**1、安装依赖件**

#vim要带python2.7的支持，curl是下载插件必须用到的软件，还有git

apt install python2.7 curl git

#安装python头文件

apt install python-dev python3-dev

#安装c/c++编译包

apt install build-essential

#安装cmake，编译YCM时候要用到。

apt install cmake

**2、下载Vundle到制定目录**

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

cd ~/.vim/bundle/YouCompleteMe

git submodule update --init --recursive

**3、vimrc 设置**

set paste

set nocompatible " be iMproved, required

filetype off " required

"设置Vundle的运行路径并初始化

set rtp+=~/.vim/bundle/Vundle.vim

call vundle#begin()

" Vundle安装位置与插件路径不同时，需要Vundle插件的路径

"call vundle#begin('~/some/path/here')

"------------------要安装的插件不能写在此行前！------------------

"Vundle对自己的调用，不可删去

Plugin 'VundleVim/Vundle.vim'

"以下是所支持的各种不同格式的示例

"需要安装的插件应写在调用的vundle#begin和vundle#end之间

"如果插件托管在Github上，写在下方，只写作者名/项目名就行了

Plugin 'Valloric/YouCompleteMe'

Plugin 'majutsushi/tagbar'

Plugin 'vim-syntastic/syntastic'

Plugin 'vim-airline/vim-airline-themes'

Plugin 'vim-airline/vim-airline'

"如果插件来自vim-scripts(官方)，写插件名就行了

" Plugin 'L9'

"如果Git仓库不在Github上，需要提供完整的链接

" Plugin 'git://git.wincent.com/command-t.git'

"本地的插件需要提供文件路径

" Plugin 'file:///home/gmarik/path/to/plugin'

"一定要确保插件就在提供路径的文件夹中(没有子文件夹，直接在这层目录下)

"运行时目录的路径

"Plugin 'rstacruz/sparkup', {'rtp': 'vim/'}

"避免插件间的命名冲突

"Plugin 'ascenator/L9', {'name': 'newL9'}

"------------------要安装的插件不能写在此行后！------------------

call vundle#end() " required

filetype plugin indent on " required

"要忽略插件缩进更改，请改用：

"filetype plugin on

"

" 安装插件的相关指令

":PluginList - 列出已安装插件

":PluginInstall - 安装新添加的插件;添加`!`或使用`:PluginUpdate`来更新已安装插件

":PluginSearch xxx - 寻找名字带有xxx的插件;添加`!`刷新本地缓存

":PluginClean - 删除已经从列表移除的插件;添加`!`静默卸载

":h - 帮助和说明文档

"Vundle的设置到此为止了

"YouCompleteMe 插件配置

let g:ycm\_global\_ycm\_extra\_conf='~/.vim/bundle/YouCompleteMe/third\_party/ycmd/cpp/ycm/.ycm\_extra\_conf.py'

nnoremap <leader>jd :YcmCompleter GoToDefinitionElseDeclaration<CR>

let g:ycm\_python\_binary\_path = '/usr/bin/python3'

nmap<C-a> :YcmCompleter FixIt<CR>

" 自动补全配置

set completeopt=longest,menu "让Vim的补全菜单行为与一般IDE一致(参考VimTip1228)

autocmd InsertLeave \* if pumvisible() == 0|pclose|endif "离开插入模式后自动关闭预览窗口

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

"上下左右键的行为 会显示其他信息

inoremap <expr> <Down> pumvisible() ? "\<C-n>" : "\<Down>"

inoremap <expr> <Up> pumvisible() ? "\<C-p>" : "\<Up>"

inoremap <expr> <PageDown> pumvisible() ? "\<PageDown>\<C-p>\<C-n>" : "\<PageDown>"

inoremap <expr> <PageUp> pumvisible() ? "\<PageUp>\<C-p>\<C-n>" : "\<PageUp>"

"youcompleteme 默认tab s-tab 和自动补全冲突

"let g:ycm\_key\_list\_select\_completion=['<c-n>']

let g:ycm\_key\_list\_select\_completion = ['<Down>']

"let g:ycm\_key\_list\_previous\_completion=['<c-p>']

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

let g:ycm\_confirm\_extra\_conf=0 "关闭加载.ycm\_extra\_conf.py提示

let g:ycm\_collect\_identifiers\_from\_tags\_files=1 " 开启 YCM 基于标签引擎

let g:ycm\_min\_num\_of\_chars\_for\_completion=2 " 从第2个键入字符就开始罗列匹配项

let g:ycm\_cache\_omnifunc=0 " 禁止缓存匹配项,每次都重新生成匹配项

let g:ycm\_seed\_identifiers\_with\_syntax=1 " 语法关键字补全

nnoremap <F5> :YcmForceCompileAndDiagnostics<CR> "force recomile with syntastic

"nnoremap <leader>lo :lopen<CR> "open locationlist

"nnoremap <leader>lc :lclose<CR> "close locationlist

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

"在注释输入中也能补全

let g:ycm\_complete\_in\_comments = 1

"在字符串输入中也能补全

let g:ycm\_complete\_in\_strings = 1

"注释和字符串中的文字也会被收入补全

let g:ycm\_collect\_identifiers\_from\_comments\_and\_strings = 0

nnoremap <leader>jd :YcmCompleter GoToDefinitionElseDeclaration<CR> " 跳转到定义处

""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""

" 显示相关

""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""

"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

"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

""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""

"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

**4、安装插件**

打开vim，在运行 :PluginInstall

YCM不同于以往其他vim插件，YCM是一款编译型的插件。安装过程种会自动下载指定版本的clang，

如果有冲突，需要卸载之前安装的clang，再进行安装。在下载完后，需要手动编译后才能使用。

cd ~/.vim/bundle/YouCompleteMe

./install.py --clang-completer

如果不需要c-family的补全，可以去掉--clang-completer。如果需要c#的补全，请加上--omnisharp-completer。  
正常来说,YCM会去下载clang的包，如果已经有，也可以用系统--system-libclang。  
就这样，安装结束。打开vim，如果没有提示YCM未编译，则说明安装已经成功了。

不同于很多vim插件，YCM首先需要编译，另外还需要有配置。在vim启动后，YCM会找寻当前路径以及上层路径的.ycm\_extra\_conf.py.在~/.vim/bundle/YouCompleteMe/third\_party/ycmd/cpp/ycm/.ycm\_extra\_conf.py中提供了默认的模板。将该文件拷贝到~/.vim/bundle/YouCompleteMe目录

配置.ycm\_extra\_conf.py

flags = [

'-Wall',

'-Wextra',

'-Werror',

'-Wno-long-long',

'-Wno-variadic-macros',

'-fexceptions',

'-DNDEBUG',

'-DUSE\_CLANG\_COMPLETER',

'-x',

'c',

'c++',

'-isystem',

'../BoostParts',

'-isystem',

get\_python\_inc(),

'-isystem',

'/usr/include',

'-isystem',

'/usr/include/c++/7',

'-isystem',

'/usr/include/c++/7.2.0',

'-isystem',

'/usr/include',

'-isystem',

'/usr/include/x86\_64-linux-gnu/\*',

'-isystem',

'/usr/include/\*.h',

]