layout: post
title: "vim实用技巧"
date: 2020-05-21
description: "linux使用"

tag: linux使用

Vim与系统剪切板交互

使用前，使用如下命令，看看自己的计算机的vim版本

```
vim --version | grep clipboard
```

如果出现“-clipboard”则说明系统的vim版本不支持与系统剪贴板的交互操作，需要采用如下的命令安装新的vim

```
sudo apt install vim-gtk
```

安装完成之后再利用代码检查一次，出现“+clipboard”，那么系统的vim没有问题。

```
vim --version | grep clipboard
```

在 vim中复制的代码或者文字，存储在寄存器（register）中，寄存器有好多个,使用如下vim命令查看系统vim的寄存器的个数

```
: help  registers
```

输出结果为

```
There are ten types of registers:           *registers* *E354*
1. The unnamed register ""
2. 10 numbered registers "0 to "9
3. The small delete register "-
4. 26 named registers "a to "z or "A to "Z
5. three read-only registers ":, "., "%
6. alternate buffer register "#
7. the expression register "=
8. The selection and drop registers "*, "+ and "~
9. The black hole register "_
10. Last search pattern register "/
```

1. 无名（unnamed）寄存器：""，缓存最后一次操作内容；
2. 数字（numbered）寄存器："0 ～"9，缓存最近操作内容，复制与删除有别, "0寄存器缓存最近一次复制的内容，"1-"9缓存最近9次删除内容
3. 行内删除（small delete）寄存器："-，缓存行内删除内容；
4. 具名（named）寄存器："a ～ "z或"A - "0Z，指定时可用；
5. 只读（read-only）寄存器：":,".,"%,"#，分别缓存最近命令、最近插入文本、当前文件名、当前交替文件名；
6. 表达式（expression）寄存器："=，只读，用于执行表达式命令；
7. 选择及拖拽（selection and drop）寄存器："*,"+,"~，存取GUI选择文本，可用于与外部应用交互，使用前提为系统剪切板（clipboard）可用；
8. 黑洞（black hole）寄存器："_，不缓存操作内容（干净删除）；

9. 模式寄存器（last search pattern）："/，缓存最近的搜索模式。
   从上述寄存器列表，可以知道，利用平时的yy，nyy等命令复制或者使用其他删除等命令操作的结果存储在unname 寄存器中，如果想要与系统的剪切板进行交互，就需要利用 "+ 这个寄存器，使用如下命令：

```
#将vim中的代码或者文字复制到剪切板
"+yy
"+nyy
#将系统剪切板中的代码或者文字复制到vim中
"+p
"+P
```

Vim基本命令

vim有三种工作模式：普通模式、插入模式、命令模式

三种工作模式的切换

```
#进入vim普通模式
vim filname

#退出vim的普通模式的两种方式
ZZ
ZQ

普通模式进入插入模式，按下字母i、a、o、I、A、O

插入模式进入普通模式，按下Esc键

普通模式进入命令模式，先输入冒号：，然后输入相应的命令以回车结束

光标移动
h  光标左移
l  光标右移
j  光标下移
k  光标上移
$  光标行尾
0(数字0)  光标行首
G  最后一行
gg  第一行
nG  第n行
```

下边这两个我自己不经常使用

```
w  后一词词首
e  后一词词尾
```

复制

```
yy   复制所在行
nyy 复制所在行向下n行
```

删除

```
dd  删除当前行
ndd  删除光标一下n行
d1G 删除光标到第一行
dG  删除光标到最后一行
d$  删除光标到行尾
d0  删除光标到行首
X  删除光标后一个字符
nX  删除光标后n个字符
```

粘贴

```
p（小写） 光标的下一行粘贴
P（大写） 光标的上一行粘贴
```

撤销

`u`

替换

```
r  替换光标所在处字符
R  进入替换模式  Esc退出
cc   替换光标所在行
```

查找替换
在命令模式下

```
: %s/旧文本/新文本/g     全局替换
: s/旧文本/新文本/g    可视区域替换
:%s/旧文本/新文本/gc    确认替换
```

光标页

```
ctrl + f 向下翻页
       b  向上翻页
       d  向下翻半页
       u  向上翻半页
H    当前页的第一行
M   当前页的中间行
L   当前页的最后一行
```

分屏命令
在命令模式下

```
:sp   filename 横向分屏
:vsp   filename  垂直分屏
ctrl + w   切换下一窗口
       r   互换窗口
       c   关闭窗口
       q   退出窗口
       o   关闭其他窗口
```

其他常用命令

```
:e  文件名  当前文件需保存
:n 文件名  新建文件
:w  文件名  另存为，但依然编辑当前文件
```

Vim的基本使用方法

1、显示行号

利用如下命令打开文件，如果没有就创建文件

```
vim /etc/vim/vimrc
#在文件的最后添加如下的代码
set number
#保存退出即可
```

2、自动缩进

`vim /etc/vim/vimrc`

#添加如下的代码

```
set tabstop=4 "表示Tab代表4个空格的宽度
set expandtab "表示Tab自动转换成空格
set autoindent "表示换行后自动缩进
```

3、语法高亮

```
vim /etc/vim/vimrc
#添加如下的代码
syntax enable
syntax on " 自动语法高亮
```

4、代码批量注释

```
1、利用ctrl+v 进入可视块模式
2、利用j k上下选择所需注释的代码
3、然后利用大写的字母 I 进如编辑模式\
4、然后加上注释#
5、然后双击esc即可实现代码批量注释
```

第二种批量注释的方式

```
1、利用V进去可视块模式，选中需要注释的代码
2、输入命令
:nomal i #
```

5、批量取消代码的注释

```
1、利用ctrl+v进入可视块模式

2、选中需要删除的代码行首的注释（如#）

3、按字母d
```

6、python自动缩进

```
set filetype=python

au BufNewFile,BufRead *.py,*.pyw setf python

set guifont=Consolas:h12:cANSI"英文字体
set guifontwide=SimSun-ExtB:h12:cGB2312
set tabstop=4"表示Tab代表4个空格的宽度
set expandtab"表示Tab自动转换成空格
set autoindent"表示换行后自动缩进
set autoread " 当文件在外部被修改时，自动重新读取
set history=400"vim记住的历史操作的数量，默认的是20
set nocompatible"使用vim自己的键盘模式,而不是兼容vi的模式
set confirm"处理未保存或者只读文件时,给出提示
set smartindent"智能对齐
set shiftwidth=4
```

7、插入空行

利用字母小写的o

8、复制删除一块代码

1、利用大写的字母V进入可视块模式

2、利用j、k上下选中一块代码

3、利用y复制代码，利用d删除代码

9、批量缩进代码

方法1（最简单）

1. 利用shift+v进入可视行模式
2. j、k上下选择需要缩进的行
3. 点击按键 >进行缩进，< 取消缩进

方法2

1. 利用ctrl+v 或者V进入可视块或者可视行模式
2. j、k上下选择需要缩进的代码段
3. 按下大写字母 I
4. 在代码首部输入所需缩进的空格数
5. 双击esc

跳转至标签分类页