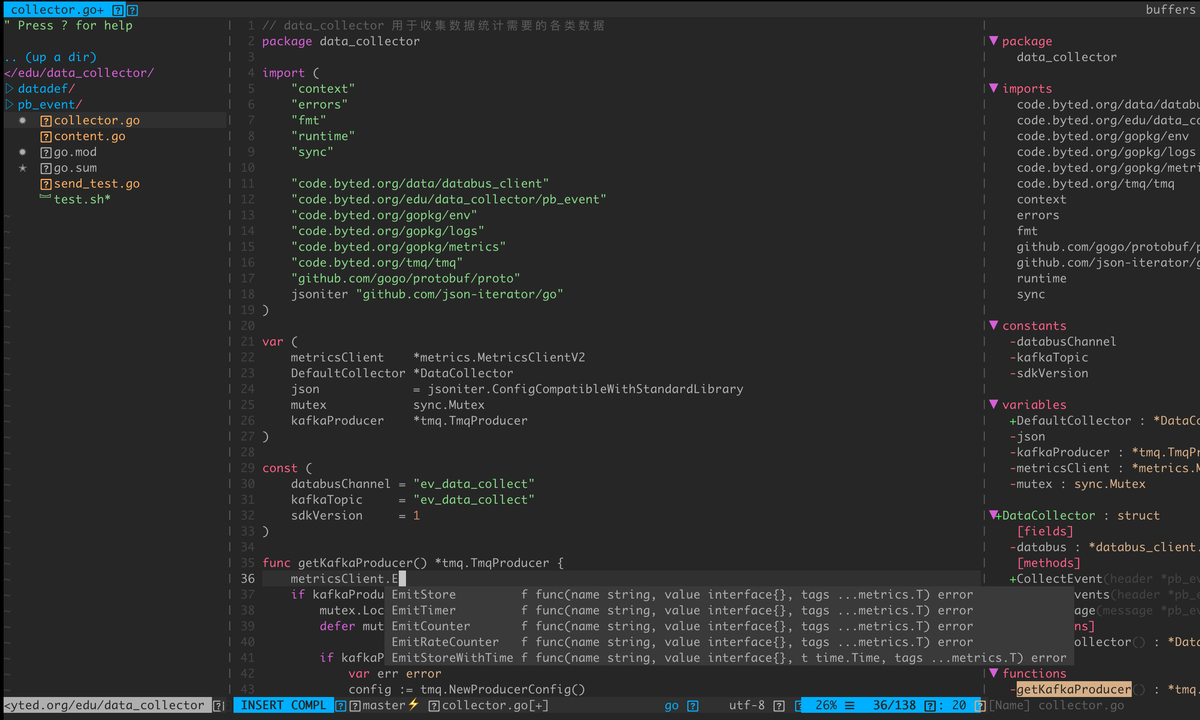
配置ide级别的go开发vim

基本效果



支持强大且快速的语义级别自动补全（基本所有都能补全），大量的go相关扩展命令等

强烈建议每个人都在devbox上配置一份，即使平时主要用mac+idea\_or\_goland，有时候也可能在devbox上修改，所以一个高效的vim环境很重要（也可以考虑以下直接在devbox上vim开发？）

如果遇到奇怪的问题可以​@刘斐浩​ ，这里面的坑基本都踩过

一、准备环境基础

确保你devbox没有动过系统的动态链接库（例如libclang.so，YomCompleteMe插件强依赖此库）。强烈建议在devbox上弄，mac用goland或idea就行，在mac配不是太必要。

**为了避免奇怪问题的发生，请先快照你的devbox**

TODO：mac的具体配置等后续完善之后再发。

二、等一个晚上

git clone https://github.com/chxuan/vimplus.git ~/.vimplus

cd ~/.vimplus

./install.sh

强烈建议：安装全程需要十几分钟到半个小时到左右，主要是YouCompleteMe插件

因此更省心的是：在长时间不用开发机时（建议晚上下班时，保证时间够长）执行第三条命令，不过是挂在后台

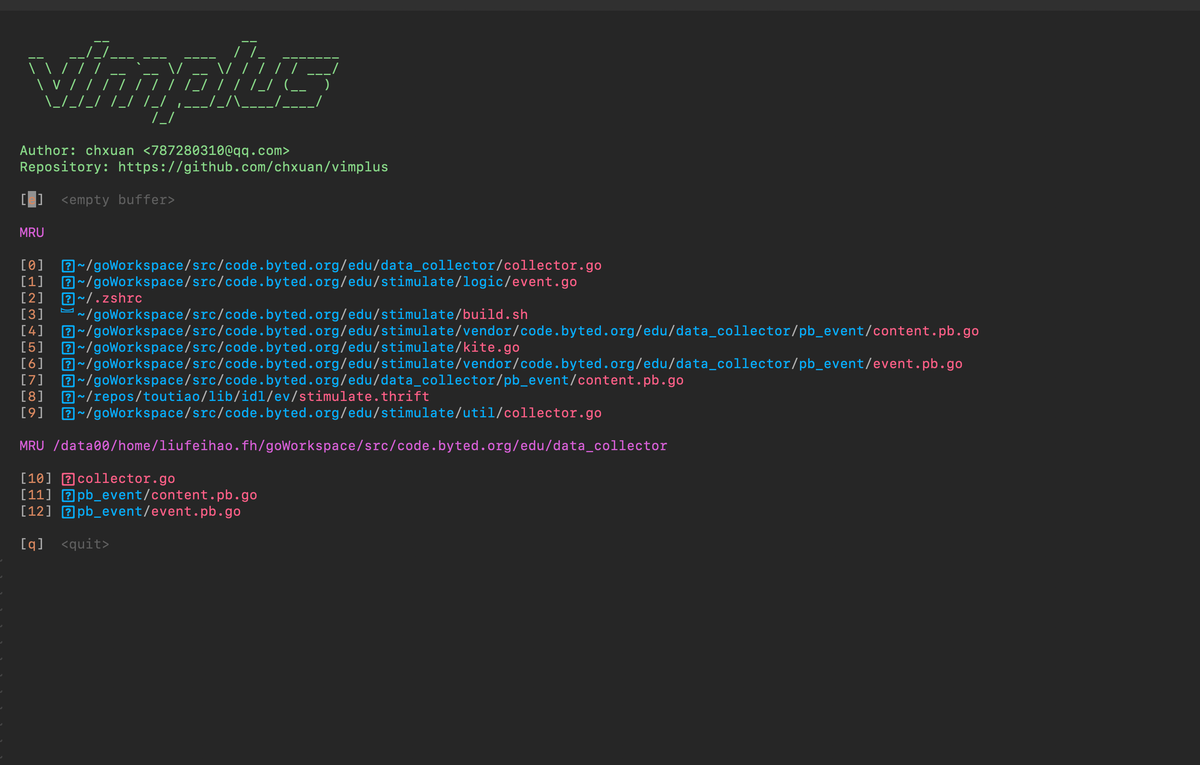
nohup ./install.sh &

现在你可以合上电脑，等第二天早上再来(?)，其实远远不需要这么久，但毕竟为了成本最小化，没人愿意等着进度条

**warning: 一旦脚本开始执行不要随意停止，因为脚本的第一步就是删除本机vim，安装vim8.0并扩展所需插件。所以真的建议挂在后台......**

三、配置vim-go

时间到了早上，假设你配置没有出现任何问题的话，你输入vim就会看到这样，下方可能会有某些报错，有时候不太要紧。



继续

vim-go是一个强大的集成go开发插件，集成了所有与go相关的vim插件

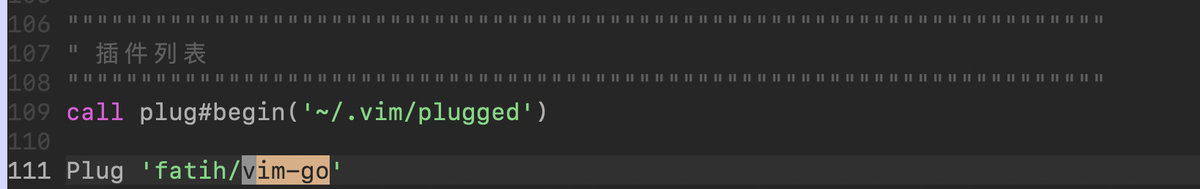
我们开始安装vim-go

1、打开你的vim配置文件

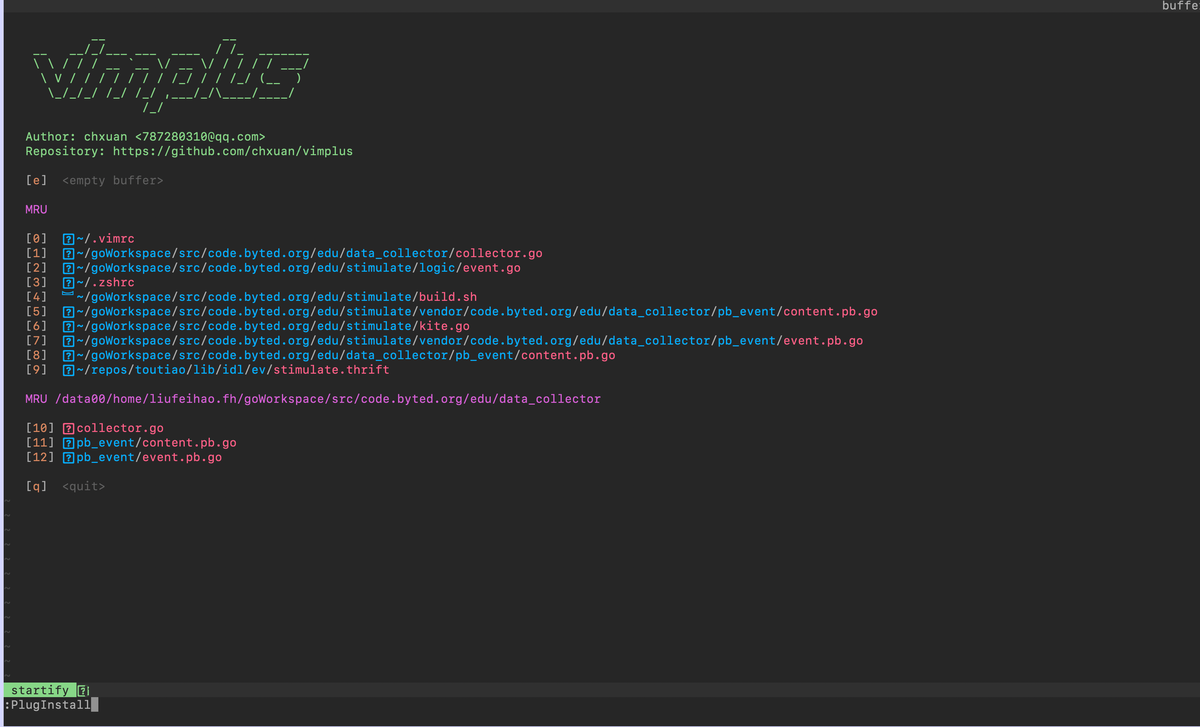
vim ~/.vimrc

2、拉取插件

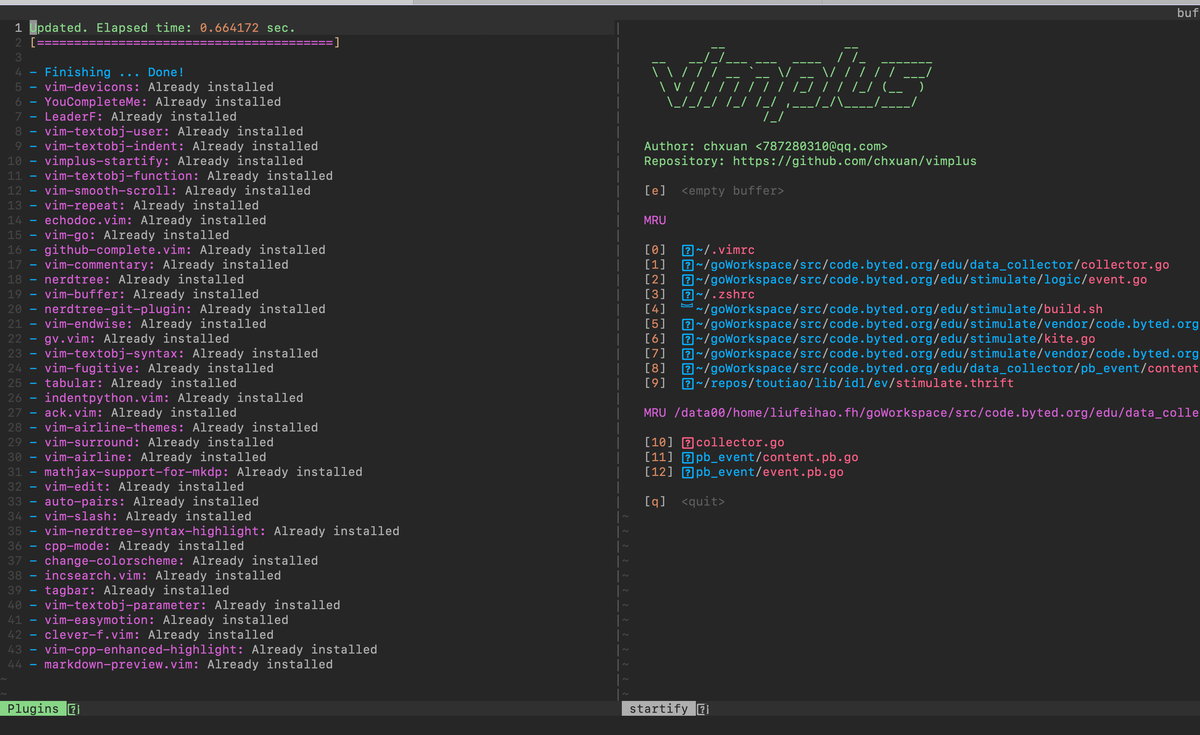
跳转到这个地方，注意在109行这样的代码之后写入这一行（以后插件都是这样安装，后续可以自行学习）



保存后退出，再直接进入vim，在命令模式输入如下

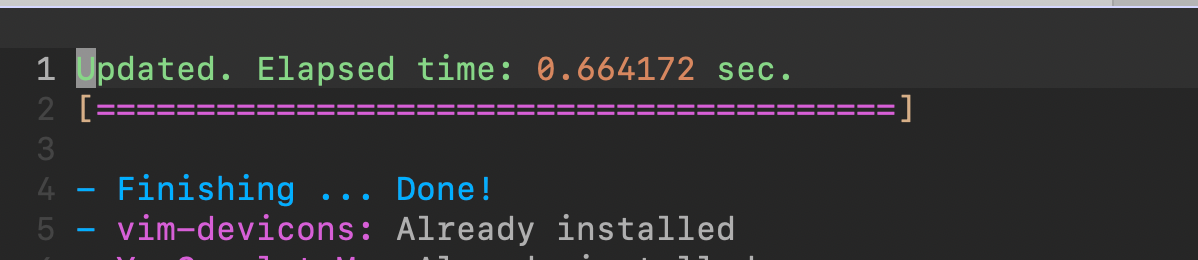


然后界面会进入从github拉取插件的界面（注意这还只是拉仓库，vim-go还未安装，不过有些插件就自动安装上了）



ps：中间可能会在某个插件卡住，这时退出再输入上面的PlugInstall命令，还卡住就重复（一般最多卡一两次）

当所有插件都拉取好后，如下



3、安装vim-go

进入vim命令模式，输入如下命令

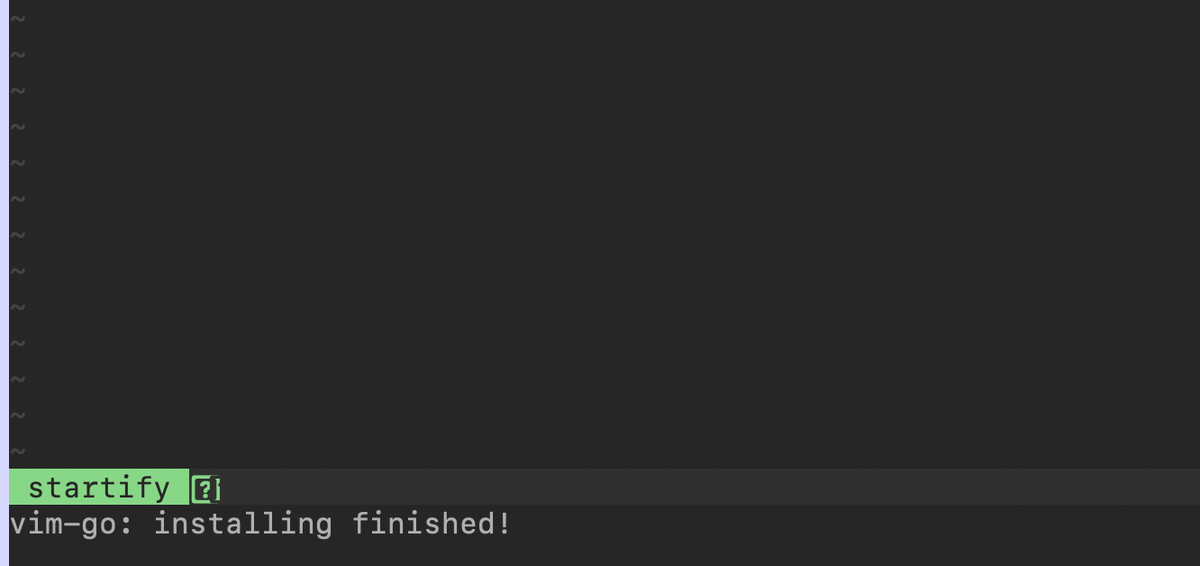
GoInstallBinaries

然后你会发现出现许多go的插件拉不到的错误，这是你有两个选择

1、挂http和https代理

2、自己去逐个单独git到对应$GOPATH下并手动安装

接下来反复执行GoInstallBinaries，最后全部安装后如下



到此为止就全部配置完了，后续就是熟练使用各个插件的过程

具体请简单学习[vim-go-tutorial](https://github.com/fatih/vim-go-tutorial" \l "hello-world" \t "/Users/yangwenshuo/Documents\\x/_blank)，大约耗时40分钟

关于一些辅助性质的插件[使用说明](https://github.com/chxuan/vimplus/blob/master/README.md" \t "/Users/yangwenshuo/Documents\\x/_blank)

最后....

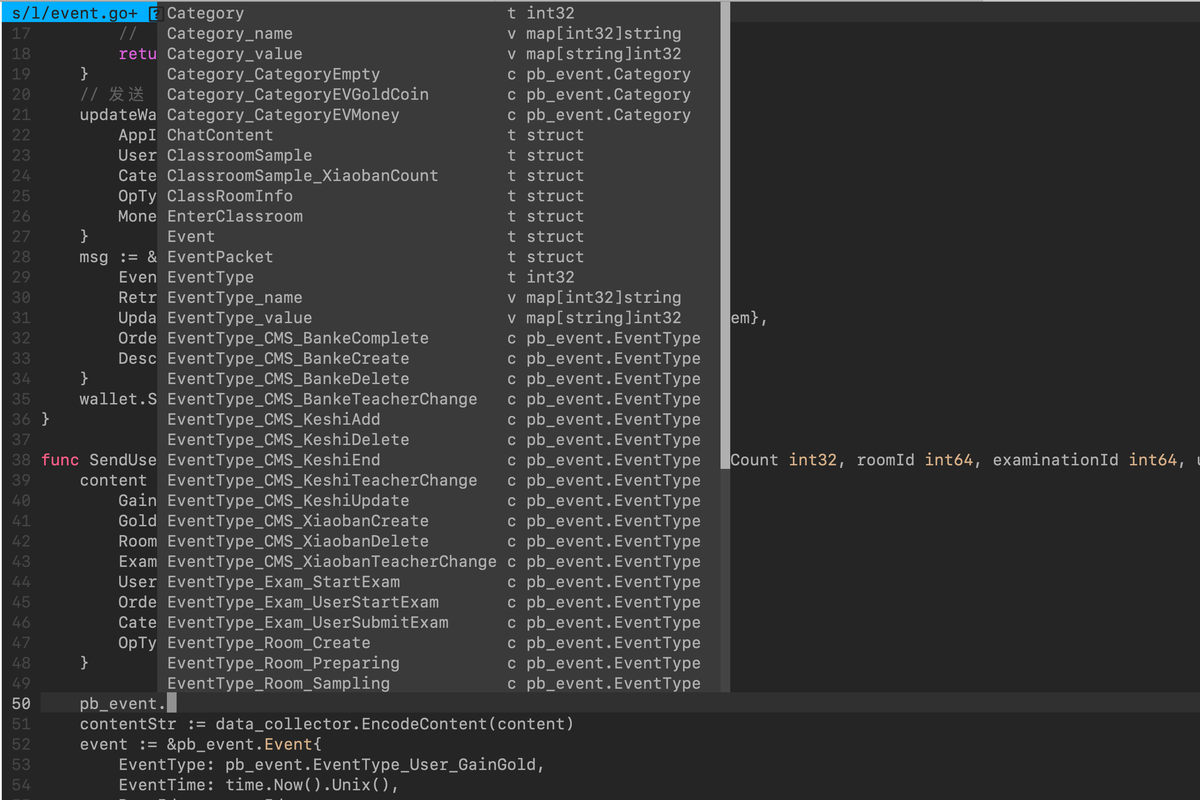
**有一个需要注意的是，自己用不到的插件最好删掉**，不然非常影响vim速度，可以通过如下方式查看是哪些插件在影响速度

vim (xxx任意文件） --startuptime output.log

//output.log中有整个启动过程中各个插件的耗时

以下是比较完美的效果（由于是全异步加载，速度极快）





这份配置关注了比较久，算是比较合理的一份配置，随着你越来越熟练，也可以自己维护起自己的一份配置。给大家分享一下我自己的配置(~/.vimrc)，这份配置在公司的devbox上的体验可能会比上面更好

"==============================================================================

" 处理 Gnome 终端不能使用 alt 快捷键

" 参考：http://landcareweb.com/questions/8623/altjian-kuai-jie-jian-bu-gua-yong-yu-dai-you-vimde-gnomezhong-duan

"==============================================================================

let c='a'

while c <= 'z'

  exec "set <A-".c.">=\e".c

  exec "imap \e".c." <A-".c.">"

  let c = nr2char(1+char2nr(c))

endw

set timeout ttimeoutlen=50

"==============================================================================

" vim 内置配置

"==============================================================================

" utf-8编码

set fileencodings=utf-8,gbk,utf-16le,cp1252,iso-8859-15,ucs-bom

set termencoding=utf-8

set encoding=utf-8

" 设置 vimrc 修改保存后立刻生效，不用在重新打开

" 建议配置完成后将这个关闭，否则配置多了之后会很卡

" autocmd BufWritePost $MYVIMRC source $MYVIMRC

set noswapfile

" 关闭兼容模式

set nocompatible

set hlsearch " 高亮搜索

set number " 设置绝对行号

" set relativenumber " 设置相对行号

set cursorline "突出显示当前行

" set cursorcolumn " 突出显示当前列

set showmatch " 显示括号匹配

set backspace=2

" tab 缩进

set tabstop=4 " 设置Tab长度为4空格

set shiftwidth=4 " 设置自动缩进长度为4空格

set autoindent " 继承前一行的缩进方式，适用于多行注释

cnoreabbrev WQ wq

cnoreabbrev Wq wq

" 移动到行开头、结尾

nnoremap <c-f> <esc>$

nnoremap <c-a> <esc>^

" 保存快捷

nnoremap fwq :wq<cr>

" 快捷推出

nnoremap fq :q<cr>

" python

let python\_highlight\_all=1

" 定义快捷键的前缀，即<Leader>

let mapleader=","

" 自定义快捷键

" 编辑vimrc文件

nnoremap <leader>e :edit $MYVIMRC<cr>

" 重新加载vimrc文件

nnoremap <leader>s :source $MYVIMRC<cr>

" split

nnoremap <leader>x :vsp .<cr>

" ==== 系统剪切板复制粘贴 ====

" v 模式下复制内容到系统剪切板

vmap <M-c> "+yy

" n 模式下复制一行到系统剪切板

nmap <M-c> "+yy

" n 模式下粘贴系统剪切板的内容

nmap <M-v> "+p

" 分屏窗口移动

nnoremap <c-j> <c-w>j

nnoremap <c-k> <c-w>k

nnoremap <c-h> <c-w>h

nnoremap <c-l> <c-w>l

nmap <Leader>nu :set relativenumber<CR>

nmap <Leader>nn :set norelativenumber<CR>

" 开启实时搜索

set incsearch

" 搜索时大小写不敏感

set ignorecase

syntax enable

syntax on                    " 开启文件类型侦测

filetype plugin indent on    " 启用自动补全

" 退出插入模式指定类型的文件自动保存

au InsertLeave \*.go,\*.sh write

"==============================================================================

" 插件配置

"==============================================================================

" 插件开始的位置

call plug#begin('~/.vim/plugged')

" Vim 中文文档

Plug 'yianwillis/vimcdoc'

" python自动格式化

" Plug 'Chiel92/vim-autoformat'

" 有点烦.......

" python语法检查

" Plug 'w0rp/ale'

" Shorthand notation; fetches https://github.com/junegunn/vim-easy-align

" 可以快速对齐的插件

Plug 'junegunn/vim-easy-align'

" 用来提供一个导航目录的侧边栏

Plug 'scrooloose/nerdtree'

" 可以使 nerdtree Tab 标签的名称更友好些

Plug 'jistr/vim-nerdtree-tabs'

" 有道词典在线翻译

Plug 'ianva/vim-youdao-translater'

" 可以在导航目录中看到 git 版本信息

Plug 'Xuyuanp/nerdtree-git-plugin'

" 查看当前代码文件中的变量和函数列表的插件，

" 可以切换和跳转到代码中对应的变量和函数的位置

" 大纲式导航, Go 需要 https://github.com/jstemmer/gotags 支持

Plug 'majutsushi/tagbar'

" 自动补全括号的插件，包括小括号，中括号，以及花括号

Plug 'jiangmiao/auto-pairs'

" Vim状态栏插件，包括显示行号，列号，文件类型，文件名，以及Git状态

Plug 'vim-airline/vim-airline'

" 代码块

Plug 'SirVer/ultisnips'

" 代码自动完成，安装完插件还需要额外配置才可以使用

Plug 'Valloric/YouCompleteMe'

" 配色方案

" colorscheme gruvbox

Plug 'morhetz/gruvbox'

" colorscheme one

Plug 'rakr/vim-one'

Plug 'altercation/vim-colors-solarized'

" go 主要插件

Plug 'fatih/vim-go', { 'tag': '\*' }

" go 中的代码追踪，输入 gd 就可以自动跳转

" Plug 'dgryski/vim-godef'

" 自动生成注释的插件

Plug 'scrooloose/nerdcommenter'

" 插件结束的位置，插件全部放在此行上面

call plug#end()

"==============================================================================

" 主题配色

"==============================================================================

" 开启24bit的颜色，开启这个颜色会更漂亮一些

" set termguicolors

set t\_Co=256

" 配色方案, 可以从上面插件安装中的选择一个使用

colorscheme gruvbox " gruvbox " 主题

" colorscheme solarized " 主题

set background=light " 主题背景 dark-深色; light-浅色

"==============================================================================

" vim-go 插件

"==============================================================================

let g:go\_fmt\_command = "goimports" " 格式化将默认的 gofmt 替换

let g:go\_autodetect\_gopath = 1

let g:go\_list\_type = "quickfix"

let g:go\_version\_warning = 1

let g:go\_highlight\_types = 1

let g:go\_highlight\_fields = 1

let g:go\_highlight\_functions = 1

let g:go\_highlight\_function\_calls = 1

let g:go\_highlight\_operators = 1

let g:go\_highlight\_extra\_types = 1

let g:go\_highlight\_methods = 1

let g:go\_highlight\_generate\_tags = 1

let g:godef\_split=2

" 直接通过 go run 执行当前文件

autocmd FileType go nmap <leader>r :GoRun %<CR>

"==============================================================================

" NERDTree 插件

"==============================================================================

" 打开和关闭NERDTree快捷键

map <F10> :NERDTreeToggle<CR>

nmap <M-m> :NERDTreeFind<CR>

" 显示行号

let NERDTreeShowLineNumbers=1

" 打开文件时是否显示目录

let NERDTreeAutoCenter=0

" 是否显示隐藏文件

let NERDTreeShowHidden=0

" 设置宽度

" let NERDTreeWinSize=31

" 忽略一下文件的显示

let NERDTreeIgnore=['\.pyc','\~$','\.swp']

" 打开 vim 文件及显示书签列表

let NERDTreeShowBookmarks=2

" 在终端启动vim时，共享NERDTree

" let g:nerdtree\_tabs\_open\_on\_console\_startup=1

"==============================================================================

" Chiel92/vim-autoformat 插件

"==============================================================================

nnoremap <F6> :Autoformat<CR>

let g:autoformat\_autoindent = 0

let g:autoformat\_retab = 0

let g:autoformat\_remove\_trailing\_spaces = 0

"==============================================================================

" ale 插件

"==============================================================================

" let g:ale\_fix\_on\_save = 1

" let g:ale\_completion\_enabled = 1

" let g:ale\_sign\_column\_always = 1

" let g:airline#extensions#ale#enabled = 1

"==============================================================================

"  majutsushi/tagbar 插件

"==============================================================================

" majutsushi/tagbar 插件打开关闭快捷键

nmap <F9> :TagbarToggle<CR>

let g:tagbar\_type\_go = {

        \ 'ctagstype' : 'go',

        \ 'kinds'     : [

                \ 'p:package',

                \ 'i:imports:1',

                \ 'c:constants',

                \ 'v:variables',

                \ 't:types',

                \ 'n:interfaces',

                \ 'w:fields',

                \ 'e:embedded',

                \ 'm:methods',

                \ 'r:constructor',

                \ 'f:functions'

        \ ],

        \ 'sro' : '.',

        \ 'kind2scope' : {

                \ 't' : 'ctype',

                \ 'n' : 'ntype'

        \ },

        \ 'scope2kind' : {

                \ 'ctype' : 't',

                \ 'ntype' : 'n'

        \ },

        \ 'ctagsbin'  : 'gotags',

        \ 'ctagsargs' : '-sort -silent'

\ }

"==============================================================================

"  nerdtree-git-plugin 插件

"==============================================================================

let g:NERDTreeIndicatorMapCustom = {

    \ "Modified"  : "✹",

    \ "Staged"    : "✚",

    \ "Untracked" : "✭",

    \ "Renamed"   : "➜",

    \ "Unmerged"  : "═",

    \ "Deleted"   : "✖",

    \ "Dirty"     : "✗",

    \ "Clean"     : "✔︎",

    \ 'Ignored'   : '☒',

    \ "Unknown"   : "?"

    \ }

let g:NERDTreeShowIgnoredStatus = 1

nmap <Leader>pwd :NERDTreeCWD<CR>

"==============================================================================

"  Valloric/YouCompleteMe 插件

"==============================================================================

" make YCM compatible with UltiSnips (using supertab)

let g:ycm\_key\_list\_select\_completion = ['<M-j>', '<DOWN>']

let g:ycm\_key\_list\_previous\_completion = ['<M-k>', '<Up>']

let g:SuperTabDefaultCompletionType = '<M-j>'

let g:ycm\_confirm\_extra\_conf = 0

" 关闭了提示再次触发的快捷键

let g:ycm\_key\_invoke\_completion = '<Leader>,'

let g:ycm\_filetype\_whitelist = {

                        \ "c":1,

                        \ "cpp":1,

                        \ "sh":1,

                        \ "zsh":1,

                        \ "go":1,

                        \ "python":1,

                        \ }

let g:ycm\_server\_python\_interpreter='/usr/bin/python3'

let g:ycm\_error\_symbol = '>>'

let g:ycm\_warning\_symbol = '>\*'

nnoremap <leader>d :YcmCompleter GoToDefinition<CR>

nnoremap <leader>g :YcmCompleter GoToDefinitionElseDeclaration<CR>

map <F4> :YcmDiags<CR>

"==============================================================================

" UltiSnips 插件

"==============================================================================

" better key bindings for UltiSnipsExpandTrigger

let g:UltiSnipsExpandTrigger = "<tab>"

let g:UltiSnipsJumpForwardTrigger = "<tab>"

let g:UltiSnipsJumpBackwardTrigger = "<s-tab>"

"==============================================================================

"  其他插件配置

"==============================================================================

" tab 标签页切换快捷键

:nn <Leader>1 1gt

:nn <Leader>2 2gt

:nn <Leader>3 3gt

:nn <Leader>4 4gt

:nn <Leader>5 5gt

:nn <Leader>6 6gt

:nn <Leader>7 7gt

:nn <Leader>8 8gt

:nn <Leader>9 8gt

:nn <Leader>0 :tablast<CR>

" 自动注释的时候添加空格

let g:NERDSpaceDelims=1

" 自动切换到当前文件所在的目录 cdpath

map <Leader>cd :cd %:h<CR>

"==============================================================================

" 光标随着插入模式改变形状

" 参考： https://vim.fandom.com/wiki/Change\_cursor\_shape\_in\_different\_modes

"==============================================================================

if has("autocmd")

        au VimEnter,InsertLeave \* silent execute '!echo -ne "\e[2 q"' | redraw!

        au InsertEnter,InsertChange \*

                \ if v:insertmode == 'i' |

                \   silent execute '!echo -ne "\e[6 q"' | redraw! |

                \ elseif v:insertmode == 'r' |

                \   silent execute '!echo -ne "\e[4 q"' | redraw! |

                \ endif

        au VimLeave \* silent execute '!echo -ne "\e[ q"' | redraw!

endif

"==============================================================================

" vim 的 session 管理

"==============================================================================

"

" 自动保存 session

autocmd VimLeave \* mks! ~/.vim/session.vim

" 加载 session 的快捷键

nmap <Leader>his :source ~/.vim/session.vim<CR>