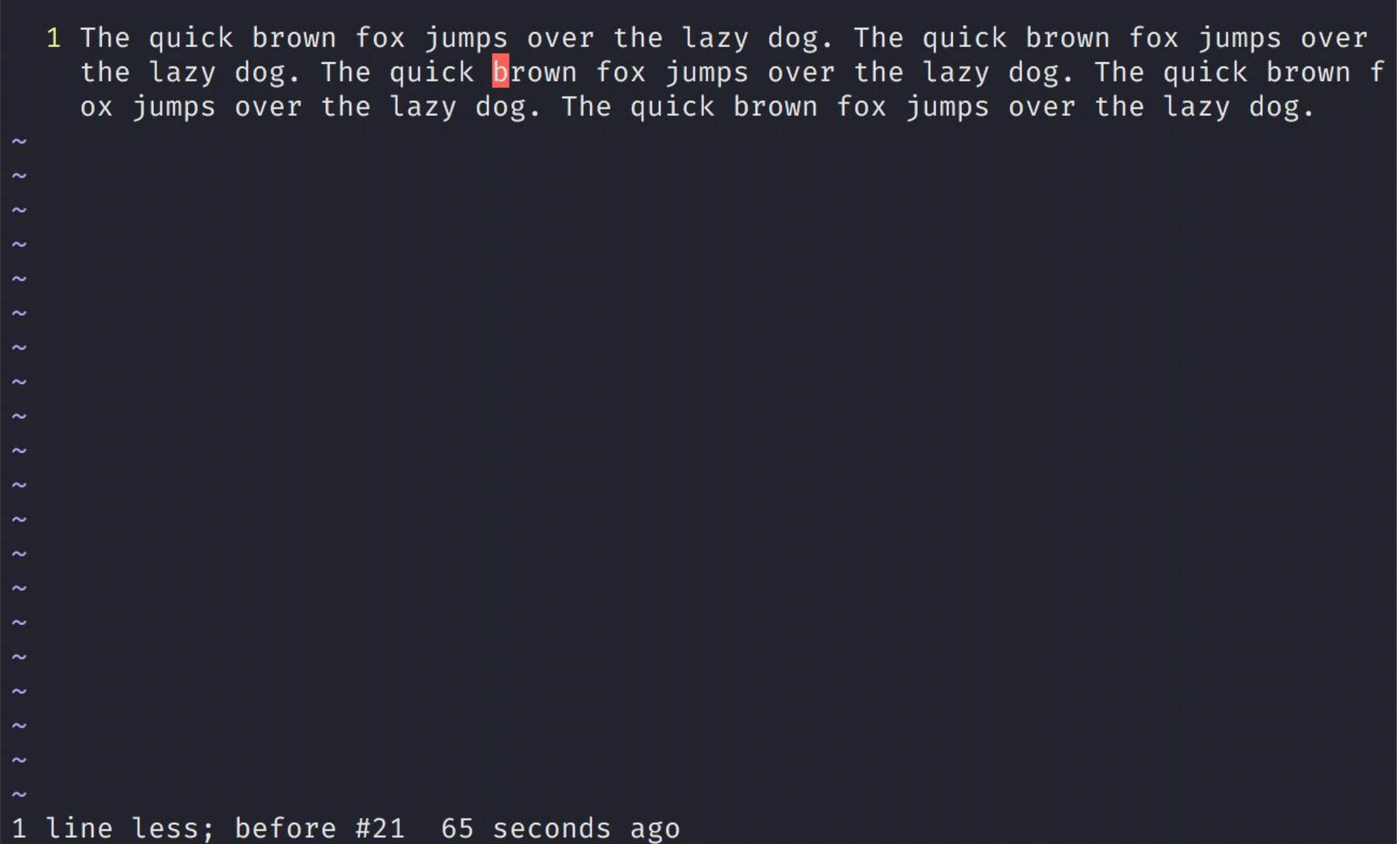
```
<ul>
        <li>Apple</li>
        <li>Banana</li>
        <li>Orange</li>
</ul>
```

```
30 fewer lines; before #31  14:37:58
```

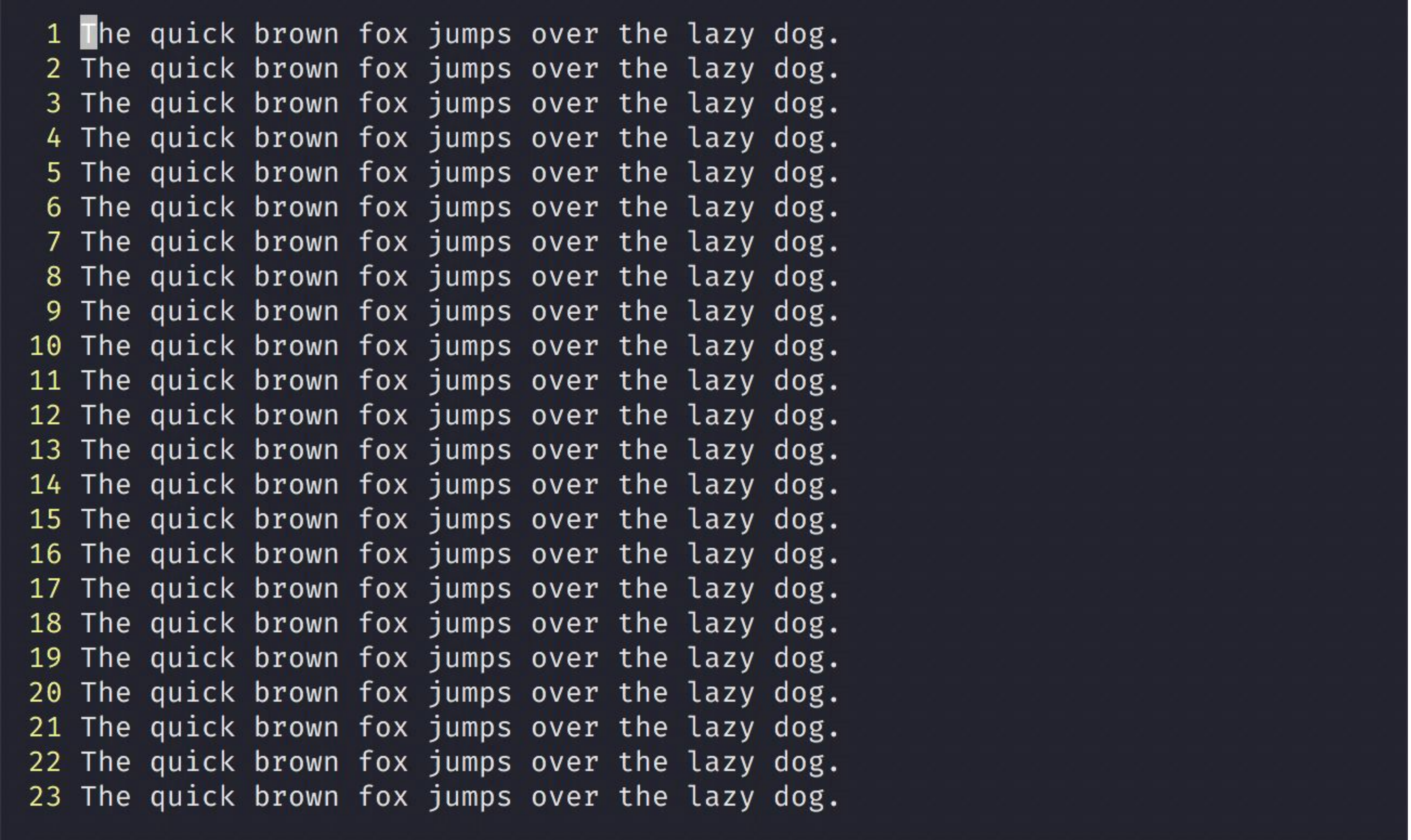
Useful Navigation

Navigating

`f + <char>`	Inline search next for `<char>` with pointer at `<char>`
`F + <char>`	Inline search previous for `<char>` with pointer at `<char>`
`t + <char>`	Inline search next for `<char>` with pointer before `<char>`
`T + <char>`	Inline search previous for `<char>` with pointer before `<char>`
`ctrl + f`	Scroll one page
`ctrl + b`	Scroll one page backward
`H`	Go to [line] from top window, default is first line
`M`	Go to middle line
`L`	Go to [line] from bottom window, default is last line



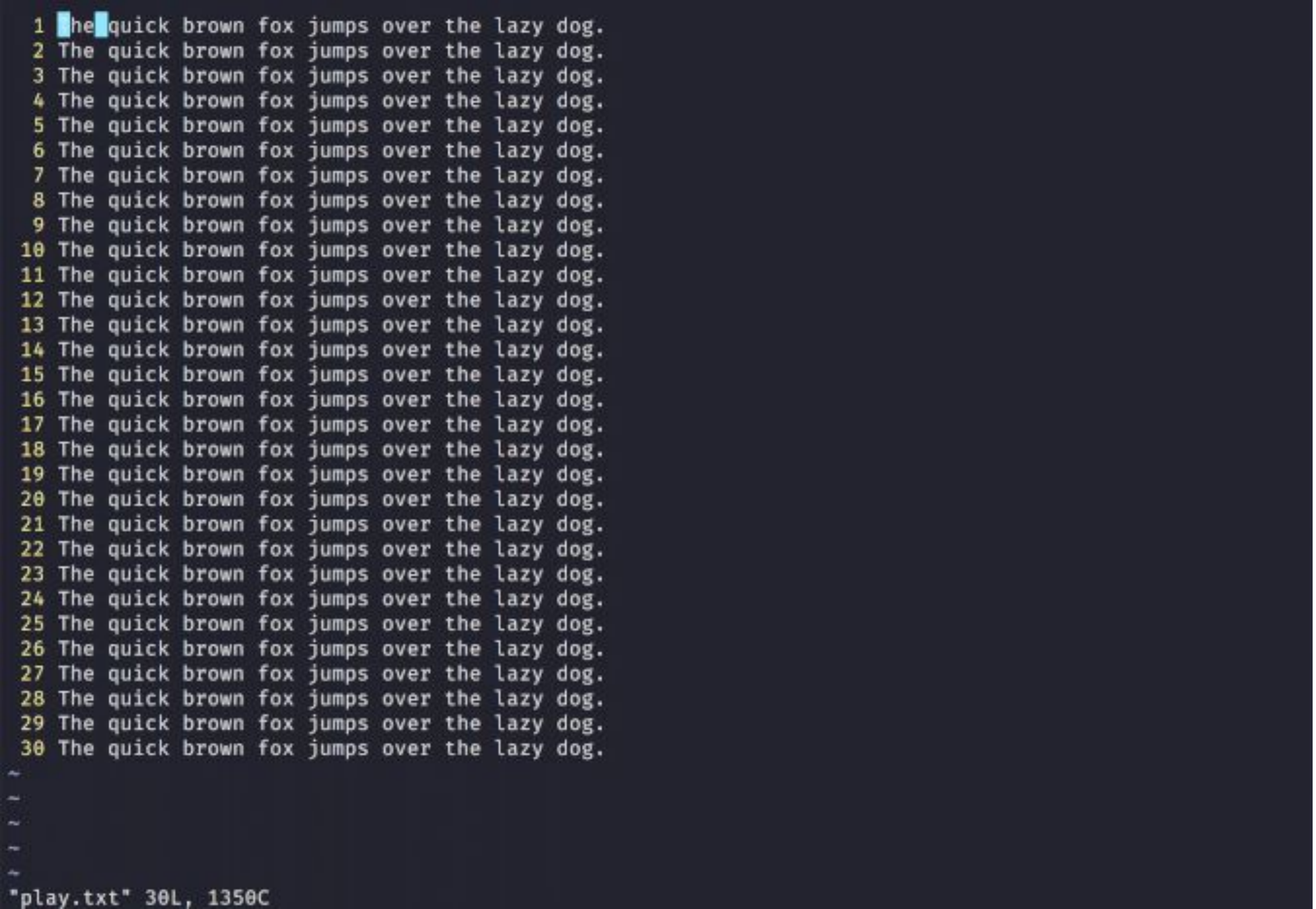
```
The quick brown fox jumps over the lazy dog. The quick brown fox jumps over
the lazy dog. The quick brown fox jumps over the lazy dog. The quick brown f
ox jumps over the lazy dog. The quick brown fox jumps over the lazy dog.
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
1 line less; before #21  65 seconds ago
```



```
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.
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.
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.
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.
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

ctrl + w _	Max height current split
ctrl + w |	Max width current split
ctrl + w =	Normalize split
ctrl + w o	Close all split except current split
ctrl + w T	Convert split to tab



```
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.
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.
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.
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.
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.
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.
~
~
~
~
~
"play.txt" 30L, 1350C
```

Netrw

What is it?

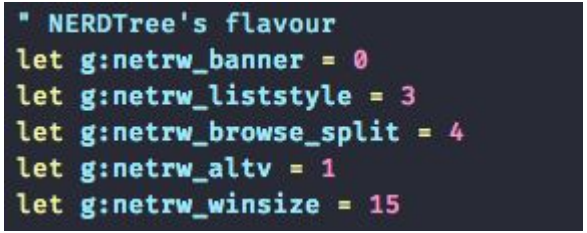
“NerdTree is a project drawer, netrw is a split explorer”

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

Commands

`:Explore`	Open netrw
`:Vexplore`	Open netrw in vertical split
`:Sexplore`	Open netrw in horizontal split

NERDTree's Flavour



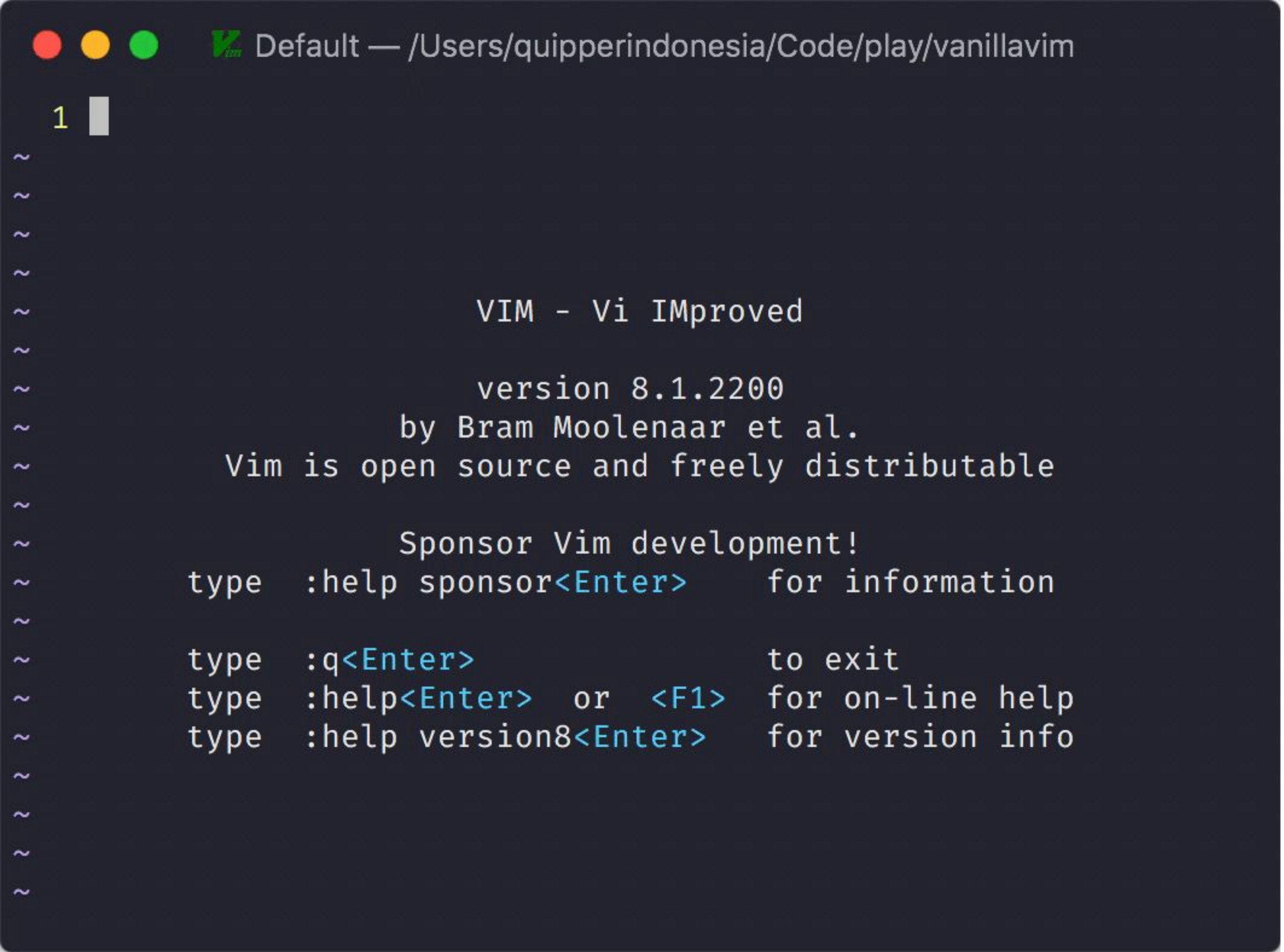
```
" NERDTree's flavour
let g:netrw_banner = 0
let g:netrw_liststyle = 3
let g:netrw_browse_split = 4
let g:netrw_altv = 1
let g:netrw_winsize = 15
```

```
~/Code/play/vanillavim                              ruby-2.4.1[2:01:45]
```

File Search

Recursive Completer

```
:set path+=**
:set wildmode=longest:full,full
:set wildignore+=**/node_modules/**,**/public/**
```

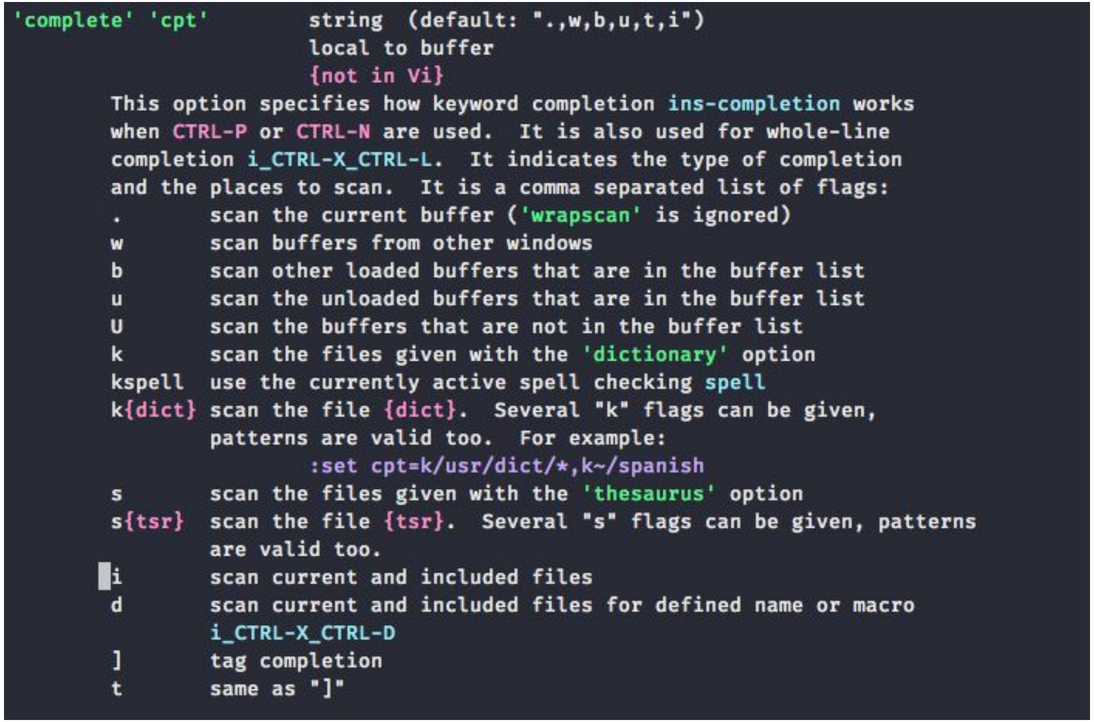


Default — /Users/quipperindonesia/Code/play/vanillavim

```
  1 
~
~
~
~
~                        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

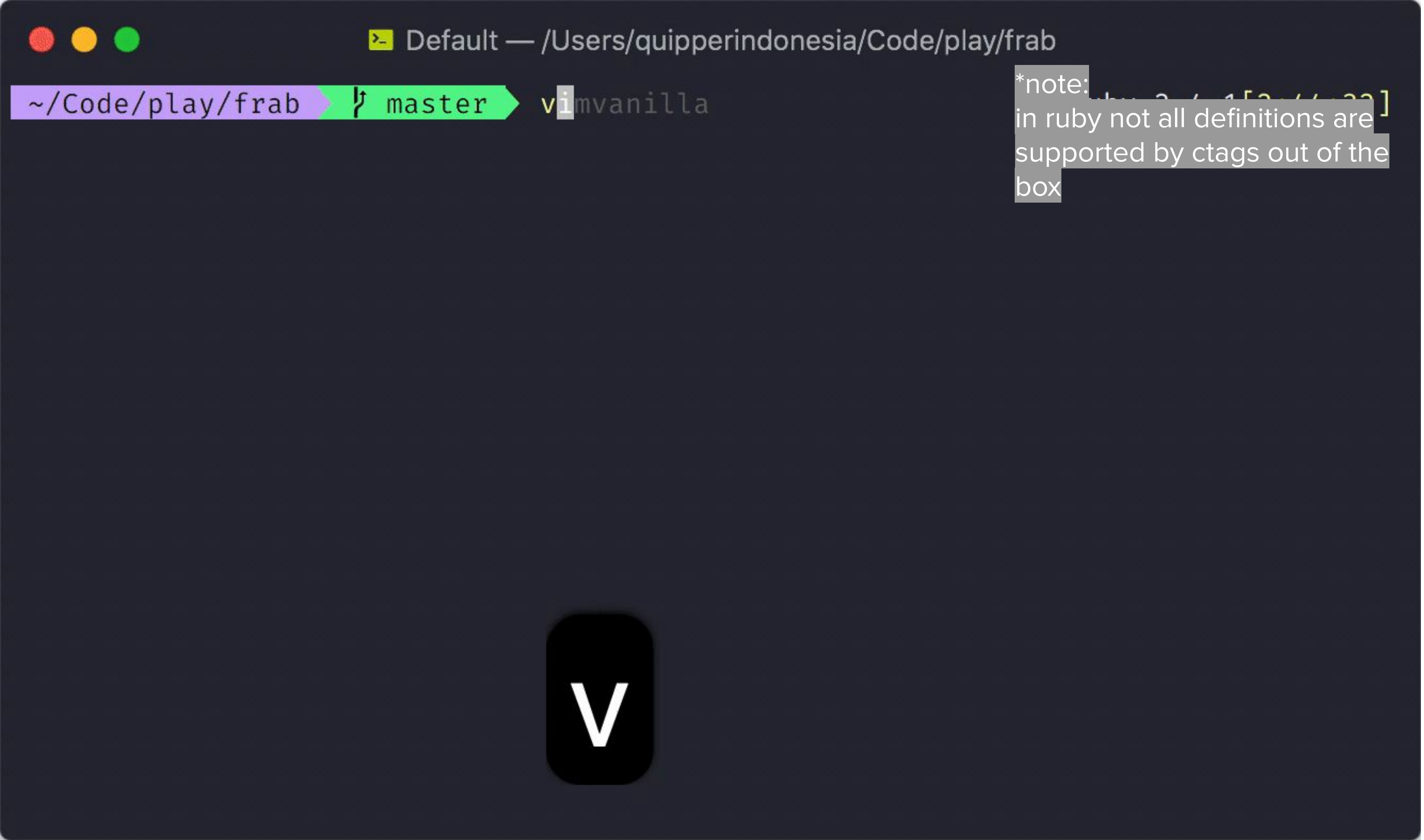


```
'complete' 'cpt'        string  (default: ".,w,b,u,t,i")
                        local to buffer
                        {not in Vi}
        This option specifies how keyword completion ins-completion works
        when CTRL-P or CTRL-N are used.  It is also used for whole-line
        completion 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:
                        :set cpt=k/usr/dict/*,k~/spanish
        s       scan the files given with the 'thesaurus' option
        s{tsr}  scan the file {tsr}.  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

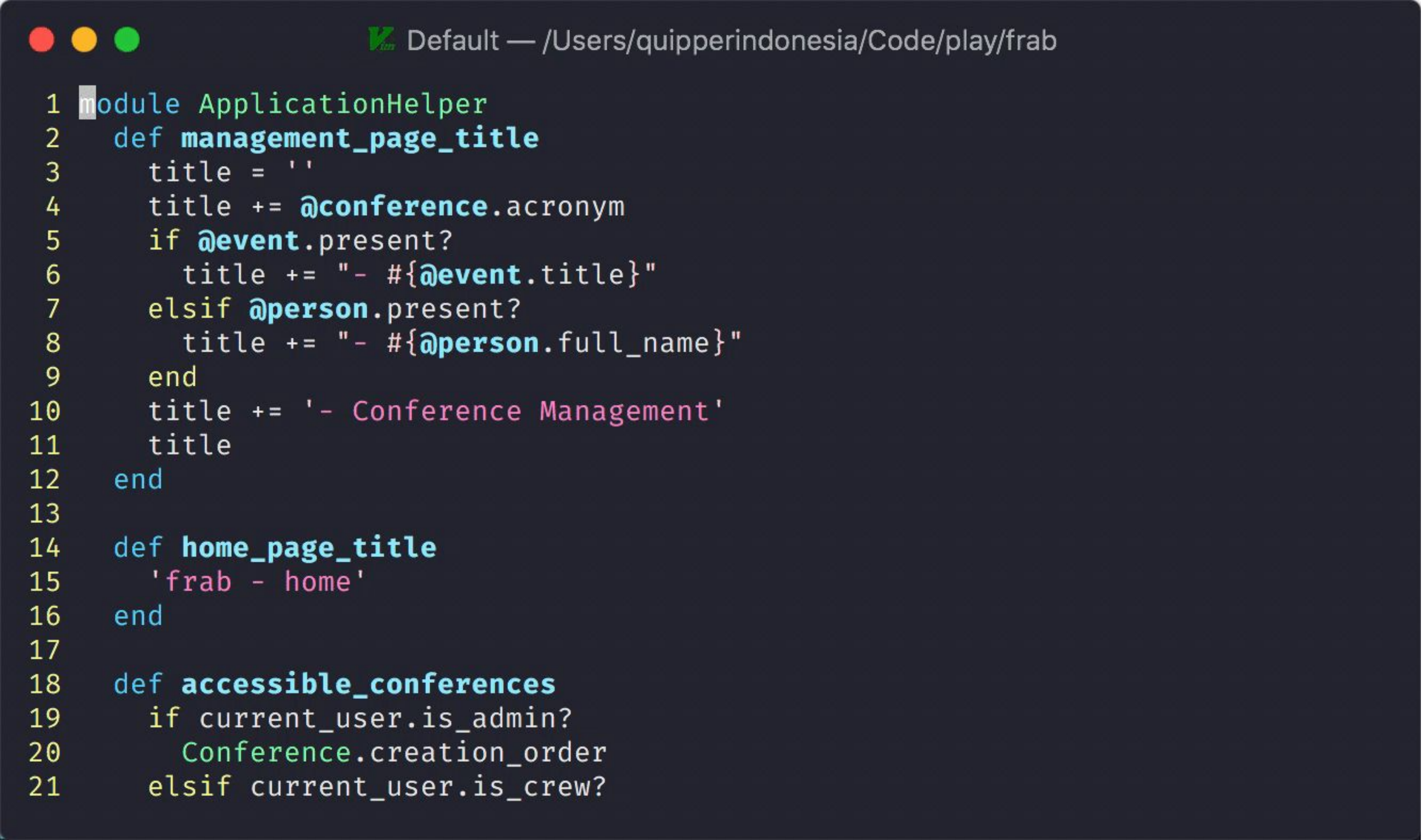
```
brew install ctags
alias ctags="`brew --prefix`/bin/ctags"
# ruby example
ctags -R --languages=ruby --exclude=.git --exclude=log .
```

`Ctrl + ]`	Open method definition
`ctrl + w ]`	Open method definition in horizontal split
`ctrl + t`	Back to method call



```
Default — /Users/quipperindonesia/Code/play/frab

~/Code/play/frab   master   vimvanilla
```

*note:
in ruby not all definitions are supported by ctags out of the box

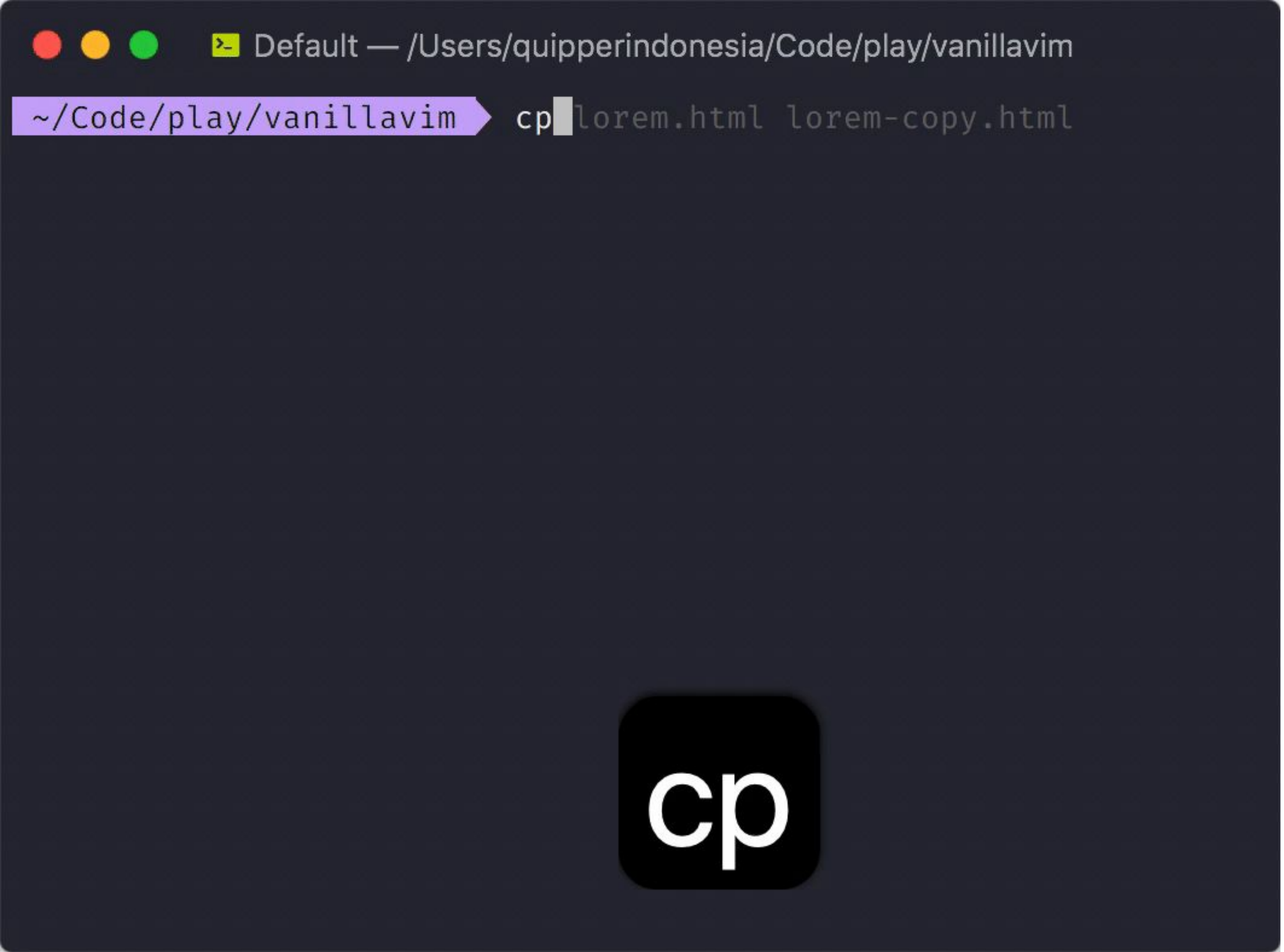


```
module ApplicationHelper
  def management_page_title
    title = ''
    title += @conference.acronym
    if @event.present?
      title += "- #{@event.title}"
    elsif @person.present?
      title += "- #{@person.full_name}"
    end
    title += '- Conference Management'
    title
  end

  def home_page_title
    'frab - home'
  end

  def accessible_conferences
    if current_user.is_admin?
      Conference.creation_order
    elsif current_user.is_crew?
```

Searching

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

Vim in its vanilla form is also awesome for productive works

Thank you