Alias

Linux alias命令用来设置指令的别名，对一些较长的命令进行简化。使用alias时，必须使用单引号将原来的命令包含，防止特殊字符导致错误。

1.设置别名

alias 别名='原命令 -选项/参数'

例如:

alias ll='ls -lt'

这样设置了ls -lt命令的别名是ll，在终端输入ll时，则相当于输入了ls -lt命令

ll

total 32

-rw-r--r-- 1 fdipzone staff 1685 8 3 2016 index.html

-rw-r--r-- 1 fdipzone staff 853 8 3 2016 web.html

-rw-r--r-- 1 fdipzone staff 5943 8 3 2016 backend.php

2.查看已经设置的别名列表

alias -p

3.删除别名

unalias 别名

例如：

unalias ll

**4.设置别名每次登入可用**

alias命令只作用于当次登入的操作。如果想每次登入都能使用这些命令的别名，则可以把相应的alias命令存放在 ~/.bashrc 文件中。

打开~/.bashrc文件，输入要设置的alias命令，保存，然后运行

source ~/.bashrc

如果这样还不行，表示没有~/.bash\_profile文件或文件中没有执行~/.bashrc文件

可以在~/.bash\_profile中加入命令 source ~/.bashrc 后保存

**换了oh-my-zsh后，需要更新的是vim ~/.zshrc，然后source ~/.bashrc**

这样就可以每次登入后都可以使用设置好的命令别名。

**alias ll="ls -lhtF"**

**alias clr=clear**

**alias upgrade="sudo pacman -Syu"**

**alias update="sudo pacman-mirrors -g"**

**alias setup="sudo pacman -S"**

**alias unsetup="sudo pacman -Rs"**

**alias free="free -h"**

**alias du="du -h --max-depth=1 |sort -hr"**

**alias df="df -h"**

**alias tarcgz="tar -zcvf"**

**alias tarxgz="tar -zxvf"**

**alias tarcbz="tar -jcvf"**

**alias tarxbz="tar -jxvf"**

**alias hop=htop**

**alias updatedb=”sudo updatedb”**

***-='cd -'***

***...=../..***

....=../../..

.....=../../../..

......=../../../../..

***1='cd -'***

2='cd -2'

3='cd -3'

4='cd -4'

5='cd -5'

6='cd -6'

7='cd -7'

8='cd -8'

9='cd -9'

***\_=sudo***

**history=omz\_history**

l='ls -lah'

la='ls -lAh'

**ll='ls -lhtF'**

ls='ls --color=tty'

lsa='ls -lah'

md='mkdir -p'

rd=rmdir

run-help=man

which-command=whence

afind='ack -il'

***g=git***

ga='git add'

gaa='git add --all'

gap='git apply'

gapa='git add --patch'

gau='git add --update'

gav='git add --verbose'

gb='git branch'

gbD='git branch -D'

gba='git branch -a'

gbd='git branch -d'

gbda='git branch --no-color --merged | command grep -vE "^(\+|\\*|\s\*(master|develop|dev)\s\*$)" | command xargs -n 1 git branch -d'

gbl='git blame -b -w'

gbnm='git branch --no-merged'

gbr='git branch --remote'

gbs='git bisect'

gbsb='git bisect bad'

gbsg='git bisect good'

gbsr='git bisect reset'

gbss='git bisect start'

gc='git commit -v'

'gc!'='git commit -v --amend'

gca='git commit -v -a'

'gca!'='git commit -v -a --amend'

gcam='git commit -a -m'

'gcan!'='git commit -v -a --no-edit --amend'

'gcans!'='git commit -v -a -s --no-edit --amend'

gcb='git checkout -b'

gcd='git checkout develop'

gcf='git config --list'

***gcl='git clone --recurse-submodules'***

gclean='git clean -id'

gcm='git checkout master'

gcmsg='git commit -m'

'gcn!'='git commit -v --no-edit --amend'

gco='git checkout'

gcount='git shortlog -sn'

gcp='git cherry-pick'

gcpa='git cherry-pick --abort'

gcpc='git cherry-pick --continue'

gcs='git commit -S'

gcsm='git commit -s -m'

gd='git diff'

gdca='git diff --cached'

gdct='git describe --tags $(git rev-list --tags --max-count=1)'

gdcw='git diff --cached --word-diff'

gds='git diff --staged'

gdt='git diff-tree --no-commit-id --name-only -r'

gdw='git diff --word-diff'

gf='git fetch'

gfa='git fetch --all --prune'

gfg='git ls-files | grep'

gfo='git fetch origin'

gg='git gui citool'

gga='git gui citool --amend'

ggpull='git pull origin "$(git\_current\_branch)"'

ggpur=ggu

ggpush='git push origin "$(git\_current\_branch)"'

ggsup='git branch --set-upstream-to=origin/$(git\_current\_branch)'

ghh='git help'

gignore='git update-index --assume-unchanged'

gignored='git ls-files -v | grep "^[[:lower:]]"'

git-svn-dcommit-push='git svn dcommit && git push github master:svntrunk'

gk='\gitk --all --branches'

gke='\gitk --all $(git log -g --pretty=%h)'

gl='git pull'

glg='git log --stat'

glgg='git log --graph'

glgga='git log --graph --decorate --all'

glgm='git log --graph --max-count=10'

glgp='git log --stat -p'

glo='git log --oneline --decorate'

glod='git log --graph --pretty='\''%Cred%h%Creset -%C(auto)%d%Creset %s %Cgreen(%ad) %C(bold blue)<%an>%Creset'\'

glods='git log --graph --pretty='\''%Cred%h%Creset -%C(auto)%d%Creset %s %Cgreen(%ad) %C(bold blue)<%an>%Creset'\'' --date=short'

glog='git log --oneline --decorate --graph'

gloga='git log --oneline --decorate --graph --all'

glol='git log --graph --pretty='\''%Cred%h%Creset -%C(auto)%d%Creset %s %Cgreen(%cr) %C(bold blue)<%an>%Creset'\'

glola='git log --graph --pretty='\''%Cred%h%Creset -%C(auto)%d%Creset %s %Cgreen(%cr) %C(bold blue)<%an>%Creset'\'' --all'

glols='git log --graph --pretty='\''%Cred%h%Creset -%C(auto)%d%Creset %s %Cgreen(%cr) %C(bold blue)<%an>%Creset'\'' --stat'

glp=\_git\_log\_prettily

glum='git pull upstream master'

gm='git merge'

gma='git merge --abort'

gmom='git merge origin/master'

gmt='git mergetool --no-prompt'

gmtvim='git mergetool --no-prompt --tool=vimdiff'

gmum='git merge upstream/master'

gp='git push'

gpd='git push --dry-run'

gpf='git push --force-with-lease'

'gpf!'='git push --force'

gpoat='git push origin --all && git push origin --tags'

gpristine='git reset --hard && git clean -dfx'

gpsup='git push --set-upstream origin $(git\_current\_branch)'

gpu='git push upstream'

gpv='git push -v'

gr='git remote'

gra='git remote add'

grb='git rebase'

grba='git rebase --abort'

grbc='git rebase --continue'

grbd='git rebase develop'

grbi='git rebase -i'

grbm='git rebase master'

grbs='git rebase --skip'

grep='grep --color=auto --exclude-dir={.bzr,CVS,.git,.hg,.svn}'

grev='git revert'

grh='git reset'

grhh='git reset --hard'

grm='git rm'

grmc='git rm --cached'

grmv='git remote rename'

groh='git reset origin/$(git\_current\_branch) --hard'

grrm='git remote remove'

grs='git restore'

grset='git remote set-url'

grss='git restore --source'

grt='cd "$(git rev-parse --show-toplevel || echo .)"'

gru='git reset --'

grup='git remote update'

grv='git remote -v'

gsb='git status -sb'

gsd='git svn dcommit'

gsh='git show'

gsi='git submodule init'

gsps='git show --pretty=short --show-signature'

gsr='git svn rebase'

gss='git status -s'

gst='git status'

gsta='git stash push'

gstaa='git stash apply'

gstall='git stash --all'

gstc='git stash clear'

gstd='git stash drop'

gstl='git stash list'

gstp='git stash pop'

gsts='git stash show --text'

gsu='git submodule update'

gsw='git switch'

gswc='git switch -c'

gtl='gtl(){ git tag --sort=-v:refname -n -l "${1}\*" }; noglob gtl'

gts='git tag -s'

gtv='git tag | sort -V'

gunignore='git update-index --no-assume-unchanged'

gunwip='git log -n 1 | grep -q -c "\-\-wip\-\-" && git reset HEAD~1'

gup='git pull --rebase'

gupa='git pull --rebase --autostash'

gupav='git pull --rebase --autostash -v'

gupv='git pull --rebase -v'

gwch='git whatchanged -p --abbrev-commit --pretty=medium'

gwip='git add -A; git rm $(git ls-files --deleted) 2> /dev/null; git commit --no-verify --no-gpg-sign -m "--wip-- [skip ci]"'