LINUX COMMAND LINE CHEAT SHEET

A QUICK REFERENCE GUIDE from:

LinuxTrainingAcademy.com

Parameter Expansion	Description
${variable:-value}	If the variable is unset or undefined then expand the value.
${variable:=value}	If the variable is unset or undefined then set the value to the variable.
${variable:+value}	If the variable is set or defined then expand the value.
${variable:start:length}	Substring will retrieve from the start position to length position of the variable.
${variable:start}	Substring will retrieve from start position to the remaining part of the variable.
${#variable}	Count the length of the variable.
${variable/pattern/string}	Replace the part of the variable with string where the pattern match for the first time.
${variable//pattern/string}	Replace all occurrences in the variable with string where all pattern matches.
${variable/#pattern/string}	If the pattern exists at the beginning of the variable, then replace the occurrence with string.
${variable/%pattern/string}	If the pattern exists at the end of the variable, then replace the occurrence with string.
${variable#pattern}	Remove the shortest match from the beginning of the variable where the pattern matches.
${variable##pattern}	Remove the longest match from the beginning of the variable where the pattern matches.
${variable%pattern}	Remove the shortest match from the end of the variable where the pattern matches.
${variable%%pattern}	Remove the longest match from the end of the variable where the pattern matches.

$0 这个程式的执行名字
$n 这个程式的第n个参数值，n=1..9
$* 这个程式的所有参数，此选项参数可超过9个
$# 这个程式的参数个数
$$ 这个程式的PID（脚本运行的当前进程ID号）
$! 执行上一个背景指令的PID（后台运行的最后一个进程的进程ID号）
$? 执行上一个指令的返回值，显示最后命令的退出状态
（0表示没有错误，其他任何值表明有错误）
$- 显示shell使用的当前选项，与set命令功能相同
$@ 跟$*类似，但是可以当作数组用

Table of Contents

try “man bash” if the thing you're looking for is not a command

^ 锚定行的开始，如：'^grep'匹配所有以grep开头的行。
$ 锚定行的结束，如：'grep$'匹配所有以grep结尾的行。
. 匹配一个非换行符的字符
* 匹配任意长度的任意字符
.* 匹配任意字符
? 匹配任意单个字符
+ 匹配+号前面的字符1次或n次（等价于{1，}）
@ 匹配软链接
[ ] 匹配制定范围内的单/多个字符
e.g. [old] -- 'o', 'l', 'd'
[^] 匹配制定范围外的单个字符
{..} 生成序列（逗号分隔，无空格）
[[:space:]] 匹配单个空白字符
[[:punct:]] 匹配单个标点符号字符
[[:lower:]] 匹配单个小写字母字符
[[:upper:]] 匹配单个大写字母字符
[[:digit:]] 匹配单个数字字符
[[:alnum:]] 匹配单个数字和字母字符
$() <=> `` – the command substitution
${} – the parameter substitution/variable expansion

SHELL SCRIPT / IF / WHILE / FOR

num_args.sh第一行总是#!/bin/bash：
指此脚本使用/bin/bash来解释执行（#!是一个特殊的表示符，其后跟着解释此脚本的shell路径）
space-sensitive: newvar="something" 和 if test -z $newvar; 和 if [[-z $newvar]];
echo ${newvar}用于variable；echo $(pwd)与 echo `pwd`用于command
(in if sentence)
echo $1 #first argument
shift #throw away the first one and make the rest of input replace $1

1 - SYSTEM INFORMATION

if [[-?]]:
man bash
/conditional expressions

```
uname -a                    # Display Linux system information
```
if [! \("$INT" -ge "$MIN_VAL" -a "$INT" -le "$MAX_VAL" \)]; <===
```
uname -r                    # Display kernel release information
```
===> if [[! ("$INT" -ge "$MIN_VAL" && "$INT" -le "$MAX_VAL")]];
```
cat /etc/redhat-release     # Show which version of Red Hat installed
```
if [“x${var}" == “x”] --> 判断${var}是否为空
```
uptime                      # Show how long the system has been running + load
```
--> 防止出现语法错误。如果不写x，当${var}为空或未设置时，语句被解释为 if [== "0"]，出现语法错误。
```
hostname                    # Show system host name
```
for i in {A..D}; do echo $i; done --> {A..D} only has two dots
```
hostname -I                 # Display all local IP addresses of the host
```
for ((i=0; i<5; i=i+1)); do echo $i; done --> using (())
```
last reboot                 # Show system reboot history

date                        # Show the current date and time

cal                         # Show this month's calendar

w                           # Display who is online
```
相同：cat input.txt | while read anything; do echo $anything | grep -v e; done
```
whoami                      # Who you are logged in as
```
while read anything; do echo $anything | grep -v e; done < input.txt
其他：ls -1f | while read anything; do echo $anything | grep -v e; done

2 - HARDWARE INFORMATION

set -uxe #in a shell script just after "#!" line;
#-u 执行时使用到未定义过的变量，则显示错误信息；-x 执行指令后，会先显示该指令及所下的参数
```
dmesg                       # Display messages in kernel ring buffer
```
cat << anylabel ..... anylabel #here document (type the label name again to stop entering and run)
```
cat /proc/cpuinfo           # Display CPU information

cat /proc/meminfo           # Display memory information

free -h                     # Display free and used memory ( -h for human readable,
                            -m for MB, -g for GB.)

lspci -tv                   # Display PCI devices

lsusb -tv                   # Display USB devices

dmidecode                   # Display DMI/SMBIOS (hardware info) from the BIOS

hdparm -i /dev/sda          # Show info about disk sda

hdparm -tT /dev/sda         # Perform a read speed test on disk sda

badblocks -s /dev/sda       # Test for unreadable blocks on disk sda
```
>表示覆盖原文件内容（文件的日期也会自动更新）
>>表示追加内容（会另起一行，文件的日期也会自动更新）

VARIABLE (Parameter Expansion见上)

MY_VAR="$MY_VAR/foo"; MY_VAR="bar/$MY_VAR"; echo "$MY_VAR" --> 输出：bar//foo
myvar=something; if [["$myvar" =~ .ome]]; then echo "pass"; fi --> pass(~ is a match)
num=5; num=$(($num + 1)) 或 num=$((num + 1)) --> (()) is for integer; 计算（可用作判断条件）

3 - PERFORMANCE MONITORING AND STATISTICS

othervar=${myvar:-defval}
-->if myvar has value, othervar=myvar; else othervar=defval

`top`	# Display and manage the top processes
newvar=${#myvar} --> get the length of myvar	
`htop`	# Interactive process viewer (top alternative)
othervar=${myvar: 2} --> omthing	
`mpstat 1`	# Display processor related statistics
othervar=${myvar: -2} --> ng	
`vmstat 1`	# Display virtual memory statistics
othervar=${myvar: m:n} --> drop the first m letters and show the next n letters (eg. m=2, n=4 --> meth)	
`iostat 1`	# Display I/O statistics
stat file #detailed information of the file (include inode)	
`tail -100 /var/log/messages`	# Display the last 100 syslog messages (Use
echo $myvar | cut -d ' ' -f 2 或 | awk -F ' ' {print $2}	`/var/log/syslog` for Debian based systems.)
--> get substring, separated by " " (space in "), 2 means the second part (can also be 2,3 OR 2-5)	
`tcpdump -i eth0`	# Capture and display all packets on interface eth0
eg. 2021-03-07 12:41:28 Just Do It Now --> 2-4 gets 12:41:28 Just Do	
`tcpdump -i eth0 'port 80'`	# Monitor all traffic on port 80 (HTTP)
`lsof`	# List all open files on the system
`lsof -u user`	# List files opened by `user`
`free -h`	# Display free and used memory (-h for human readable, -m for MB, -g for GB.)
`watch df -h`	# Execute "df -h", showing periodic updates

4 - USER INFORMATION AND MANAGEMENT

`id`	# Display the user and group ids of your current user.
`last`	# Display the last users who have logged onto the system.
`who`	# Show who is logged into the system.
`w`	# Show who is logged in and what they are doing.
`groupadd test`	# Create a group named "test".
`useradd -c "John Smith" -m john`	# Create an account named john, with a comment of "John Smith" and create the user's home directory.
`userdel john`	# Delete the john account.
`usermod -aG sales john`	# Add the john account to the sales group

ctrl-x ctrl-e连用可将正在输入行导入新vim进行编辑，:wq退出后直接运行vim中的所有命令

5 - FILE AND DIRECTORY COMMANDS

ls -lf #"-f" -> enable -aU (with hidden files)
ls -l #list in a long listing (no hidden file) (with hidden files)

```
ls -al                          # List all files in a long listing (detailed) format same as "ll"
```

ls -la | less #same as above but (with "| less") read only one screen at a time of information

```
pwd                             # Display the present working directory
```

ls -1F #-1, --format=single-column -> list one file per line. #-F, Append a character to each file name indicating the file type.
`*' -> regular executable file
`/' -> directories
`@' -> symbolic links
`|' -> FIFOs
`=' -> sockets
nothing for regular files

```
mkdir directory                 # Create a directory

rm file                         # Remove (delete) file

rm -r directory                 # Remove the directory and its contents
                                recursively

rm -f file                      # Force removal of file without prompting for
                                confirmation

rm -rf directory                # Forcefully remove directory recursively

cp file1 file2                  # Copy file1 to file2

cp -r source_directory destination  # Copy source_directory recursively to
                                destination. If destination exists, copy
                                source_directory into destination,
                                otherwise create destination with the
                                contents of source_directory.

mv file1 file2                  # Rename or move file1 to file2. If file2 is
                                an existing directory, move file1 into directory
                                file2

ln -s /path/to/file linkname    # Create symbolic link to linkname

touch file                      # Create an empty file or update the access
                                and modification times of file.

cat file                        # View the contents of file

less file                       # Browse through a text file

head file                       # Display the first 10 lines of file

tail file                       # Display the last 10 lines of file

tail -f file                    # Display the last 10 lines of file and "follow"
                                the file as it grows.
```

tee #read from standard input and write to standard output and files
#eg. echo -e "negative\nor\n-ve" | tee output.txt
sort file #sort in order
sort -u file #remove duplicate lines from output lines
sort -n file #sort by number (for 10 and 2, compare 10 with 2, not 1 with 2 -> as string)
uniq #report or omit (BUT NOT DELETE) repeated lines
tr [option] set1 [set2] #translate or delete characters
cat << EOF > test.txt #EOF: End Of File; send a sentence block to cat and output cat to test.txt
> alalala
> EOF

ctrl + z to exit the running process
any command (e.g. sleep 10) & -- ask the command to run in background

6 - PROCESS MANAGEMENT

ps -ejH #print a process tree
service nginx start #then use ps -ef to make sure the process is working well

`ps`	# Display your currently running processes
`ps -ef`	# Display all the currently running processes on the system.

ps aux会截断command列，而-ef不会。当结合grep时这种区别会影响到结果。

`ps -ef | grep processname`	# Display process information for `processname`
`top`	# Display and manage the top processes
`htop`	# Interactive process viewer (top alternative)
`kill pid`(process id)	# Kill process with process ID of `pid`
`killall processname`	# Kill all processes named `processname`

kill -l #list all the signal name

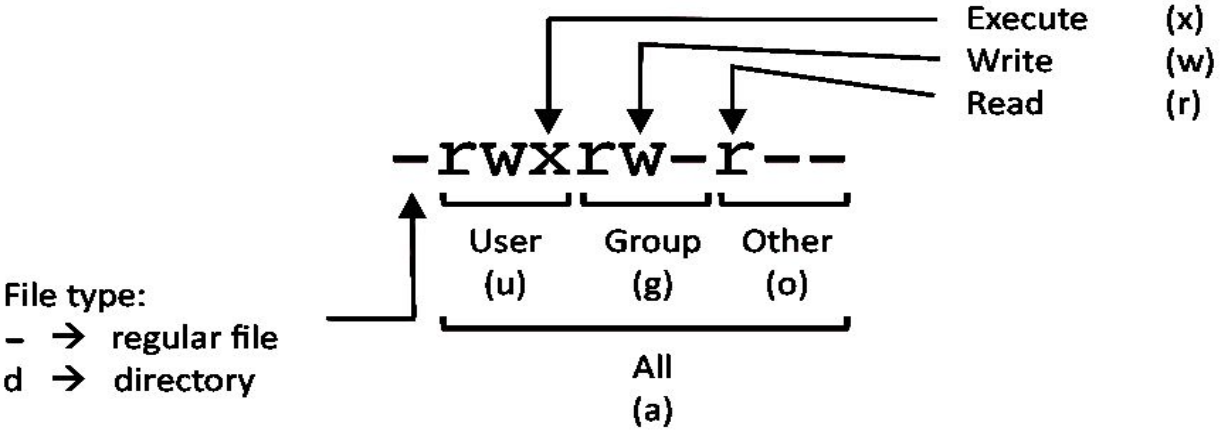
`program &`	# Start `program` in the background
`bg`	# Display stopped or background jobs
`fg`	# Brings the most recent background job to foreground

jobs #list the jobs (but stopped)

`fg n`	# Brings job `n` to the foreground

echo $$ #ask to print the current process id
echo $? #what happened to the last command you run (0 - no problem; non-zero - problems)

7 - FILE PERMISSIONS



PERMISSION			EXAMPLE	
U	G	W		
rwx	rwx	rwx	`chmod 777 filename`	# Use sparingly!
rwx	rwx	r-x	`chmod 775 filename`	
rwx	r-x	r-x	`chmod 755 filename`	
rw-	rw-	r--	`chmod 664 filename`	
rw-	r--	r--	`chmod 644 filename`	

root@62a2b2ff2a07:~/signal# sleep 100
Killed
root@62a2b2ff2a07:~/signal#

root@62a2b2ff2a07:~/signal# ps -ef | grep sleep
root 42 14 0 09:41 pts/1 00:00:00 sleep 100
root 44 27 0 09:41 pts/2 00:00:00 grep --color=auto sleep
root@62a2b2ff2a07:~/signal# kill -9 42
root@62a2b2ff2a07:~/signal#

```
ls -1f ; echo "hi there"          #";" separates two commands; execute in order
ls input.txt || echo "ls failed"  #condition "or"
                                  #if the first command can run, the second doesn't run;
                    LEGEND        #if the first cannot run, the second runs
ls input.txt && echo "ls failed"  #condition "and"; both commands run or not run together
                    U = User      chmod [ugoa] [+-=] [rwx]
                    G = Group
                    W = World

                    r = Read                0 -- stdin
                    w = write               1 -- stdout
                    x = execute             2 -- stderr
                    - = no access           (used for redirection from the screen to a file)
                                            e.g. ls dne.file 2>/dev/null
```

8 - NETWORKING

`ip a`	# Display all network interfaces and IP address
`ip addr show dev eth0`	# Display eth0 address and details
`ethtool eth0`	# Query or control network driver and hardware settings
`ping host`	# Send ICMP echo request to `host`
`whois domain`	# Display whois information for `domain`
`dig domain`	# Display DNS information for `domain`
`dig -x IP_ADDRESS`	# Reverse lookup of `IP_ADDRESS`
`host domain`	# Display DNS IP address for `domain`
`hostname -i`	# Display the network address of the host name.
`hostname -I`	# Display all local IP addresses of the host.
`wget http://domain.com/file`	# Download `http://domain.com/file`
`netstat -nutlp`	# Display listening tcp and udp ports and corresponding programs

9 - ARCHIVES (TAR FILES)

`tar cf archive.tar directory`	# Create tar named `archive.tar` containing `directory`.
`tar xf archive.tar`	# Extract the contents from `archive.tar`.
`tar czf archive.tar.gz directory`	# Create a gzip compressed tar file name `archive.tar.gz`.

```
tar xzf archive.tar.gz                  # Extract a gzip compressed tar file.
tar cjf archive.tar.bz2 directory       # Create a tar file with bzip2 compression
tar xjf archive.tar.bz2                 # Extract a bzip2 compressed tar file.
```

10 - INSTALLING PACKAGES

(sudo) apt-get install package
(sudo) apt-get remove package

```
yum search keyword'                     # Search for a package by keyword.
yum install package                     # Install package.
yum info package                        # Display description and summary information
                                        about package.
rpm -i package.rpm                      # Install package from local file named
                                        package.rpm
yum remove package                      # Remove/uninstall package.
tar zxvf sourcecode.tar.gz              # Install software from source code.
cd sourcecode
./configure
make
make install
```

Meta-Sequences:
[[:alnum:]], [[:alpha:]], [[:cntrl:]], [[:digit:]], [[:lower:]], [[:print:]], [[:punct:]], [[:space:]], [[:upper:]], [[:xdigit:]]

Characters:
abc - Literal abc
[0-9] - Range 0 - 9
[a-z] - Range a - z
. - Any character

Constructs:
( ) - Group
| - Or
^ - Start of string
$ - End of string

Quantifiers:
? - Zero or Once
* - Zero or More
+ - One or More
{n} - Exactly n times
{n,} - n or More times
{,m} - At most m times
{n,m} - n to m times

11 - SEARCH

```
grep pattern file                       # Search for pattern in file
grep -r pattern directory               # Search recursively for pattern in directory
locate name                             # Find files and directories by name
find /home/john -name 'prefix*'         # Find files in /home/john that start with "prefix".
find /home -size +100M                  # Find files larger than 100MB in /home
find . | xargs grep PATH    #xargs展开find获得的结果，使其作为grep的参数
echo "one two three" | awk '{print $1}'    #chop the things, for this you will get: one
```

12 - SSH LOGINS

```
ssh host                                # Connect to host as your local username.
ssh user@host                           # Connect to host as user
ssh -p port user@host                   # Connect to host using port
```

13 - FILE TRANSFERS

`scp file.txt server:/tmp`	# Secure copy `file.txt` to the /tmp folder on `server`
`scp server:/var/www/*.html /tmp`	# Copy `*.html` files from `server` to the local /tmp folder.
`scp -r server:/var/www /tmp`	# Copy all files and directories recursively from `server` to the current system's /tmp folder.
`rsync -a /home /backups/`	# Synchronize `/home` to `/backups/home`
`rsync -avz /home server:/backups/`	# Synchronize files/directories between the local and remote system with compression enabled

14 - DISK USAGE

`df -h`	# Show free and used space on mounted filesystems
`df -i`	# Show free and used inodes on mounted filesystems
`fdisk -l`	# Display disks partitions sizes and types
`du -ah`	# Display disk usage for all files and directories in human readable format
`du -sh`	# Display total disk usage off the current directory

15 - DIRECTORY NAVIGATION

`cd ..` # To go up one level of the directory tree. (Change into the parent directory.)

cd . **#Current directory**

`cd` # Go to the $HOME directory

cd ~ **#To the home directory of current user (root user: /root; username: /home/username)**

`cd /etc` # Change to the /etc directory

cd / **#To /root**

cd one **#one is a path under the current directory -> This is a relative path**

cd /home/one **#it's the same as root/home/one -> This is an absolute path (start with "/")**