

# Vim项目目录下

Apache

```
1 cmake -DCMAKE_EXPORT_COMPILE_COMMANDS=1 ../..
```

[https://gitee.com/CloudGuan/nvim-for-server?\\_from=gitee\\_search](https://gitee.com/CloudGuan/nvim-for-server?_from=gitee_search)

目 VIM操作

## 指令

### · ctrl+p

作用：Leaderf文件搜索，相当于小番茄的alt+shift+o

位置：nvim/setting/leaderf.vim

技巧：

如果新加的文件搜不到，按F5

ctrl+j/k 选择文件后ctrl+T可以在新页签打开.可以自己映射其他快捷键在多页签切换，比如我的：

HTML

```
1 " 前一个 后一个 tab
2 noremap <C-h> :tabp <cr>
3 noremap <C-l> :tabn <cr>
```

### · <leader>+p (,p)

作用：列出当前文件的函数，相当于vs的alt+m

位置：nvim/setting/leaderf.vim

### · gd,gr

作用：gd跳转至定义 gr列出引用

位置：nvim/setting/cocconf.vim

### · Ctrl + o ， Ctrl + I

作用：往后/前跳转，

- F2

作用：打开目录

位置: nvim/setting/defx.vim

目录和代码切换的快捷键命令: jk

ctrl+w+h:回到目录界面

ctrl+w+l:回到右边的代码界面

## 函数参数补全

输入函数前几个字符时，下面会列出提示，ctrl+N/P可以上下移动选择，选中之后按ctrl+y可以确认选择并且补全函数参数，还会包含相应头文件.可以用于：在头文件定义了函数，要到cpp实现函数的时候使用。

```
void Role::GetRolesByReleation()
void Role: GetRoleAllInfo() const~ f [LS] bool
{ GetRoleAllInfoPtr()~ f [LS]
  Set<Ro GetRoleByObject(R0GameLibs::R0Object *object)~ f [LS]
  Set<Ro GetRoleID() const~ f [LS]
  Set<Ro GetRoleLifeSkill()~ f [LS]
  Set<Ba GetRolesByReleation( RoleReleationType releation, std::set<uint64_t> &releation roles, uint64_t &brief uid) const~ f [LS]
  Set<Ro GetGameRole(...)~ f [LS]
  Set<Ro GetRole() const~ f [LS]
  Set<Ro GetTeamByRoleUUID [A]
  Set<Ro GetMemberByRoleUUID [A]
  Set<RoleOpenSystem>(NEW_SERVER_OBJECT(RoleOpenSystem, this));
```

## 插件

需要加在~/.config/nvim/init.vim中

如果在编辑模式添加，关闭文件后又会失效

- DoxygenToolkit

作用：自动生成注释

添加: `Plug 'vim-scripts/DoxygenToolkit.vim'`

使用：光标移至函数上一行，命令模式下输入 `:Dox`

附件：

 Doxyten

- Gtags

作用：代码引用搜索

添加: `Plug 'gtags.vim'`

使用：参考<https://www.gnu.org/software/global/>教程

附件: <https://www.gnu.org/software/global/>

## • grep.vim

作用: 关键字快速搜索

添加: `Plug 'grep.vim'`

使用: 命令模式:Rgrep

附件: [https://www.vim.org/scripts/script.php?script\\_id=311](https://www.vim.org/scripts/script.php?script_id=311)

## • asynchrn.vim

作用: 异步代码执行这个

添加: `Plug 'skywind3000/asynchrn.vim'`

使用: 命令模式:asynchrn xxx

附件: [https://www.vim.org/scripts/script.php?script\\_id=311](https://www.vim.org/scripts/script.php?script_id=311)

## 常用插件地址

<https://github.com/vim-scripts/DoxygenToolkit.vim>

<https://github.com/universal-ctags/ctags>

<https://ftp.gnu.org/pub/gnu/global/global-6.6.tar.gz>

## 常用脚本

当clone的时候有指定分支, 需要新拉一个已存在的分支时:

两个参数, 已在在的分支名和拉的depth. 在工程根目录执行

## Bash

```
1  #!/bin/bash
2
3  if (( $# < 2 )); then
4      echo "需要两个参数: 分支名和日志深度"
5      exit 1
6  fi
7
8  branch=${1}
9  depth=${2}
10
11
12  cat .git/config | grep "refs/heads/${branch}:refs/remotes/origin/${branch}"
13  >/dev/null
14  if (( $? > 0 )); then #说明没添加
15      git remote set-branches --add origin ${branch}
16  fi
17
18  git branch -r | grep origin/${branch} > /dev/null
19  if (( $? > 0 )); then # 说明没有拉下来
20      git fetch origin ${branch}:${branch} --depth ${depth}
21      if (( $? > 0 )); then
22          echo "git fetch 失败, 请手动重试"
23          exit 1
24      fi
25  fi
26
27  git checkout ${branch}
28  if (( $? > 0 )); then
29      echo "git checkout ${branch} 失败, 请手动重试"
30      exit 1
31  fi
32
33  git push -u origin ${branch}
34
35  echo "Done"
```

修改东西太多, 想要一个个提交确认:

空格表示要提交, 回车表示跳过

## Bash

```
1  files=$(git status | grep -v "# On branch" | grep -v "\"git\"")
2
3  # 最终要add的文件
4  final_add=""
```

```
5 final_rem=""
6 function try_read()
7 {
8     if [[ $# < 2 ]]; then
9         return
10    fi
11    while read -n1 rlt; do
12        if [[ -z ${rlt}
13            || ${rlt} == ' ' ]]; then
14            break;
15        fi
16    done
17    if [[ ${rlt} == ' ' ]]; then
18        if (( ${2} == 1 )); then
19            final_rem="${final_rem} ${1}"
20        else
21            final_add="${final_add} ${1}"
22        fi
23    fi
24 }
25
26 IFS=$'\n'
27 not_stage_str="# Changes not staged for commit:"
28 untrack_str="# Untracked files:"
29 # 过滤掉已add过的内容
30 for fd in ${files[@]}; do
31     if [[ ${fd} == ${not_stage_str}
32         || ${fd} == ${untrack_str} ]]; then
33         break
34     fi
35 done
36
37 # 判断是否无视的文件
38 ignore_files_reg=(^.cache/
39     ^G
40     ^rogamelib
41     ^lib/
42     ^vimjson/
43     ^buildtool
44     ^tscancode
45 )
46
47 ignore_files=("$(
48     "gitadd.sh"
49     "compile_commands.json"
50     "add_exist_branch.sh"
51 )
```

```
52 function CheckIgnore()
53 {
54     if [[ $# < 1 ]]; then
55         return 1
56     fi
57     for ig in ${ignore_files[@]}; do
58         if [[ ${1} == ${ig} ]]; then
59             return 1
60         fi
61     done
62
63     for ig in ${ignore_files_reg[@]}; do
64         if [[ ${1} =~ ${ig} ]]; then
65             return 1
66         fi
67     done
68     return 0
69 }
70
71 # 处理修改中的内容
72 modify=0
73 modify_reg="^#.*modified:*"
74 delete_reg="^#.*deleted:*"
75 for fd in ${files[@]}; do
76     if [[ ${fd} == ${untrack_str} ]]; then
77         break;
78     fi
79     if [[ ${fd} == ${not_stage_str} ]]; then
80         modify=1
81         continue
82     fi
83     if (( modify == 1 )); then
84         if [[ ${fd} == "#" ]]; then
85             continue
86         fi
87         rfd=$(echo ${fd} | awk '{print $3}')
88         CheckIgnore ${rfd}
89         if (( $? != 0 )); then
90             continue
91         fi
92         if [[ ${fd} =~ ${modify_reg} ]]; then
93             echo "修改的文件: ${rfd}"
94             try_read ${rfd} 0
95         elif [[ ${fd} =~ ${delete_reg} ]]; then
96             echo "删除的文件: ${rfd}"
97             try_read ${rfd} 1
98         else
99             echo "未知类型文件: ${fd}"
```

```
100         fi
101     fi
102 done
103
104 # 未追踪的文件(夹)
105 untrack=0
106 for fd in ${files[@]}; do
107     if [[ ${fd} == ${untrack_str} ]]; then
108         untrack=1
109         continue
110     fi
111     if (( ${untrack} == 1 )); then
112         if [[ ${fd} == "#" ]]; then
113             continue
114         fi
115         rfd=$(echo ${fd} | awk '{print $2}')
116         CheckIgnore ${rfd}
117         if (( $? != 0 )); then
118             continue
119         fi
120         echo "未添加文件(夹): ${rfd}"
121         try_read ${rfd} 0
122     fi
123 done
124
125 IFS=$' '
126 if (( ${#final_add} > 0 )); then
127     git add ${final_add[@]}
128 else
129     echo "没有要提交的"
130 fi
131 if (( ${#final_rem} > 0 )); then
132     git rm ${final_rem[@]}
133 else
134     echo "没有要删除的"
135 fi
136 echo "-----"
137
138 git status
```