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

" DesCRiption: 适合自己使用的vimrc文件，for Linux/Windows, GUI/Console

" Author: 陈昌栋

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

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

" Vim全局设置

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

"插入模式下用绝对行号<leader>lr开关绝对行号

" <leader>lh行号开关，用于鼠标复制代码用

" 为方便复制，用<leader>lh开启/关闭行号显示:

"-------------------行号相关结束--------------------

"-------------------操作相关开始--------------------

"使用Ctrl+Z保存

imap <C-Z> <C-O>:update<CR>

vmap <C-Z> <C-O>:update<CR>

nmap <C-Z> :update<CR>

"使用Ctrl+B撤回

imap <C-B> <C-O>:u<CR>

vmap <C-B> <C-O>:u<CR>

nmap <C-B> :u<CR>

" <leader>s 语法开关，关闭语法可以加快大文件的展示

nnoremap <leader>s :exec exists('syntax\_on') ? 'syn off' : 'syn on'<CR>

" Go to home and end using capitalized directions

noremap H ^ " H一行第一个字符处

noremap L $ "L一行最后一个字符处

" Map ; to : and save a million keystrokes 用于快速进入命令行

nnoremap ; :

" => 选中及操作改键

" 调整缩进后自动选中，方便再次操作

vnoremap < <gv

vnoremap > >gv

" y$ -> Y Make Y behave like other capitals

map Y y$

" 复制选中区到系统剪切板中

vnoremap <leader>p "+y

" select all 全选

map <Leader>a ggVG"

" select block 选择块

nnoremap <leader>v V`}

" Buffers操作快捷方式!

nnoremap <C-RETURN> :bnext<CR>

nnoremap <C-S-RETURN> :bprevious<CR>

" Tab操作快捷方式!

nnoremap <C-TAB> :tabnext<CR>

nnoremap <C-S-TAB> :tabprev<CR>

"关于tab的快捷键

map tn :tabnext<CR>

map tp :tabprevious<CR>

map td :tabnew .<CR>

map te :tabedit

map tc :tabclose<CR>

"-------------------操作相关结束--------------------

"-------------------粘贴相关开始--------------------

set pastetoggle=<C-G> " 插入模式下进入粘贴模式。when in insert mode, press Ctrl-G to go to paste mode.

au InsertLeave \* set nopaste " 离开插入模式后关闭粘贴模式。disbale paste mode when leaving insert mode

" Ctrl-G set paste问题已解决, 粘贴代码前不需要按Ctrl-G了

" Ctrl-G 粘贴模式paste\_mode开关,用于有格式的代码粘贴

" Automatically set paste mode in Vim when pasting in insert mode

"-------------------粘贴相关结束--------------------

"-------------------搜索相关开始--------------------

" Map <leader>' to / (search) and Ctrl-<Space> to ? (backwards search) 通模式下 “空格”查找

map <leader>' /

" 进入搜索Use sane regexes"

nnoremap / /\v

vnoremap / /\v

" <leader>.去掉搜索高亮

noremap <silent><leader>. :nohls<CR>

"-------------------搜索相关结束--------------------

======================================

"-------------------文件设置快捷键相关开始--------------------

"一些不错的映射转换语法（如果在一个文件中混合了不同语言时有用）

nnoremap <leader>1 :set filetype=c<CR>

nnoremap <leader>2 :set filetype=cpp<CR>

nnoremap <leader>3 :set filetype=python<CR>

nnoremap <leader>4 :set filetype=java<CR>

"设置文件格式unix,dos,mac

set fileformats=unix,dos,mac

nmap <leader>fd :se fileformat=dos<CR>

nmap <leader>fu :se fileformat=unix<CR>

"编码格式改变fileencodings=utf-8,cp936,gb2312,gbk,gb18030

nnoremap <leader>gu :set fileencoding=utf-8<CR>

nnoremap <leader>g2 :set fileencoding=gb2312<CR>

nnoremap <leader>g9 :set fileencoding=cp936<CR>

nnoremap <leader>g1 :set fileencoding=gb18030<CR>

nnoremap <leader>gk :set fileencoding=gbk<CR>

"-------------------文件设置快捷键相关结束--------------------

"-------------------不同文件类型对应不同设置开始--------------------

" Python 文件的一般设置，比如不要 tab 等

autocmd FileType python map <F5> :!ipython --pdb %<CR> "运行Python如果有错误就用ipdb调试

"-------------------不同文件类型对应不同设置结束--------------------

"-------------------编译运行快捷键设置开始--------------------

"C，C++ ,Java,Python按F1编译运行

map <F1> :call CompileRunGcc()<CR>

"C，C++ ,Python的按F9调试

map <F2> :call Rungdb()<CR>

" 映射全选+复制 Ctrl+A

map <C-A> ggVGY

map! <C-A> <Esc>ggVGY

"Ctrl+C b-去空行

nnoremap <C-C>b :g/^\s\*$/d<CR>

"Ctrl+C c比较文件

nnoremap <C-C>c :vert diffsplit

" 新建Tab, Ctrl+T

nnoremap <C-T> :tabnew<CR>

inoremap <C-T> <Esc>:tabnew<CR>

"-------------------编译运行设置结束--------------------

"-----------------------------------------------------------------

" plugin - bufexplorer.vim Buffers切换

" <leader>be 全屏方式查看全部打开的文件列表

" <leader>bv 左右方式查看 <leader>bs 上下方式查看

"-----------------------------------------------------------------

"-----------------------------------------------------------------

" plugin - taglist.vim 查看函数列表，需要ctags程序

" F4 打开隐藏taglist窗口

"-----------------------------------------------------------------

nnoremap <silent><F4> :TlistToggle<CR>

"-------------------plugin - taglist.vim结束------------------------

"-----------------------------------------------------------------

" plugin - NERD\_tree.vim 以树状方式浏览系统中的文件和目录

" :ERDtree 打开NERD\_tree :NERDtreeClose 关闭NERD\_tree

" o 打开关闭文件或者目录 t 在标签页中打开

" T 在后台标签页中打开 ! 执行此文件

" p 到上层目录 P 到根目录

" K 到第一个节点 J 到最后一个节点

" u 打开上层目录 m 显示文件系统菜单（添加、删除、移动操作）

" r 递归刷新当前目录 R 递归刷新当前根目录

"-----------------------------------------------------------------

" F3 NERDTree 切换

map <F3> :NERDTreeToggle<CR>

imap <F3> <ESC>:NERDTreeToggle<CR>

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

" 设置FuzzyFinder \*

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

map <leader>fF :FufFile<CR>

map <leader>ff :FufTaggedFile<CR>

map <leader>fg :FufTag<CR>

map <leader>fb :FufBuffer<CR>

"-----------------------------------------------------------------

" davidhalter/jedi-vim'

" Python语法提示

"-----------------------------------------------------------------

"Goto assignments(typical goto function)

let g:jedi#goto\_assignments\_command = "<leader>ja"

"Goto definitions(follow identifier as far as possible,includes imports and statements)

let g:jedi#goto\_definitions\_command = "<leader>jd"

"how Documentation/Pydoc K (shows a popup with assignments)

let g:jedi#documentation\_command = "K"

"Usages <leader>ju (shows all the usages of a name)

let g:jedi#usages\_command = "<leader>ju"

let g:jedi#completions\_command = "<C-X>" "Completion

"Renaming

let g:jedi#rename\_command = "<leader>jr"

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

" EasyGrep配置 \*

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

" Ctrl+C f 开始查找

map <C-C>f :Grep

imap <C-C>f <ESC>:Grep

" Ctrl+C r 开始替换

map <C-C>r :Replace

imap <C-C>r <ESC>:Replace

" quickfix模式,<leader>+空格—执行保存文件执行make命令

autocmd FileType c,cpp map <buffer> <leader>q<space> :w<cr>:make<cr>

"-----------------------------------------------------------------

" plugin - indentLine

" vim垂直缩进对齐线插件——indentLine

"-----------------------------------------------------------------

" 开启/关闭对齐线

nmap <leader>il :IndentLinesToggle<CR>

"-----------------------------------------------------------------

" 插件fholgado/minibufexpl.vim

" 多文档编辑

"-----------------------------------------------------------------

"使用Ctrl+D关闭Buffer中的一个文件

imap <C-D> <C-O>:bd<CR>

vmap <C-D> <C-O>:bd<CR>

nmap <C-D> :bd<CR>

" buffer 切换快捷键，默认方向键左右可以切换buffer

map <C-H> :MBEbn<cr>

map <C-L> :MBEbp<cr>

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

" YouCompleteMe配置 \*

"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

"按<leader>yj 会跳转到定义

nnoremap <leader>yj :YcmCompleter GoToDefinitionElseDeclaration<CR>

" YCM 集成 OmniCppComplete 补全引擎，设置其快捷键

inoremap <leader>yc <C-x><C-o>

" 修改对函数的补全快捷键，默认是CTRL + space，修改为CTRL + X;

let g:ycm\_key\_invoke\_completion = '<C-X>'

"This option controls the key mapping used to show the full diagnostic

"text when the user's cursor is on the line with the diagnostic.

let g:ycm\_key\_detailed\_diagnostics = '<leader>yd'

" 设置转到定义处的快捷键为CTRL + J，这个功能非常赞

" mapping

inoremap <expr> <CR> pumvisible()?"\<C-Y>":"\<CR>" " 回车即选中当前项

inoremap <expr> <C-K> pumvisible()?"\<C-E>":"\<C-U>" " 关闭补全列表

"-----------------------------------------------------------------

" plugin - mark.vim 给各种tags标记不同的颜色，便于观看调式的插件。

" <leader>m mark or unmark the word under (or before) the cursor

" <leader>r manually input a regular expression. 用于搜索.

" <leader>n clear this mark (i.e. the mark under the cursor), or clear all highlighted marks .

" <leader>\* 当前MarkWord的下一个 \# 当前MarkWord的上一个

" <leader>/ 所有MarkWords的下一个 \? 所有MarkWords的上一个

"-----------------------------------------------------------------

"-----------------------------------------------------------------

" plugin - NERD\_commenter.vim 注释代码用的，

" [count]<leader>cc 光标以下count行逐行添加注释(7,cc)

" [count]<leader>cu 光标以下count行逐行取消注释(7,cu)

" [count]<leader>cm 光标以下count行尝试添加块注释(7,cm)

" <leader>cA 在行尾插入 /\* \*/,并且进入插入模式。 这个命令方便写注释。

" 注：count参数可选，无则默认为选中行或当前行

"-----------------------------------------------------------------

" -----------------------------------------------------------------------------

" < a.vim 插件配置 > 用于切换C/C++头文件

" -----------------------------------------------------------------------------

" :A ---切换头文件并独占整个窗口

" :AV ---切换头文件并垂直分割窗口

" :AS ---切换头文件并水平分割窗口

" :close

" :only

" -----------------------------------------------------------------------------

" < SrcExpl 插件配置 > 增强源代码浏览，其功能就像Windows中的"Source Insight"

" -----------------------------------------------------------------------------

nmap <C-m>s :SrcExplToggle<CR> "打开/闭浏览窗口

let g:SrcExpl\_jumpKey = "<C-m><ENTER>"

let g:SrcExpl\_gobackKey = "<C-m><SPACE>"

let g:SrcExpl\_prevDefKey = "<C-m><F3>"

let g:SrcExpl\_nextDefKey = "<C-m><F4>"

" -----------------------------------------------------------------------------

" Zoom / Restore window.设置

command! ZoomToggle call s:ZoomToggle()

nnoremap <silent> <Leader>z :ZoomToggle<CR> "<leader>z 触发放大 (没有作用)