# 前言

# vi使用指南

## vi的工作模式

* 一般模式： vi 处理文件时，一进入该文件，就是一般模式了。
* 编辑模式：在一般模式下可以进行删除，复制，粘贴等操作，却无法进行编辑操作。等按下“i，I，o，O，a，A，r，R”等字母之后才能进入编辑模式。通常在linux中，按下上述字母时，左下方会出现'INSERT'或者‘REPLACE’字样,才可以输入任何文字到文件中.要回到一般模式，按下[ESC]键即可.
* 命令行模式：在一般模式中，输入“: 或者/或者?”,即可将光标移动到最下面一行，在该模式下，您可以搜索数据，而且读取，存盘，大量删除字符，离开vi，显示行号等操作。

## vi常用命令

进入vi的命令

vi filename：打开或新建文件，并将光标置于第一行首

vi +n filename：打开文件，并将光标置于第n行首

vi + filename：打开文件，并将光标置于最后一行首

vi +/pattern filename：打开文件，并将光标置于第一个与pattern匹配的串处

vi -r filename：在上次正用vi编辑时发生系统崩溃，恢复filename

vi filename....filename：打开多个文件，依次进行编辑

移动光标类命令

h：光标左移一个字符

l：光标右移一个字符

space：光标右移一个字符

Backspace：光标左移一个字符

k或Ctrl+p：光标上移一行

j或Ctrl+n ：光标下移一行

Enter：光标下移一行

w或W：光标右移一个字至字首

b或B：光标左移一个字至字首

e或E：光标右移一个字至字尾

)：光标移至句尾

(：光标移至句首

}：光标移至段落开头

{：光标移至段落结尾

nG：光标移至第n行首

n+：光标下移n行

n-：光标上移n行

n$：光标移至第n行尾

H：光标移至屏幕顶行

M：光标移至屏幕中间行

L：光标移至屏幕最后行

0：（注意是数字零）光标移至当前行首

$：光标移至当前行尾

屏幕翻滚类命令

Ctrl+u：向文件首翻半屏

Ctrl+d：向文件尾翻半屏

Ctrl+f：向文件尾翻一屏

Ctrl＋b：向文件首翻一屏

nz：将第n行滚至屏幕顶部，不指定n时将当前行滚至屏幕顶部。

插入文本类命令

i：在光标前

I：在当前行首

a：光标后

A：在当前行尾

o：在当前行之下新开一行

O：在当前行之上新开一行

r：替换当前字符

R：替换当前字符及其后的字符，直至按ESC键

s：从当前光标位置处开始，以输入的文本替代指定数目的字符

S：删除指定数目的行，并以所输入文本代替之

ncw或nCW：修改指定数目的字

nCC：修改指定数目的行

删除命令

ndw或ndW：删除光标处开始及其后的n-1个字

do：删至行首

d$：删至行尾

ndd：删除当前行及其后n-1行

x或X：删除一个字符，x删除光标后的，而X删除光标前的

Ctrl+u：删除输入方式下所输入的文本

搜索及替换命令

/pattern：从光标开始处向文件尾搜索pattern

?pattern：从光标开始处向文件首搜索pattern

n：在同一方向重复上一次搜索命令

N：在反方向上重复上一次搜索命令

：s/p1/p2/g：将当前行中所有p1均用p2替代

：n1,n2s/p1/p2/g：将第n1至n2行中所有p1均用p2替代

：g/p1/s//p2/g：将文件中所有p1均用p2替换

s选项设置

all：列出所有选项设置情况

term：设置终端类型

ignorance：在搜索中忽略大小写

list：显示制表位(Ctrl+I)和行尾标志（$)

number：显示行号

report：显示由面向行的命令修改过的数目

terse：显示简短的警告信息

warn：在转到别的文件时若没保存当前文件则显示NO write信息

nomagic：允许在搜索模式中，使用前面不带“\”的特殊字符

nowrapscan：禁止vi在搜索到达文件两端时，又从另一端开始

mesg：允许vi显示其他用户用write写到自己终端上的信息

最后行方式命令

：n1,n2 co n3：将n1行到n2行之间的内容拷贝到第n3行下

：n1,n2 m n3：将n1行到n2行之间的内容移至到第n3行下

：n1,n2 d ：将n1行到n2行之间的内容删除

：w ：保存当前文件

：e filename：打开文件filename进行编辑

：x：保存当前文件并退出

：q：退出vi

：q!：不保存文件并退出vi

：!command：执行shell命令command

：n1,n2 w!command：将文件中n1行至n2行的内容作为command的输入并执行之，若不指定n1，n2，则表示将整个文件内容作为command的输入

：r!command：将命令command的输出结果放到当前行

寄存器操作

"?nyy：将当前行及其下n行的内容保存到寄存器？中，其中?为一个字母，n为一个数字

"?nyw：将当前行及其下n个字保存到寄存器？中，其中?为一个字母，n为一个数字   
"?nyl：将当前行及其下n个字符保存到寄存器？中，其中?为一个字母，n为一个数字

"?p：取出寄存器？中的内容并将其放到光标位置处。这里？可以是一个字母，也可以是一个数字

ndd：将当前行及其下共n行文本删除，并将所删内容放到1号删除寄存器中。

## Makefile例子

## Other defines

DEFINES = -DNDEBUG

ORACLE\_HOME = /home/oracle/app/oracle/product/11.2.0/dbhome\_1

ORACLE\_INCPATH = $(ORACLE\_HOME)/rdbms/public

ORACLE\_LIBPATH = $(ORACLE\_HOME)/lib

CC = g++

AR = ar

CP = cp

RM = rm

CXXFLAGS = -I$(ORACLE\_INCPATH) -O2 -Wall $(DEFINES)

LIBS = -L$(ORACLE\_LIBPATH) -lclntsh

SRCS = VerifyServer.cpp ConfigFile.cpp PublicFunc.cpp Log.cpp

OBJS = VerifyServer.o ConfigFile.o PublicFunc.o Log.o

BINS = VerifyServer

all: \

PublicFunc \

ConfigFile \

Log \

VerifyServer \

bin

clean:

$(RM) -f \*.o $(BINS)

bin:

$(CC) $(CXXFLAGS) -o $(BINS) $(OBJS) $(LIBS)

VerifyServer:

$(CC) -c $(CXXFLAGS) VerifyServer.cpp

ConfigFile:

$(CC) -c $(CXXFLAGS) ConfigFile.cpp

PublicFunc:

$(CC) -c $(CXXFLAGS) PublicFunc.cpp

Log:

$(CC) -c $(CXXFLAGS) Log.cpp

xmlParser: