

提高开发效率——5步打造vim IDE

文中提到的所有依赖插件ctags、cscope、配置文件，包括vim都给出了安装文件，就如公司说的，提供一站式服务，呵呵。

1: test.cpp [7: cscope.out] [8: tags] 当前打开的文件索引

MiniBufExplorer-

```

MAX_RICH_INTR 40
MAX_RICH_SHOR 41 #pragma pack()
MAX_RCD_REASO 42 int main(int argc, char *argv[])
MAX_TOUIN_NUM 43 {
44     //char s[60]={1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16};
typedef 45 char s[80];
RcdImgInfo 46 strcpy(s, "012345678");
RcdInfo 47 //char *src=s;
48 //char *dst=&s[1];
function 49
main 50 printf("first buf is:%s\n", s);
51 strcpy(s+2, s);
52
53 printf("second buf is:%s\n", s);
54 printf("sizeof(RcdInfo)=%d\n", sizeof(RcdInfo));
55 // for(int i=0; i<10; i++)
56 // {
57 //     printf("src[%d]=%d, dst[%d]=%d\n", i, src[i], i, dst[i]);
58 // }
59 return 0;
60 }

```

宏、类型、函数接口展示区域

代码编辑区域

1, 1 全部

```

4 "= /data/home/ju
5 ../
6 cscope.in.out
7 cscope.out
8 cscope.po.out
9 tags
10 test
11 test.cpp
12 .test.cpp.swp

```

当前路径下的目录、文件

<g List 18, 5 底端 test.cpp 59, 5 底端 <le List 11, 1 底端

```

1 test.cpp [33] <<global>> }RcdInfo;
2 test.cpp [54] <<main>> printf("sizeof(RcdInfo)=%d\n", sizeof(RcdInfo));

```

搜索结果、make输出结果区域

[Quickfix 列表] :cs find s RcdInfo

:call <SNR>15_EditEntry(0, "winmanager")

搭建步骤：

- 1、首先确认vim 的版本号是否大于7.3且是巨型版本，如下图：

```
juneliang@suse_64_dev173:~> vim --version
VIM - Vi IMproved 7.3 (2010 Aug 15, compiled May 16 2013 19:10:11)
  编译器: juneliang@suse_64_dev173
  巨型版本 无图形界面。 可使用(+)与不可使用(-)的功能:
  +arabic +autocmd -balloon_eval -browse +builtin_terms +byte_offset +cinde
```

注意：因为我们的开发编译，都没有root权限，所以需要将vim安装在自己的home目录下，例如~/local/ ；另外需要选择巨型版本。

```
./configure --prefix=~/.local/ --with-features=huge
```

配置完后，直接make;make install

再在主目录下文件中.bashrc中添加一行

```
export PATH="$HOME/local/bin:$PATH"
```

这样以后每次登陆使用的vim就是自己安装的了。

2、安装ctags，只需配置一下安装路径： ./configure --prefix=~ /local/ ; 然后make ;make install

3、安装cscope，同样只需配置一下安装路径： ./configure --prefix=~ /local/ ; 然后make ;make install

4、现在所需工具安装OK后，还需要vim把这些工具集成起来：

把附件中的vim.tar 在 ~ 目录下解压

把附件中的 .vimrc 拷贝到 ~ 目录下。

5、进入工程目录，运行以下两个命令

```
ctags -R
```

```
cscope -Rbq
```

，然后通过vim打开文件，分别输入 tl 、 wm ， 就可以看到vim IDE 开发环境了。

注意： **1)、必须从ctags或cscope执行的目录，用vim打开文件；**

2)、工程目录下每次添加或更改原代码，都需要重新运行以上两个命令。

附：常用命令

A、vim中的常用命令：

;gt 跳转到光标所在关键词的定义处（用户函数、类型、宏、全局变量）

;gr 跳回原光标处

;lg 跳转到当前行的行首

;le 跳转到当前行的行尾

tl 打开/关闭taglist

wm 打开/关闭winmanger

;cw 打开/关闭结果区域quickfix

;cd 在C 和 H 文件之间 切换

;gs 在多个窗口之间逆时针切换

;w 保存文件

;m 执行make命令

;y 将选中的文本拷贝到缓冲区

;p 将缓冲区拷贝到vim

;q 退出当前窗口

注意，以上命令都是在vim 命令模式下、顺序、连续单击对应字符键操作完成。;作为前缀符号，在文件 .vimrc 定义。当然你也可以重新定义、使用其他字符作为快捷键。

```
* 定义快捷键的前缀，即<Leader>
let mapleader=";"

* 定义快捷键 跳转到当前行的行首
nmap lg 0
* 定义快捷键 跳转到当前行的行尾
nmap le $
* 定义快捷键 关闭当前分割窗口
nmap <Leader>q :q<CR>
* 定义快捷键 保持当前窗口内容
nmap <Leader>w :w<CR>
* 设置快捷键 将选中文本块复制至系统剪贴板
vnoremap <Leader>y "+y
* 设置快捷键 将系统剪贴板内容粘贴至vim
nmap <Leader>p "+p
* 设置快捷键 进行工程编译及链接，并同时在quickfix输出make结果
* 前提是工程目录中有Makefile文件
nmap <Leader>m :wa<CR>:make<CR>:cw<CR><CR>

* 定义快捷键 跳转到光标所在关键词的定义处
nmap <Leader>gt <C-]>
* 定义快捷键 跳回原关键词 与 :gr 配合使用
nmap <Leader>gr <C-T>
* 定义快捷键 跳到当前屏幕倒数第二行
nmap <Leader>gf <C-f>
* 定义快捷键 跳到当前屏幕第二行
nmap <Leader>gb <C-b>
*快速切换C H源文件
nmap <Leader>cd :A<CR>
* 使用Grep.vim插件在工程内全局查找，设置快捷键。快捷键速记法：search in project
nnoremap <Leader>sp :Grep<CR>
* 设置快捷键gs遍历各分割窗口。快捷键速记法：goto the next spilt window
nnoremap <Leader>gs <C-W><C-W>
```

```
" 定义快捷键 打开/关闭 tag list
nmap tl :TlistToggle<CR>
```

```
" 定义快捷键 打开/关闭 winmanger
nmap wm :WMToggle<cr>
```

B、在source insight中，有一个重要的功能：可以搜索关键词在哪些文件哪些行出现过；

在vim中，也可以实现：将光标移到所要搜索的词上，先同时按住 ctrl+shift+'_', 再单击's' 键，则会在quickfix窗口出现匹配到的结果（如果quickfix区域没有出现，则按顺序单击 ;cw ，弹出quickfix窗口）

C、如果只想在某个指定文件搜索关键词出现的行，则直接在vim命令模式下输入 :vimgrep 关键词 指定文件

即可。

D 、 vim 常用命令：

Ctrl +w l 跳转到右边窗口

Ctrl +w h 跳转到左边窗口

Ctrl +w j 跳转到下边窗口

Ctrl +w k 跳转到上边窗口

都是先同时按住ctrl 和 w 键，再按 j键（k、l、h）

zo 将光标所在的折叠区域打开

zc 将光标所在的行对应的区域折叠起来

当然，这只是vim强大功能中很小很小的一部分，作为我平常的程序开发，已经足够；如果还想更深入了解各个工具集合，推荐看看这篇文章《手把手教你配置VIM作为IDE》。