An Intro to Tmux and Vim Configs

A whirlwind tour through some things that I found helpful initially

Barrett Clark - @barrettclark

`https://github.com/barrettclark/intro-to-tmux-vim-configs.git`

Disclaimer: I have preferences that I will be using as the basis for this workshop.

My hope is that you feel generally comfortable configuring vim and tmux after this workshop and can start forming your own preferences.

Hopefully enough information to help you start customizing your shell environment to your liking.

I am not an expert on any of this

I am an EM at HashiCorp
and a member of Ruby Central

Workshop Flow

- Initial Setup
- Vim - Step 0: How To Even
- Vim - Step 1: Some Very Basic Settings
- Tmux - Step 1: Some Very Basic Settings
- Tmux - Step 2: Session Management
- Tmux - Step 3: Plugins
- Vim - Step 2: Plugins
- Vim - Step 3: Powerline, But It's Airline
- Tmux - Step 4: Powerline-ish
- Vim - Step 4: NERDTree (Bonus Content)
- Dotfiles (Bonus Content)

Initial Setup

Install Things

Step 1 - Clone Companion Repo

```
git clone https://github.com/barrettclark/intro-to-tmux-vim-configs.git
```

Step 2 - Install Things

```
# Using Homebrew
brew bundle --verbose

# Linux (apt)
apt-get install curl fonts-powerline git powerline sudo tmux tmuxp vim wget
```

Vim - Step 0

How To Even

Common Keystrokes and Commands

```
r - edit single character
R - enter edit (replace) mode
i - enter edit (insert) mode
i - enter edit (insert) mode
x - delete character
. - repeat last edit
~ - change case

cw - change word, from the cursor
cW - change line, from the cursor

/ - search
n - find next match
```

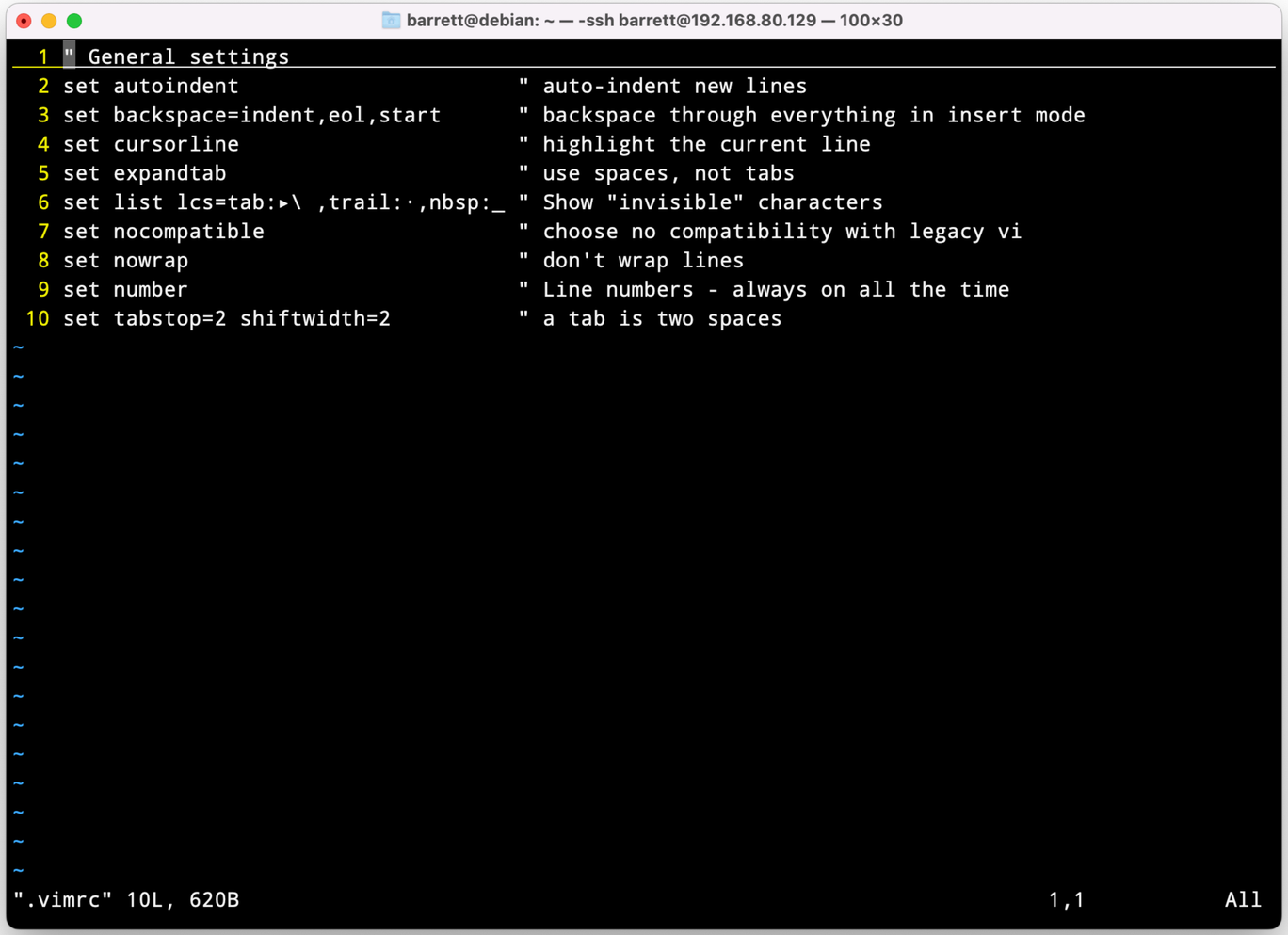
```
h - left arrow
j - down arrow
k - up arrow
l - right arrow
w - move forward one word/punctuation at a time
b - move back one word/punctuation at a time
V - visually highlight line(s)

: - enter command mode
:q - quit
:q! - quit and don't save changes
:wq - quit and save changes (write, quit)
:wq! - more forceful quit and
        save changes (write, quit)
```

Vim - Step 1

Some Very Basic Settings

~/.vimrc

```
" General settings
set autoindent                          " auto-indent new lines
set backspace=indent,eol,start          " backspace through everything in insert mode
set cursorline                          " highlight the current line
set expandtab                           " use spaces, not tabs
set list lcs=tab:▸\ ,trail:·,nbsp:_ " Show "invisible" characters
set nocompatible                        " choose no compatibility with legacy vi
set nowrap                              " don't wrap lines
set number                              " Line numbers - always on all the time
set tabstop=2 shiftwidth=2              " a tab is two spaces
```


barrett@debian: ~ — -ssh barrett@192.168.80.129 — 100×30
" General settings
set autoindent " auto-indent new lines
set backspace=indent,eol,start " backspace through everything in insert mode
set cursorline " highlight the current line
set expandtab " use spaces, not tabs
set list lcs=tab:▸\ ,trail:·,nbsp:_ " Show "invisible" characters
set nocompatible " choose no compatibility with legacy vi
set nowrap " don't wrap lines
set number " Line numbers - always on all the time
set tabstop=2 shiftwidth=2 " a tab is two spaces
".vimrc" 10L, 620B
1,1
All


barrett@debian: ~ — -ssh barrett@192.168.80.129 — 100×30
1 class Greeter
2 def initialize(name = 'World')
3 @name = name
4 end
5
6 def say_hi
7 puts "Hi #{@name}!"
8 end
9
10 def say_bye
11 puts "Bye #{@name}, come back soon."
12 end
13 end
"greeter.rb" 13L, 180B

Tmux - Step 1

Some Very Basic Settings

"Tmux is the real power tool for the command line."

Barrett Clark (me)

Tmux lets you switch easily between several programs in one terminal, detach them (they keep running in the background) and reattach them to a different terminal.

Tmux Objects Overview

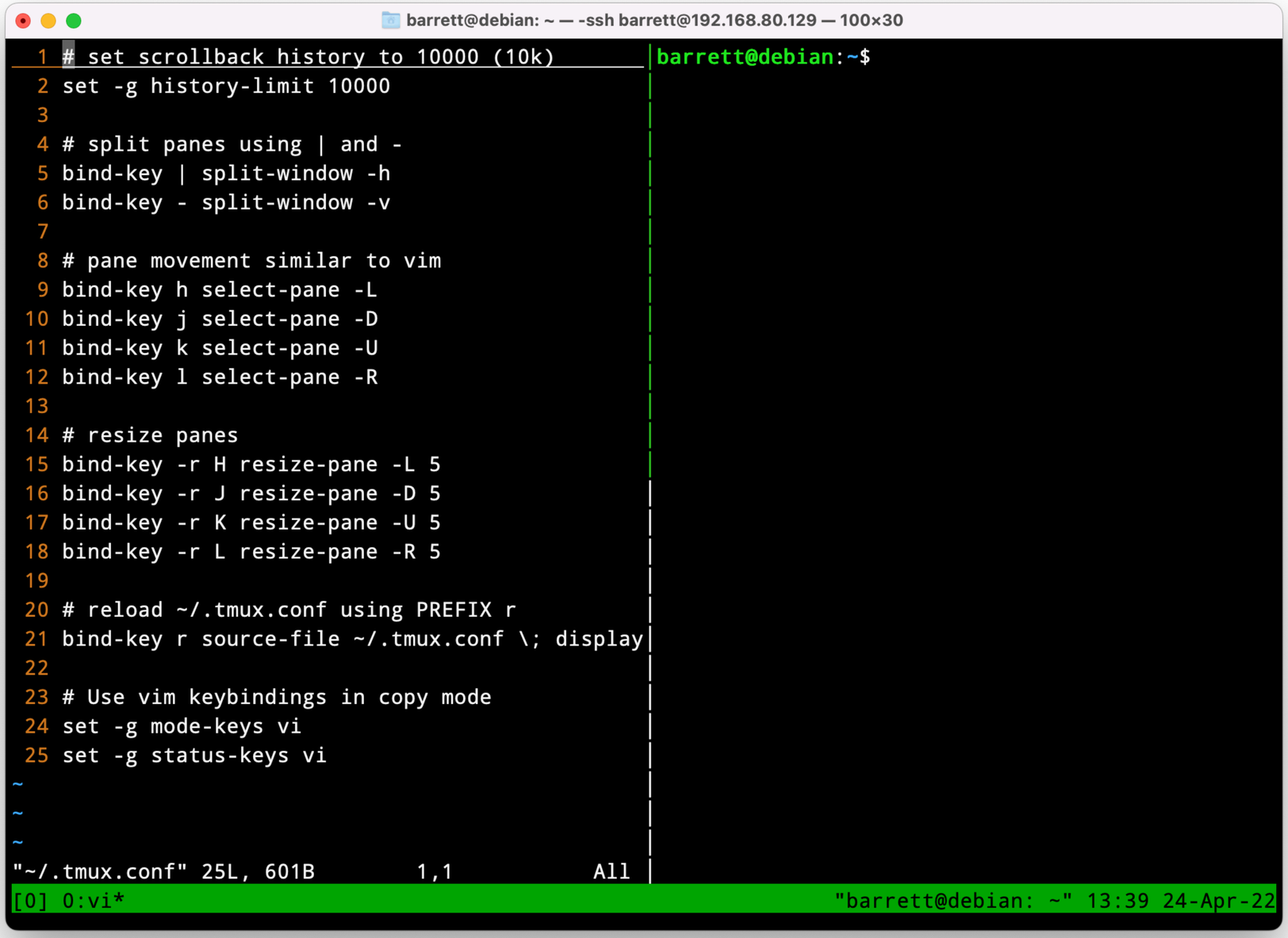
- Session
  - A named collection of one or more windows
- Window
  - A single screen within tmux
- Pane
  - A portion of a window running a single process
- Status Bar
  - Information on the bottom of the tmux session screen

~/.tmux.conf

```
# set scrollback history to 10000 (10k)
set -g history-limit 10000

# split panes using | and -
bind-key | split-window -h
bind-key - split-window -v

# pane movement similar to vim
bind-key h select-pane -L
bind-key j select-pane -D
bind-key k select-pane -U
bind-key l select-pane -R

# resize panes
bind-key -r H resize-pane -L 5
bind-key -r J resize-pane -D 5
bind-key -r K resize-pane -U 5
bind-key -r L resize-pane -R 5

# reload ~/.tmux.conf using PREFIX r
bind-key r source-file ~/.tmux.conf \; display "Reloaded!"

# Use vim keybindings in copy mode
set -g mode-keys vi
set -g status-keys vi

# Enable mouse mode (select/resize panes)
set -g mouse on
```

The prefix key

```
Ctrl + b
```


barrett@debian: ~ — -ssh barrett@192.168.80.129 — 100×30
1 # set scrollback history to 10000 (10k)
2 set -g history-limit 10000
3
4 # split panes using | and -
5 bind-key | split-window -h
6 bind-key - split-window -v
7
8 # pane movement similar to vim
9 bind-key h select-pane -L
10 bind-key j select-pane -D
11 bind-key k select-pane -U
12 bind-key l select-pane -R
13
14 # resize panes
15 bind-key -r H resize-pane -L 5
16 bind-key -r J resize-pane -D 5
17 bind-key -r K resize-pane -U 5
18 bind-key -r L resize-pane -R 5
19
20 # reload ~/.tmux.conf using PREFIX r
21 bind-key r source-file ~/.tmux.conf \; display
22
23 # Use vim keybindings in copy mode
24 set -g mode-keys vi
25 set -g status-keys vi
~
~
~
"~/.tmux.conf" 25L, 601B 1,1 All
barrett@debian:~$
[0] 0:vi*
"barrett@debian: ~" 13:39 24-Apr-22

Turtles all the way down

They're shells. Get it...?

- Each window and pane is a subshell
- Typing exit will close that pane
- If there is only 1 pane that will close the window
- If there is only 1 window that will exit that session

Tmux Practice Time

- Create your first session
  - `tmux new -s first-session`
- Create a pane
  - `Ctrl + b + |`
- Navigate between the panes
  - `Ctrl + b + l` (move right)
  - `Ctrl + b + h` (move left)
- Create a new window
  - `Ctrl + b + c`
- Navigate between windows
  - `Ctrl + b + n`
- Rename the session
  - `Ctrl + b + $`

Tmux - Step 2

Session Management

Basic Session Management

```
tmux new -s session-name                # create a new session
prefix+d                                # detach from session
tmux attach-session -t session-name     # reconnect to existing session
```

Let's create a session called session1, detach from that session, then create another session called session2

```
prefix+s
```

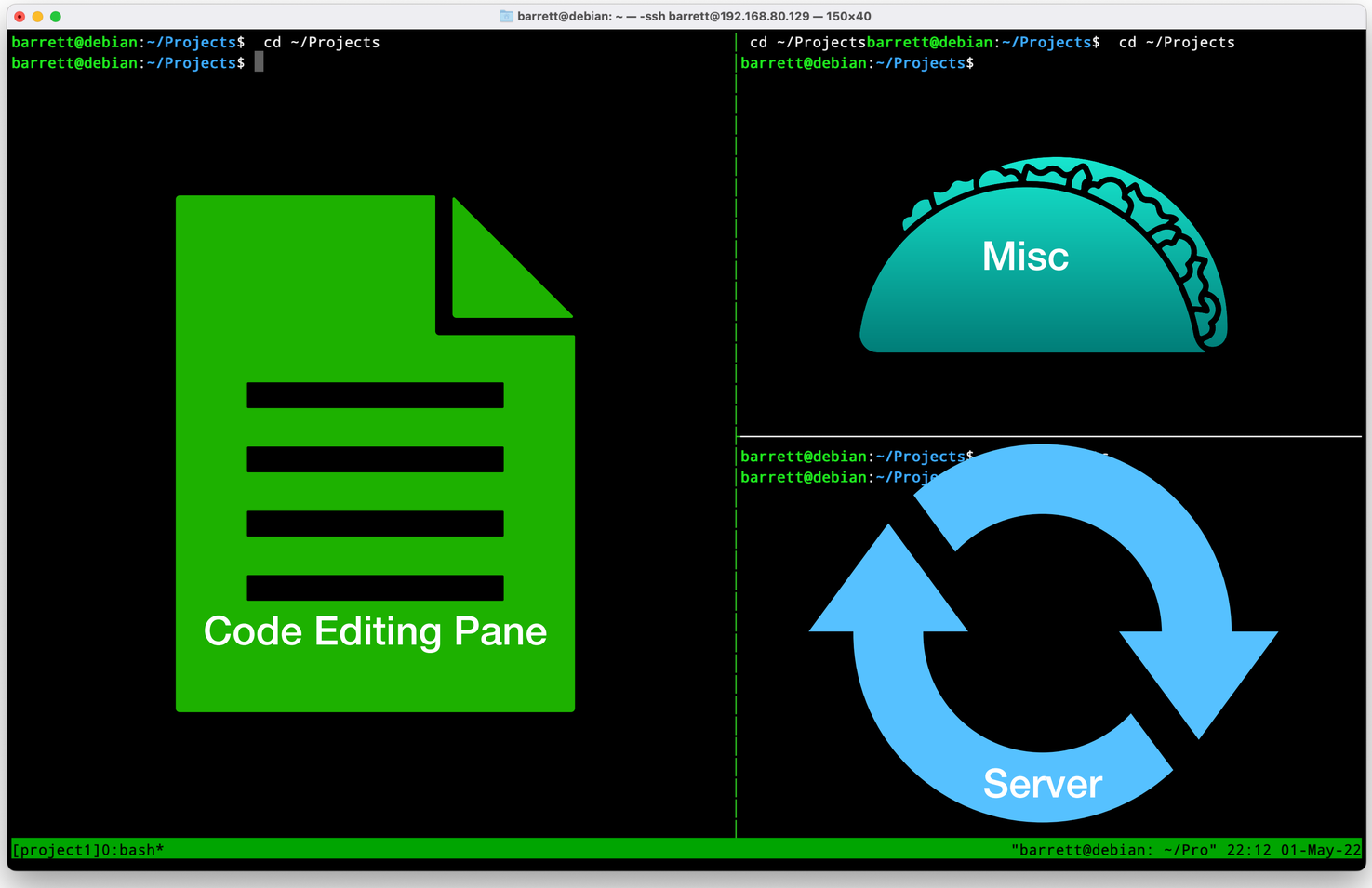
tmuxp

We can use a tool called tmuxp to create new sessions from a template.
Each project can have it's own template, that includes things like what directory be in for each pane.

https://github.com/tmux-python/tmuxp

~/.tmuxp/default.yaml

```
windows:
  - layout: main-vertical
    focus: 'true'
    options:
      main-pane-width: 55%
      automatic-rename: 'off'
    shell_command_before:
      - cd ~/Projects
    panes:
      - focus: true
      - blank
      - blank
```



tmux-up.sh

```
#!/bin/bash

if [ $# -gt 0 ]; then
  session=$1
  tmux has-session -t $session 2>/dev/null

  if [ $? != 0 ]; then
    if [ -f ~/.tmuxp/$session.yaml ]; then
      tmuxp load $session
    else
      tmuxp load -s $session default
    fi
  else
    tmux attach-session -t $session
  fi
else
  echo " *** Try again with a session name"
fi
```

Now you can create and attach to a session using tmux-up.sh

```
./tmux-up.sh awesome-sauce
```

Tmux - Step 3

Plugins

Tmux Plugin Manager

https://github.com/tmux-plugins/tpm

Step 1 - Install tpm

```
git clone https://github.com/tmux-plugins/tpm ~/.tmux/plugins/tpm
```

Make sure these directories exist

Step 2 - Very Basic Config

```
# List of plugins
set -g @plugin 'tmux-plugins/tpm'

# Initialize TMUX plugin manager (keep this line at the very bottom of tmux.conf)
run '~/.tmux/plugins/tpm/tpm'
```

Step 3 - Reload tmux

```
prefix + r
```

Step 4 - Install Plugins

```
prefix + I
```

tpm plugins

```
# List of plugins
set -g @plugin 'tmux-plugins/tmux-battery'   # display battery status in tmux status-right
set -g @plugin 'tmux-plugins/tmux-continuum' # save tmux environment
set -g @plugin 'tmux-plugins/tmux-resurrect' # restore saved tmux environment
set -g @plugin 'tmux-plugins/tmux-yank'      # copy to the system clipboard in tmux
set -g @plugin 'tmux-plugins/tpm'            # tmux plugin manager

# prefix + Ctrl-s to save session
set -g @continuum-restore 'on'
set -g @continuum-boot 'on'
set -g @continuum-boot-options 'fullscreen'
set -g @shell_mode 'vi'
set -g @continuum-save-interval '60'
set -g @resurrect-save 'S'
set -g @resurrect-strategy-vim 'session'
set -g @resurrect-capture-pane-contents 'on'

# Initialize TMUX plugin manager (keep this line at the very bottom of tmux.conf)
run '~/.tmux/plugins/tpm/tpm'
```

Vim - Step 2

Plugins

vim-plug

A Vim Plugin Manager

https://github.com/junegunn/vim-plug

"It's really easy to go overboard with plugins and customization in vim. Keep it simple and constrained to what you can actually remember."

Barrett Clark (still me)

Vim Leader Key

In vim the default leader (prefix) key is the backslash.

Step 1 - Install vim-plug

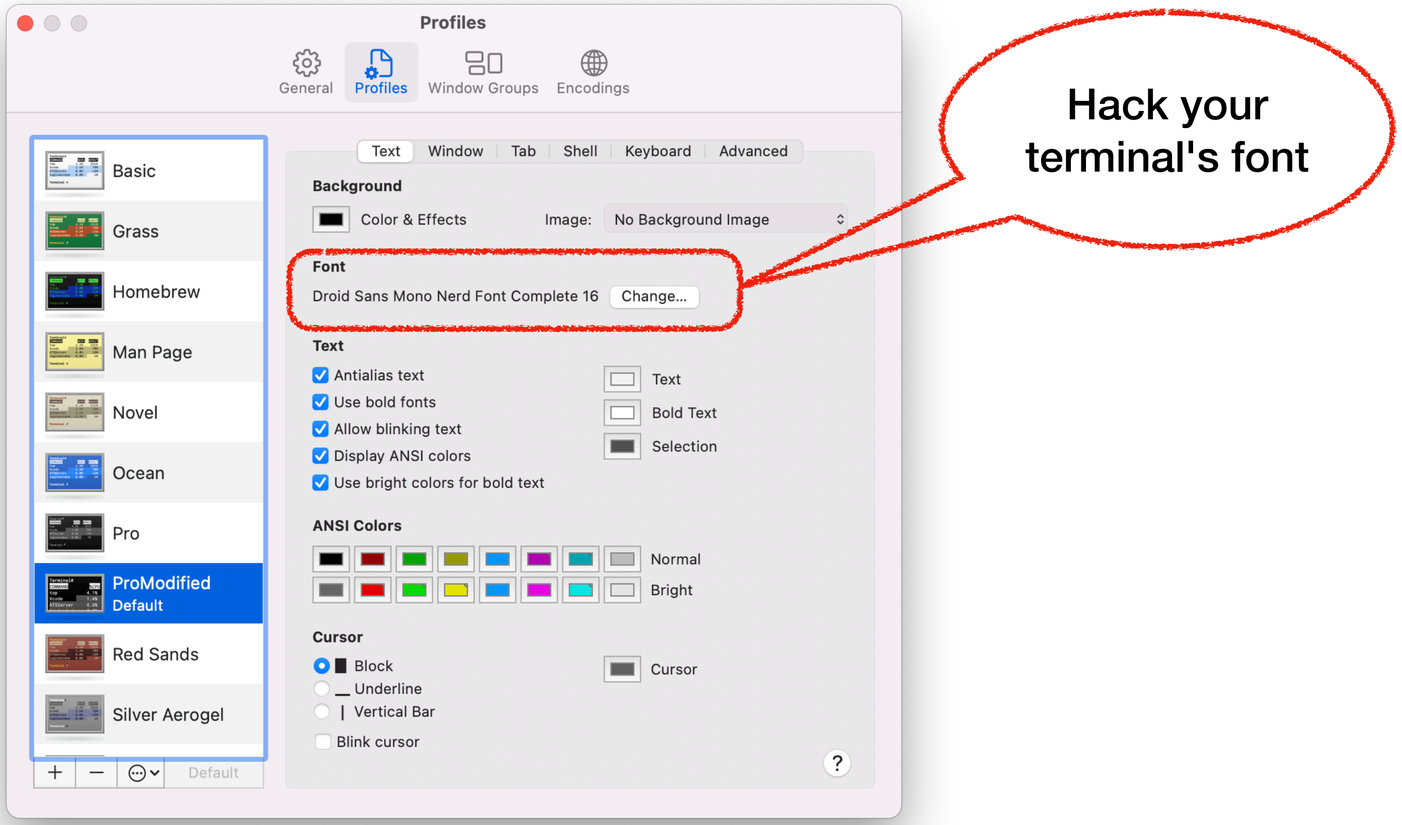
```
curl -fLo ~/.vim/autoload/plug.vim --create-dirs \
    https://raw.githubusercontent.com/junegunn/vim-plug/master/plug.vim
```

Step 2 - update .vimrc

```
" Vim plugins
" Brief help
" :PlugInstall    - installs plugins; append `!` to update or just :PlugUpdate
" :PlugUpdate     - update plugins
" :PlugClean      - confirms removal of unused plugins; append `!` to auto-approve removal
" :PlugUpgrade    - upgrade vim-plug itself
let data_dir = has('nvim') ? stdpath('data') . '/site' : '~/.vim'
if empty(glob(data_dir . '/autoload/plug.vim'))
  silent execute '!curl -fLo '.data_dir.'/autoload/plug.vim --create-dirs  https://raw.githubusercontent.com/junegunn/vim-plug/master/plug.vim'
  autocmd VimEnter * PlugInstall --sync | source $MYVIMRC
endif

call plug#begin('~/.vim/plugged')
" Utilities
Plug 'jiangmiao/auto-pairs'     " Insert or delete brackets, parens, quotes in pair
Plug 'mileszs/ack.vim'          " :Ack in vim
Plug 'tomtom/tcomment_vim'      " comment lines with <Leader>__ (and other cool tricks)
Plug 'tpope/vim-endwise'        " end structures automatically
Plug 'https://github.com/tpope/vim-dispatch.git'  " run commands in vim asynch

" Git utilities
Plug 'airblade/vim-gitgutter'   " show git status in the gutter
Plug 'tpope/vim-fugitive'       " vim Git wrapper

" Ruby
Plug 'tpope/vim-bundler'
Plug 'tpope/vim-rails'
Plug 'vim-ruby/vim-ruby'
call plug#end()

" Show commits for every source line
nnoremap <Leader>gb :Git blame<CR>
```

Vim - Step 3

Powerline, but it's Airline

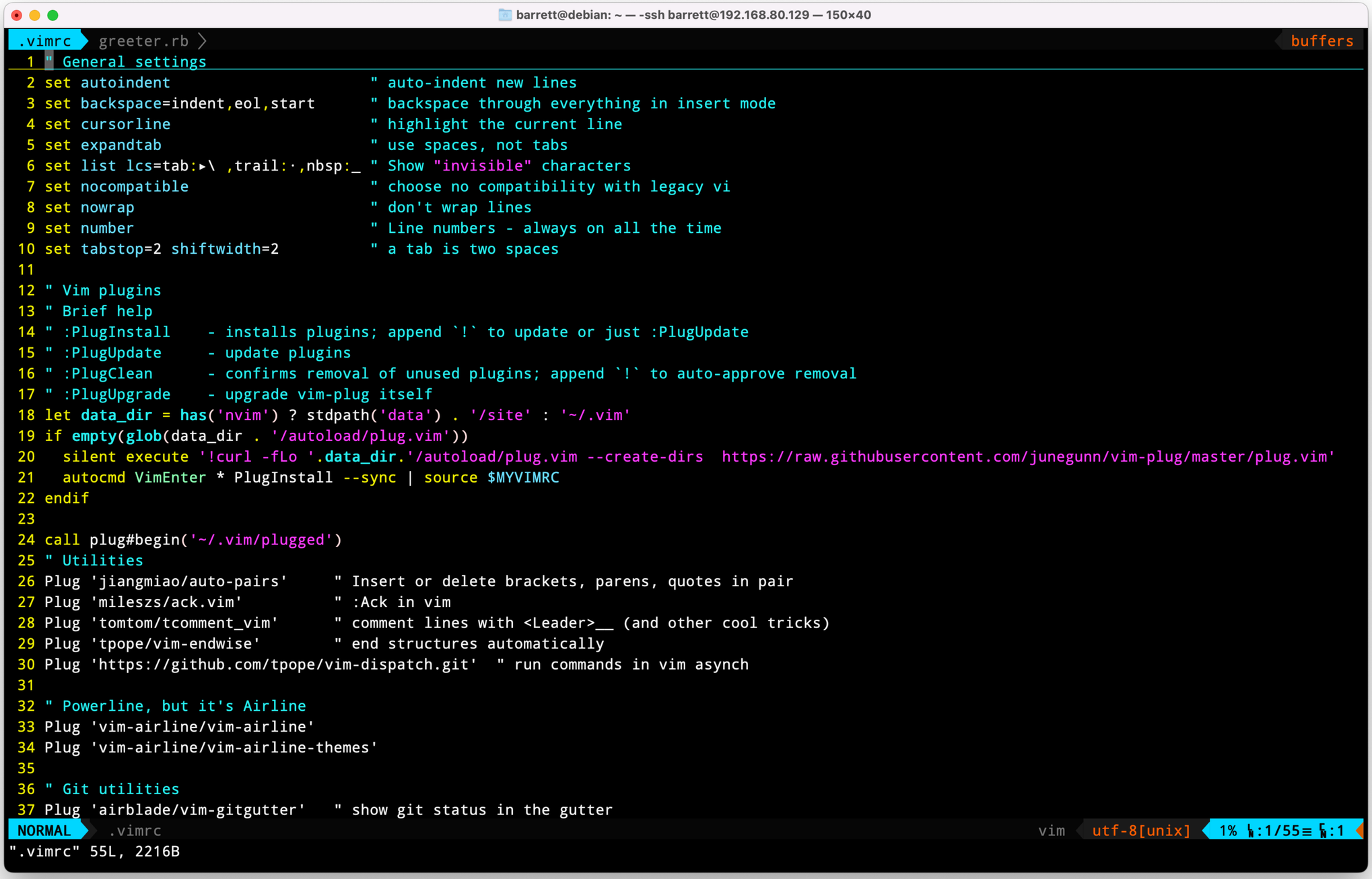

Profiles
General
Profiles
Window Groups
Encodings
Basic
Grass
Homebrew
Man Page
Novel
Ocean
Pro
ProModified
Default
Red Sands
Silver Aerogel
Default
Text
Window
Tab
Shell
Keyboard
Advanced
Background
Color & Effects
Image:
No Background Image
Font
Droid Sans Mono Nerd Font Complete 16
Change...
Text
Antialias text
Use bold fonts
Allow blinking text
Display ANSI colors
Use bright colors for bold text
Text
Bold Text
Selection
ANSI Colors
Normal
Bright
Cursor
Block
Underline
Vertical Bar
Blink cursor
Cursor
Hack your terminal's font

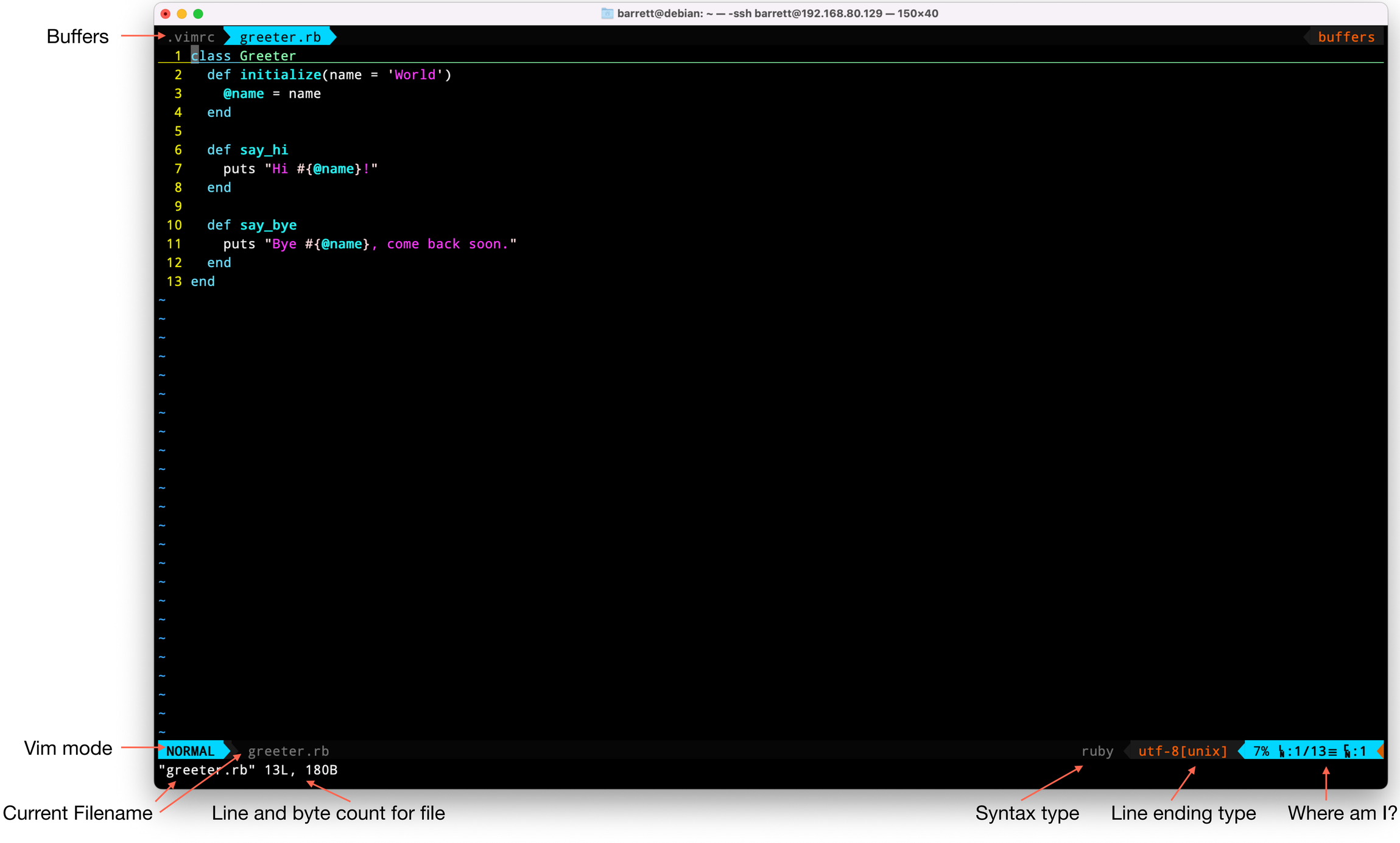
Step 1 - Add these plugins

```
" Color scheme for vim
Plug 'junegunn/seoul256.vim'

" Powerline, but it's Airline
Plug 'vim-airline/vim-airline'
Plug 'vim-airline/vim-airline-themes'
```

Step 2 - Add this config

```
" Color scheme settings
let g:seoul256_background = 234
colorscheme seoul256

" Vim Airline
set laststatus=2
let g:airline#extensions#tabline#enabled = 1
let g:airline_powerline_fonts = 1
let g:airline_theme='simple'
let g:airline#extensions#bufferline#enabled = 0
```


barrett@debian: ~ — -ssh barrett@192.168.80.129 — 150×40
.vimrc greeter.rb
buffers
1 " General settings
2 set autoindent " auto-indent new lines
3 set backspace=indent,eol,start " backspace through everything in insert mode
4 set cursorline " highlight the current line
5 set expandtab " use spaces, not tabs
6 set list lcs=tab:▸\ ,trail:·,nbsp:_ " Show "invisible" characters
7 set nocompatible " choose no compatibility with legacy vi
8 set nowrap " don't wrap lines
9 set number " Line numbers - always on all the time
10 set tabstop=2 shiftwidth=2 " a tab is two spaces
11
12 " Vim plugins
13 " Brief help
14 " :PlugInstall - installs plugins; append `!` to update or just :PlugUpdate
15 " :PlugUpdate - update plugins
16 " :PlugClean - confirms removal of unused plugins; append `!` to auto-approve removal
17 " :PlugUpgrade - upgrade vim-plug itself
18 let data_dir = has('nvim') ? stdpath('data') . '/site' : '~/.vim'
19 if empty(glob(data_dir . '/autoload/plug.vim'))
20 silent execute '!curl -fLo '.data_dir.'/autoload/plug.vim --create-dirs https://raw.githubusercontent.com/junegunn/vim-plug/master/plug.vim'
21 autocmd VimEnter * PlugInstall --sync | source $MYVIMRC
22 endif
23
24 call plug#begin('~/.vim/plugged')
25 " Utilities
26 Plug 'jiangmiao/auto-pairs' " Insert or delete brackets, parens, quotes in pair
27 Plug 'mileszs/ack.vim' " :Ack in vim
28 Plug 'tomtom/tcomment_vim' " comment lines with <Leader>__ (and other cool tricks)
29 Plug 'tpope/vim-endwise' " end structures automatically
30 Plug 'https://github.com/tpope/vim-dispatch.git' " run commands in vim asynch
31
32 " Powerline, but it's Airline
33 Plug 'vim-airline/vim-airline'
34 Plug 'vim-airline/vim-airline-themes'
35
36 " Git utilities
37 Plug 'airblade/vim-gitgutter' " show git status in the gutter
NORMAL .vimrc vim utf-8[unix] 1% ℓ:1/55≡ ℅:1
".vimrc" 55L, 2216B


barrett@debian: ~ — -ssh barrett@192.168.80.129 — 150×40
Buffers
.vimrc
greeter.rb
buffers
1 class Greeter
2 def initialize(name = 'World')
3 @name = name
4 end
5
6 def say_hi
7 puts "Hi #{@name}!"
8 end
9
10 def say_bye
11 puts "Bye #{@name}, come back soon."
12 end
13 end
Vim mode
NORMAL
greeter.rb
ruby
utf-8[unix]
7%
"greeter.rb" 13L, 180B
Current Filename
Line and byte count for file
Syntax type
Line ending type
Where am I?

Tmux - Step 4

Powerline-ish

"Funny story. As I did the research for this workshop I realized that I am not actually using Powerline on my Mac. I'm using Powerline-patched fonts in the Tmux statusline."

Anonymous

NOTE: You probably don't need to do this

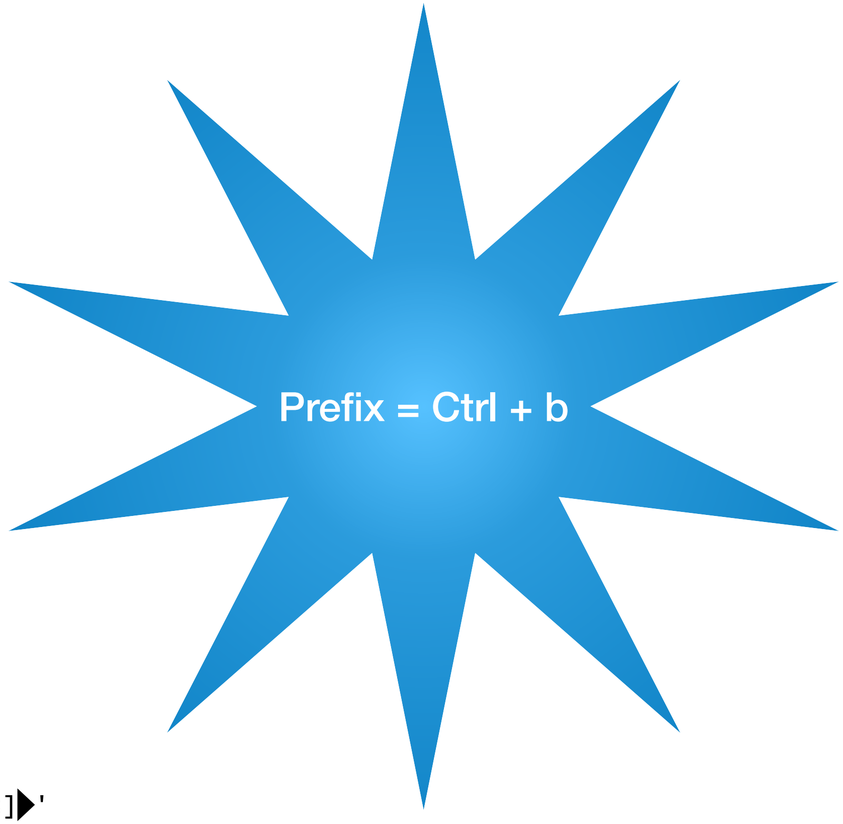
Step 1 - Actually Install Powerline

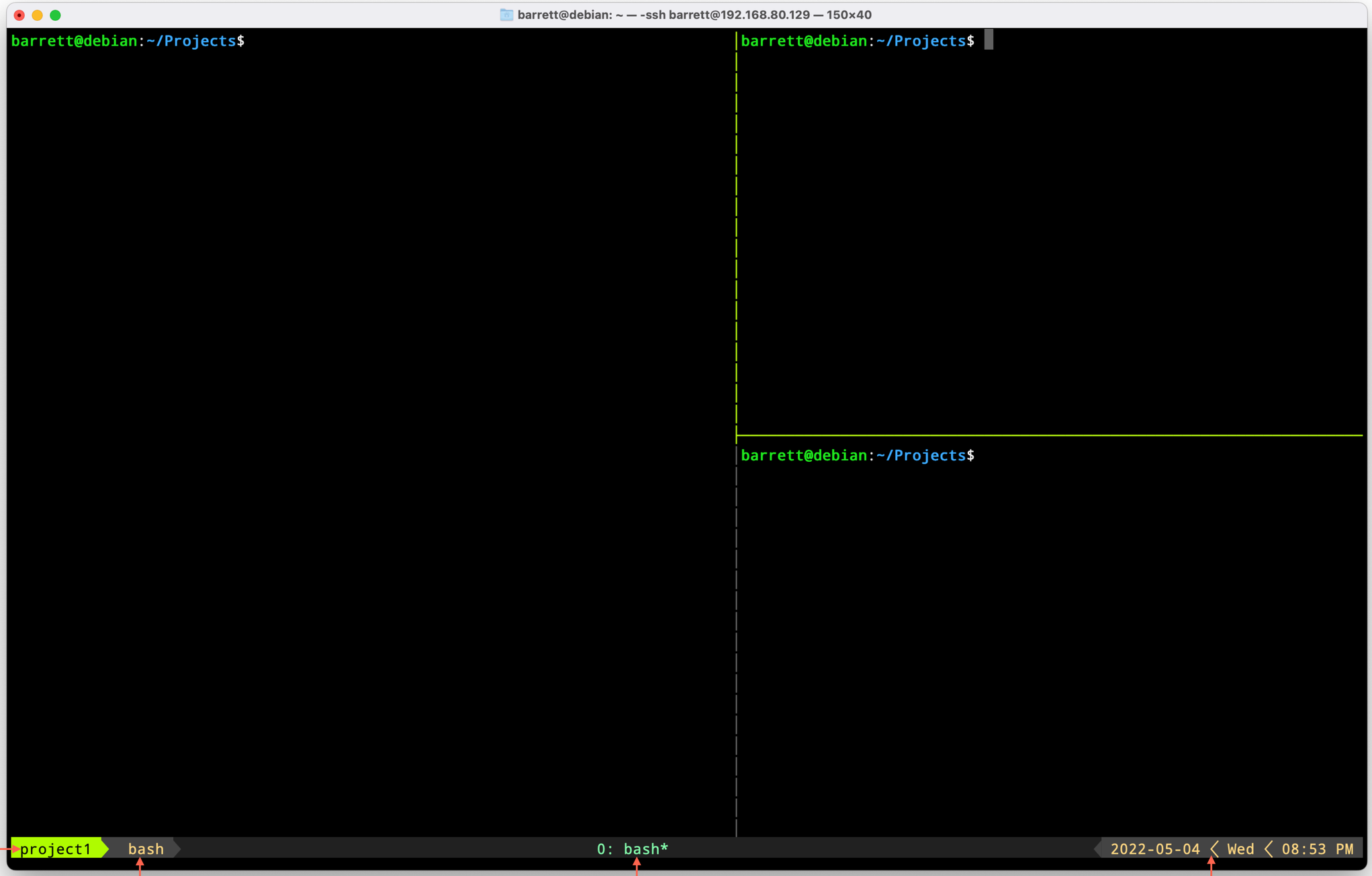
```
sudo pip3 install powerline-status
```

Step 2 - Setup Tmux for Powerline

```
powerline-config tmux setup
```

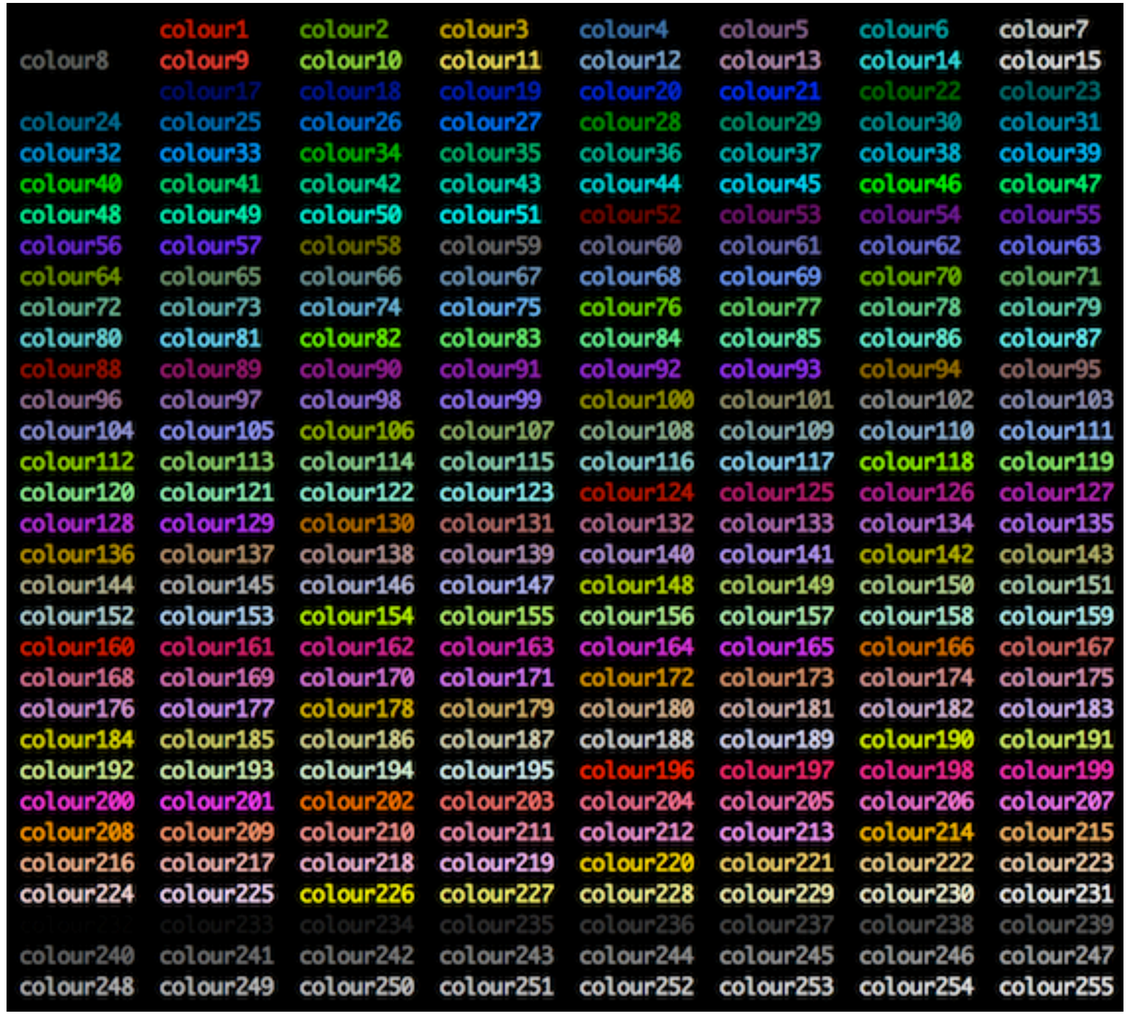
```
# tmux pane styling
set -g pane-active-border-style fg='colour154'
set -g pane-border-style fg='colour238'

# powerline status layout
set -g status 'on'
set -g status-justify 'centre'
set -g status-left-length '100'
set -g status-right-length '100'
set -g status-right-style 'none'
set -g status-style 'none'
set -g status-style bg='colour235','none'

# powerline colors
set -g message-style bg='colour238',fg='colour222'
set -g window-status-activity-style bg='colour235',fg='colour154','none'
set -g window-status-separator ''
set -g window-status-style bg='colour235',fg='colour121','none'

# status left
grey_left_triangle='#[fg=colour238,bg=colour235,nobold,nounderscore,noitalics]▶'
dark_grey_left_triangle='#[fg=colour235,bg=colour235,nobold,nounderscore,noitalics]▶'
status_session_name="#[fg=colour232,bg=colour154] #S g=colour154,bg=colour238,nobold,nounderscore,noitalics]▶"
status_process="#[fg=colour222,bg=colour238] #W $grey_left_triangle"
status_uptime="#[fg=colour121,bg=colour235] #(uptime  | cut -d \" \" -f 1,2,3) $dark_grey_left_triangle"
set -g status-left "$status_session_name $status_process"

# window status (middle, very simple without powerline characters)
set -g window-status-current-format "#I: #W#F"
set -g window-status-format "#I: #W"

# status right
grey_left_triangle='#[fg=colour238,bg=colour235,nobold,nounderscore,noitalics]◀'
status_datetime="$grey_left_triangle#[fg=colour222,bg=colour238] %Y-%m-%d < %a < %I:%M %p "
set -g status-right "$status_datetime"
```




barrett@debian: ~ — -ssh barrett@192.168.80.129 — 150×40
barrett@debian:~/Projects$
barrett@debian:~/Projects$
barrett@debian:~/Projects$
project1
bash
0: bash*
2022-05-04 〈 Wed 〈 08:53 PM
Session name
Current process
Windows
Current date and time

Tmux Colors

https://superuser.com/a/1104214

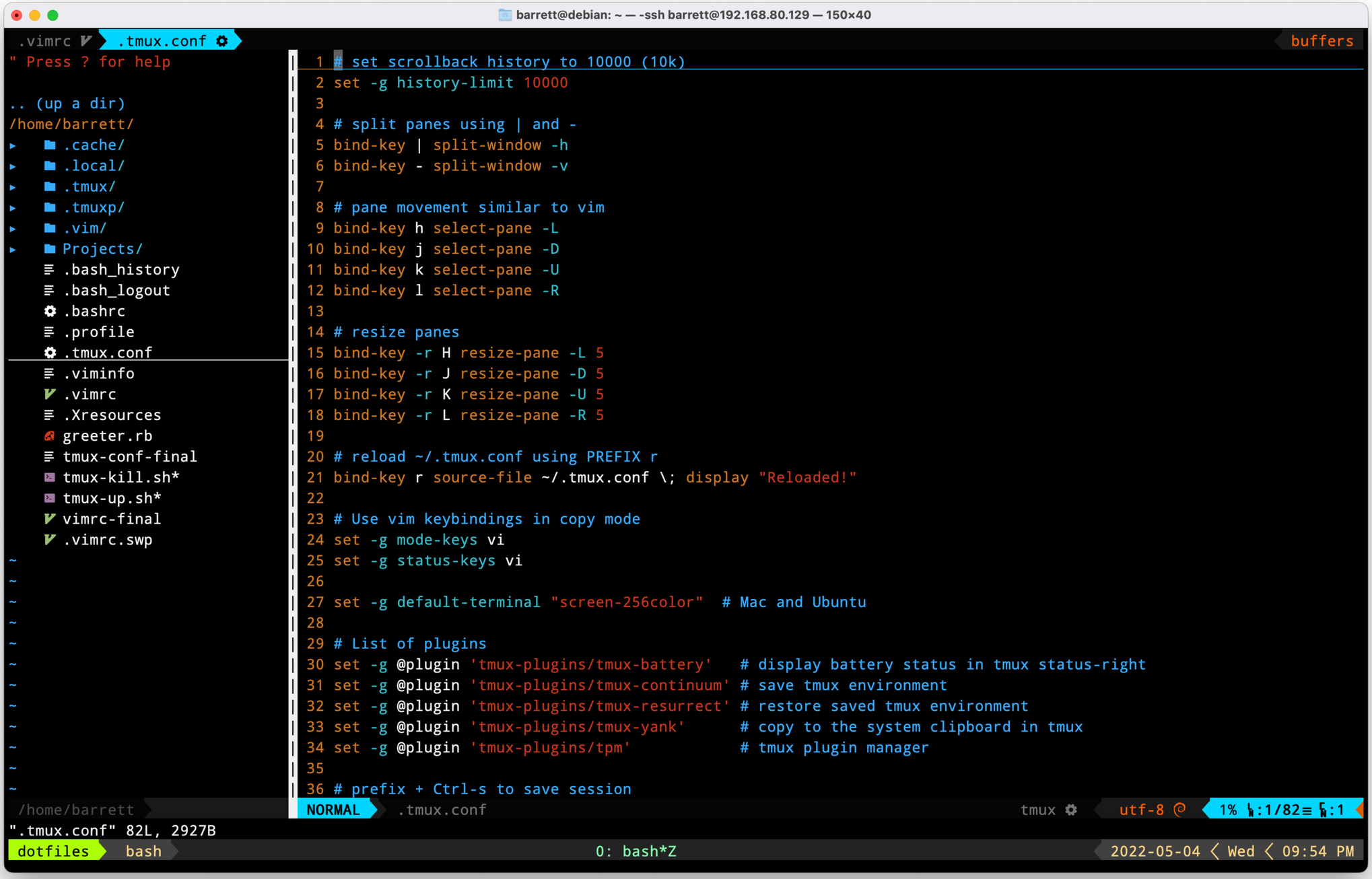


Vim - Step 4

Bonus Content - NERDTree

```
Plug 'preservim/nerdtree' |
      \ Plug 'Xuyuanp/nerdtree-git-plugin' |
      \ Plug 'ryanoasis/vim-devicons' |
      \ Plug 'tiagofumo/vim-nerdtree-syntax-highlight' |
      \ Plug 'unkiwii/vim-nerdtree-sync' |
      \ Plug 'tyok/nerdtree-ack'
```


```
" NERDTree config
nnoremap <C-n> :NERDTree<CR>
nmap <leader>d :NERDTreeToggle<CR>
nmap <leader>f :NERDTreeFind<CR>

" close the last buffer (file)
nnoremap <leader>c :bp\|bd #<CR>

" Start NERDTree and put the cursor back in the other window.
autocmd VimEnter * NERDTree | wincmd p

" Exit Vim if NERDTree is the only window remaining in the only tab.
autocmd BufEnter * if (winnr("$") == 1 && exists("b:NERDTree") && b:NERDTree.isTabTree()) | q | endif

let g:NERDSpaceDelims = 1
let g:NERDTreeGitStatusUseNerdFonts = 1
let g:NERDTreeHighlightCursorline = 1
let g:NERDTreeShowHidden = 1
let g:airline_powerline_fonts = 1
let g:nerdtree_sync_cursorline = 1
```


barrett@debian: ~ — -ssh barrett@192.168.80.129 — 150×40
.vimrc
.tmux.conf
buffers
" Press ? for help
.. (up a dir)
/home/barrett/
.cache/
.local/
.tmux/
.tmuxp/
.vim/
Projects/
.bash_history
.bash_logout
.bashrc
.profile
.tmux.conf
.viminfo
.vimrc
.Xresources
greeter.rb
tmux-conf-final
tmux-kill.sh*
tmux-up.sh*
vimrc-final
.vimrc.swp
1 # set scrollback history to 10000 (10k)
2 set -g history-limit 10000
3
4 # split panes using | and -
5 bind-key | split-window -h
6 bind-key - split-window -v
7
8 # pane movement similar to vim
9 bind-key h select-pane -L
10 bind-key j select-pane -D
11 bind-key k select-pane -U
12 bind-key l select-pane -R
13
14 # resize panes
15 bind-key -r H resize-pane -L 5
16 bind-key -r J resize-pane -D 5
17 bind-key -r K resize-pane -U 5
18 bind-key -r L resize-pane -R 5
19
20 # reload ~/.tmux.conf using PREFIX r
21 bind-key r source-file ~/.tmux.conf \; display "Reloaded!"
22
23 # Use vim keybindings in copy mode
24 set -g mode-keys vi
25 set -g status-keys vi
26
27 set -g default-terminal "screen-256color" # Mac and Ubuntu
28
29 # List of plugins
30 set -g @plugin 'tmux-plugins/tmux-battery'   # display battery status in tmux status-right
31 set -g @plugin 'tmux-plugins/tmux-continuum' # save tmux environment
32 set -g @plugin 'tmux-plugins/tmux-resurrect' # restore saved tmux environment
33 set -g @plugin 'tmux-plugins/tmux-yank'   # copy to the system clipboard in tmux
34 set -g @plugin 'tmux-plugins/tpm'   # tmux plugin manager
35
36 # prefix + Ctrl-s to save session
/home/barrett
NORMAL
.tmux.conf
tmux
utf-8
1% ℓ:1/82≡ ℅:1
".tmux.conf" 82L, 2927B
dotfiles
bash
0: bash*Z
2022-05-04 〈 Wed 〈 09:54 PM

Dotfiles

Bonus Content

https://dotfiles.github.io/

https://github.com/barrettclark/dotfiles

Thank you!

I hope you enjoyed your whirlwind tour through some things that I found helpful initially

https://github.com/barrettclark/intro-to-tmux-vim-configs.git