**[ctags的使用脚本](http://blog.csdn.net/peng_cao/article/details/48522383)**

**使用ctags生成tags**

**通用的ctags使用方法**

很多博文中都只是提供了使用ctags直接生成的tags方法。但是博主在使用过程中发现有诸多不便。例如不能很好的过滤某些目录，指定某些目录等。常见使用如下命令   
ctags -R --languages=c++,c,[**Java**](http://lib.csdn.net/base/17),Make,Sh, --c++-kinds=+px --fields=+aiKSz --extra=+q

* --languages=c++,c,java,Make,Sh, 就是制定了本次ctags扫描的语言是什么
* --c++-kinds=+px 指示ctags扫描一些函数声明

**高效的ctags使用方法**

这里使用到了ctags的**-L**参数，可以实现扫描指定目录下的指定文件！

#!/bin/sh

OUT\_PATH=${PWD}

SRC\_PATH=${PWD}

if [ -d ${OUT\_PATH} ];

then

echo "Use current directory \"${OUT\_PATH}\""

else

echo "mkdir \"${OUT\_PATH}\""

mkdir ${OUT\_PATH}

fi

if [ -d ${SRC\_PATH}/frameworks -a -d ${SRC\_PATH}/packages -a -d ${SRC\_PATH}/hardware -a -d ${SRC\_PATH}/system ];

then

echo "Found framework packages hardware and system directory!!"

find -L ${SRC\_PATH}/frameworks -name "\*.h" -o -name "\*.cpp" -o -name "\*.cc" -o -name "\*.c" -o -name "\*.java"|grep -vi '\.git' >${OUT\_PATH}/names.file

echo "framework scann finish!"

find -L ${SRC\_PATH}/packages -name "\*.h" -o -name "\*.cpp" -o -name "\*.cc" -o -name "\*.c" -o -name "\*.java"|grep -vi '\.git' >>${OUT\_PATH}/names.file

echo "packages scann finish!!"

find -L ${SRC\_PATH}/hardware -name "\*.h" -o -name "\*.cpp" -o -name "\*.cc" -o -name "\*.c" -o -name "\*.java"|grep -vi '\.git' >>${OUT\_PATH}/names.file

echo "hardware scann finish!!"

find -L ${SRC\_PATH}/system -name "\*.h" -o -name "\*.cpp" -o -name "\*.cc" -o -name "\*.c" -o -name "\*.java"|grep -vi '\.git' >>${OUT\_PATH}/names.file

echo "sytstem scann finish!!"

else

echo "Please run this script under the root of your project!!!"

fi

echo "For create ctags databases.."

ctags -L ${OUT\_PATH}/names.file -f ${OUT\_PATH}/tags

echo "DONE!!"

**注意事项**

生成**tags**之后，会在当前目录(例如/a/b/c)有一个tags的文件生成，那么我们在使用vim打开文件的时候，需要首先进入到当前目录(/a/b/c)这样就可以了

**不同文件之间的转跳**

这里先介绍两个最常用的，后面更高级的稍后再介绍,**这里所有的转跳操作需要将编标移动到对应位置**

* Ctrl +] 转跳到调用的当前函数的实现部分
* Ctrl +t 跳转回上一步