## 1 Vim设置~/.vimrc或者 /etc/vim/vimrc

|  |
| --- |
| "=========================================================================  "  "" DesCRiption: 适合自己使用的vimrc文件，for Linux/Windows, GUI/Console  "  ""=========================================================================  syntax on "自动语法高亮  "winpos 5 5 " 设定窗口位置  "set lines=40 columns=155 " 设定窗口大小  "set nu " 显示行号  set go= " 不要图形按钮  "color asmanian2 " 设置背景主题  set guifont=Courier\_New:h10:cANSI " 设置字体  "syntax on " 语法高亮  "autocmd InsertLeave \* se nocul " 用浅色高亮当前行  "autocmd InsertEnter \* se cul " 用浅色高亮当前行  "set ruler " 显示标尺  set showcmd " 输入的命令显示出来，看的清楚些  "set cmdheight=1 " 命令行（在状态行下）的高度，设置为1  "set whichwrap+=<,>,h,l " 允许backspace和光标键跨越行边界(不建议)  "set scrolloff=3 " 光标移动到buffer的顶部和底部时保持3行距离  set novisualbell " 不要闪烁(不明白)  "set statusline=%F%m%r%h%w\ [FORMAT=%{&ff}]\ [TYPE=%Y]\ [POS=%l,%v][%p%%]\ %{strftime(\"%d/%m/%y\ -\ %H:%M\")} "状态行显示的内容  "set laststatus=1 " 启动显示状态行(1),总是显示状态行(2)  set foldenable " 允许折叠  set foldmethod=manual " 手动折叠  "set background=dark "背景使用黑色  set nocompatible "去掉讨厌的有关vi一致性模式，避免以前版本的一些bug和局限  " 显示中文帮助  if version >= 603  set helplang=cn  set encoding=utf-8  endif  " 设置配色方案  "colorscheme murphy  "字体  "if (has("gui\_running"))  " set guifont=Bitstream\ Vera\ Sans\ Mono\ 10  "endif    set fencs=utf-8,ucs-bom,shift-jis,gb18030,gbk,gb2312,cp936  set termencoding=utf-8  set encoding=utf-8  set fileencodings=ucs-bom,utf-8,cp936  set fileencoding=utf-8"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""  """""新文件标题""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""  "新建.c,.h,.sh,.java文件，自动插入文件头  autocmd BufNewFile \*.cpp,\*.[ch],\*.sh,\*.java exec ":call SetTitle()"  ""定义函数SetTitle，自动插入文件头  func SetTitle()  "如果文件类型为.sh文件  if &filetype == 'sh'  call setline(1,"\#########################################################################")  call append(line("."), "\# File Name: ".expand("%"))  call append(line(".")+1, "\# Author: 冷倾殇")  call append(line(".")+2, "\# mail: 1500428751@qq.com")  call append(line(".")+3, "\# Created Time: ".strftime("%c"))  call append(line(".")+4, "\#########################################################################")  call append(line(".")+5, "\#!/bin/bash")  call append(line(".")+6, "")  else  call setline(1, "/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")  call append(line("."), " > File Name: ".expand("%"))  call append(line(".")+1, " > Author: 冷倾殇")  call append(line(".")+2, " > Mail: 1500428751@qq.com ")  call append(line(".")+3, " > Created Time: ".strftime("%c"))  call append(line(".")+4, " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/")  call append(line(".")+5, "")  endif  " if &filetype == 'cpp'  " call append(line(".")+6, "#include<iostream>")  " call append(line(".")+7, "using namespace std;")  " call append(line(".")+8, "")  " endif  " if &filetype == 'c'  " call append(line(".")+6, "#include<stdio.h>")  " call append(line(".")+7, "")  " endif  "新建文件后，自动定位到文件末尾  autocmd BufNewFile \* normal G  endfunc  """"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""  "键盘命令  """"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""  nmap <leader>w :w!<cr>  nmap <leader>f :find<cr>  " 映射全选+复制 ctrl+a  map <C-A> ggVGY  map! <C-A> <Esc>ggVGY  map <F12> gg=G  " 选中状态下 Ctrl+c 复制  vmap <C-c> "+y  "去空行  nnoremap <F2> :g/^\s\*$/d<CR>  "比较文件  nnoremap <C-F2> :vert diffsplit  "新建标签  map <M-F2> :tabnew<CR>  "列出当前目录文件  map <F3> :tabnew .<CR>  "打开树状文件目录  map <C-F3> \be  "C，C++ 按F5编译运行  map <F5> :call CompileRunGcc()<CR>  func! CompileRunGcc()  exec "w"  if &filetype == 'c'  exec "!g++ % -o %<"  exec "! ./%<"  elseif &filetype == 'cpp'  exec "!g++ % -o %<"  exec "! ./%<"  elseif &filetype == 'java'  exec "!javac %"  exec "!java %<"  elseif &filetype == 'sh'  :!./%  endif  endfunc  "C,C++的调试  map <F8> :call Rungdb()<CR>  func! Rungdb()  exec "w"  exec "!g++ % -g -o %<"  exec "!gdb ./%<"  endfunc  """"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""  ""实用设置  """""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""  " 设置当文件被改动时自动载入  set autoread  " quickfix模式  autocmd FileType c,cpp map <buffer> <leader><space> :w<cr>:make<cr>  "代码补全  set completeopt=preview,menu  "允许插件  filetype plugin on  "共享剪贴板  set clipboard+=unnamed  "从不备份  set nobackup  "make 运行  :set makeprg=g++\ -Wall\ \ %  "自动保存  set autowrite  "set ruler " 打开状态栏标尺  set cursorline " 突出显示当前行  set magic " 设置魔术  set guioptions-=T " 隐藏工具栏  set guioptions-=m " 隐藏菜单栏  "set statusline=\ %<%F[%1\*%M%\*%n%R%H]%=\ %y\ %0(%{&fileformat}\ %{&encoding}\ %c:%l/%L%)\  " 设置在状态行显示的信息  set foldcolumn=0  set foldmethod=indent  set foldlevel=3  set foldenable " 开始折叠  " 不要使用vi的键盘模式，而是vim自己的  set nocompatible  " 语法高亮  set syntax=on  " 去掉输入错误的提示声音  set noeb  " 在处理未保存或只读文件的时候，弹出确认  set confirm  " 自动缩进  set autoindent  set cindent  " Tab键的宽度  set tabstop=4  " 统一缩进为4  set softtabstop=4  set shiftwidth=4  " 不要用空格代替制表符  set noexpandtab  " 在行和段开始处使用制表符  set smarttab  " 显示行号  set number  " 历史记录数  set history=1000  "禁止生成临时文件  set nobackup  set noswapfile  "搜索忽略大小写  set ignorecase  "搜索逐字符高亮  set hlsearch  set incsearch  "行内替换  set gdefault  "编码设置  set enc=utf-8  set fencs=utf-8,ucs-bom,shift-jis,gb18030,gbk,gb2312,cp936  "语言设置  set langmenu=zh\_CN.UTF-8  set helplang=cn  " 我的状态行显示的内容（包括文件类型和解码）  "set statusline=%F%m%r%h%w\ [FORMAT=%{&ff}]\ [TYPE=%Y]\ [POS=%l,%v][%p%%]\ %{strftime(\"%d/%m/%y\ -\ %H:%M\")}  "set statusline=[%F]%y%r%m%\*%=[Line:%l/%L,Column:%c][%p%%]  " 总是显示状态行  "set laststatus=2  " 命令行（在状态行下）的高度，默认为1，这里是2  set cmdheight=2  " 侦测文件类型  filetype on  " 载入文件类型插件  filetype plugin on  " 为特定文件类型载入相关缩进文件  filetype indent on  " 保存全局变量  set viminfo+=!  " 带有如下符号的单词不要被换行分割  set iskeyword+=\_,$,@,%,#,-  " 字符间插入的像素行数目  set linespace=0  " 增强模式中的命令行自动完成操作  set wildmenu  " 使回格键（backspace）正常处理indent, eol, start等  set backspace=2  " 允许backspace和光标键跨越行边界  set whichwrap+=<,>,h,l  " 可以在buffer的任何地方使用鼠标（类似office中在工作区双击鼠标定位）  set mouse=a  set selection=exclusive  set selectmode=mouse,key  " 通过使用: commands命令，告诉我们文件的哪一行被改变过  set report=0  " 在被分割的窗口间显示空白，便于阅读  set fillchars=vert:\ ,stl:\ ,stlnc:\  " 高亮显示匹配的括号  set showmatch  " 匹配括号高亮的时间（单位是十分之一秒）  set matchtime=1  " 光标移动到buffer的顶部和底部时保持3行距离  set scrolloff=3  " 为C程序提供自动缩进  set smartindent  " 高亮显示普通txt文件（需要txt.vim脚本）  au BufRead,BufNewFile \* setfiletype txt  "自动补全  :inoremap ( ()<ESC>i  :inoremap ) <c-r>=ClosePair(')')<CR>  :inoremap { {<CR>}<ESC>O  :inoremap } <c-r>=ClosePair('}')<CR>  :inoremap [ []<ESC>i  :inoremap ] <c-r>=ClosePair(']')<CR>  :inoremap " ""<ESC>i  :inoremap ' ''<ESC>i  function! ClosePair(char)  if getline('.')[col('.') - 1] == a:char  return "\<Right>"  else  return a:char  endif  endfunction  filetype plugin indent on  "打开文件类型检测, 加了这句才可以用智能补全  set completeopt=longest,menu  """""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""  " CTags的设定  """""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""  let Tlist\_Sort\_Type = "name" " 按照名称排序  let Tlist\_Use\_Right\_Window = 1 " 在右侧显示窗口  let Tlist\_Compart\_Format = 1 " 压缩方式  let Tlist\_Exist\_OnlyWindow = 1 " 如果只有一个buffer，kill窗口也kill掉buffer  let Tlist\_File\_Fold\_Auto\_Close = 0 " 不要关闭其他文件的tags  let Tlist\_Enable\_Fold\_Column = 0 " 不要显示折叠树  autocmd FileType java set tags+=D:\tools\java\tags  "autocmd FileType h,cpp,cc,c set tags+=D:\tools\cpp\tags  "let Tlist\_Show\_One\_File=1 "不同时显示多个文件的tag，只显示当前文件的  "设置tags  set tags=tags  "set autochdir  """"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""  "其他东东  """""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""  "默认打开Taglist  let Tlist\_Auto\_Open=1  """"""""""""""""""""""""""""""  " Tag list (ctags)  """"""""""""""""""""""""""""""""  let Tlist\_Ctags\_Cmd = '/usr/bin/ctags'  let Tlist\_Show\_One\_File = 1 "不同时显示多个文件的tag，只显示当前文件的  let Tlist\_Exit\_OnlyWindow = 1 "如果taglist窗口是最后一个窗口，则退出vim  let Tlist\_Use\_Right\_Window = 1 "在右侧窗口中显示taglist窗口  " minibufexpl插件的一般设置  let g:miniBufExplMapWindowNavVim = 1  let g:miniBufExplMapWindowNavArrows = 1  let g:miniBufExplMapCTabSwitchBufs = 1  let g:miniBufExplModSelTarget = 1 |

## 2 插件管理器Vundle

Vundle类似软件管家，管理vim中安装的插件

### 安装Vundle

A 下载安装

git clone [https://github.com/VundleVim/Vundle.vim.git ~/.vim/bundle/Vundle.vim](https://github.com/VundleVim/Vundle.vim.git%20%20~/.vim/bundle/Vundle.vim)

红色部分代表自己的本地安装位置

B 编辑~/.vimrc文件

|  |
| --- |
| .vimrc文件见.vimrc.txt文件 |

C 在vim的命令行模式下，执行：PluginInstall命令

此命令会安装在.vimrc中定义的插件

D :PluginInstall <plugin-name> 安装指定的插件。

### 清理插件

打开vim输入:PluginClean

从命令行安装vim +:PluginClean

### 查询插件

打开vim输入:PluginSearch <text-list>

从命令行安装vim +:PluginSearch <text-list>。

### 更新插件

在vim中输入:PluginUpdate

### 安装插件过程（以自动补全插件nerdtree为例）

|  |
| --- |
|  |

### 安装YouCompleteMe

|  |
| --- |
| git clone https://github.com/chxuan/vimplus.git ~/.vimplus  cd ~/.vimplus  ./install.sh |

## 3 Vim命令

### Vim命令模式下的使用

Vim 是vi的增强

Vimtutor 查看系统自带的vim教程

光标移动

使用hjkl四个键控制上下左右的移动

k

h lmax

j

0 行首

$行尾

gg 文件头

G 文件尾部

nG 指定光标到n行

vim删除内容

x 删除光标所在的字母

X 删除光标前的字母

dw 删除单词（从光标处开始删除）

d0 删除光标到行首

d$ (D) 删除光标到行尾

dd 删除光标所在行（内容放在剪切板中）

ndd 删除光标行开始n行（内容放在剪切板中）

u 撤销上部操作

ctrl +r 反撤销

vim复制粘贴

yy 复制一行内容

dd 剪切一行内容

nyy 复制多行内容（从光标所在行开始）

p/P 粘贴（注意：p是在光标下一行开始，P在光标所在行开始）

r 替换 输入r后再输入一个字母，可以将光标所在位置的字母替换

### vim可视模式

v进入可视模式

移动光标选中内容

按y复制内容

移动光标到目的地

按p/P粘贴内容

Vim查找

/str n/N遍历 n往下遍历 N往上遍历

?str N/n遍历 N往下遍历 n往上遍历

光标移动到查找的字符串上，按#可以查找光标选中的内容，按n/N上下移动

Vim调整文件格式

gg=G 整体调整格式

>>向右缩进tab

<<向左缩进tab

n>>当前行开始n行向右缩进一个tab

n<<当前行开始n行向左缩进一个tab

vim中man查看函数实现

光标放在函数上，按K进入man 1页

按n+K进入man n 页

一般的库函数都在2、3页

### Vim命令模式和编辑模式的转换

小I 在光标前插入

小a在光标后插入kkk

I在行首插入

A在行尾插入

小o 在光标上一行开始插入模式

O在光标下一行开始插入模式

小s删除当前光标所在的字母变成插入模式

S 删除当前行，进行插入模式

### Vim末行模式

末行模式中在屏幕最下方出现NORMAL字样

可以执行命令

:!ls –l

代码替换

一般对于函数名、变量等批量替换操作

:s/src/des 只替换光标所在行的第一个匹配的src为des

:s/src/des/g 只替换光标所在行的所有匹配的src为des

:%s/src/des 只替换所有行的第一个匹配的src为des

:%s/src/des/g替换文件内容中所有匹配的src为des

保存退出

:wq保存退出

:w 只保存

:q 只退出，文件不能修改

:q! 强制退出

:x 保存退出

ZZ 保存退出（在命令模式下）

分屏

:sp filename横向分屏

:vsp filename竖向分屏

:qall 全退不保存

:wqall全保存

多屏切换：ctrl +w+w

Vim安装插件

自行百度学习  
Vim配置

Cd



指令中使用vi的快捷键

设置 ~/.bashrc

添加 set -o vi ------可以直接使用vi的各种快捷键

## 4 使用

### 设置外部内容可以复制在vim中

 sudo apt-get install vim-gnome

“+y表示从vim复制到系统剪切版，

“+p表示从外部文件(系统剪切版)粘帖到vim

“+d表示剪切.

（如果在第一步时遇到vim-common的版本冲突问题，请参考Linux命令笔记->在ubuntu中更换apt源的方法）