BRUNNA CROCHES

PARTE 3

TERMINAL

Guia do TERMINAL

ABOUT ME

Brunna Croches

Developer Full Stack

Brunna Croches é Dev FullStack, advogada e empreendedora.

Apaixonada por tech, vem adquirido vasto conhecimento na área.

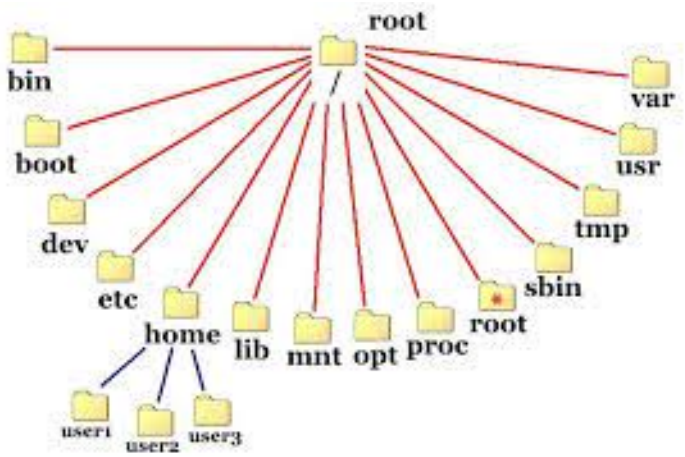
Desenvolveu projetos ricos em diversidade, buscando captar as próximas tendências e necessidades do mercado.

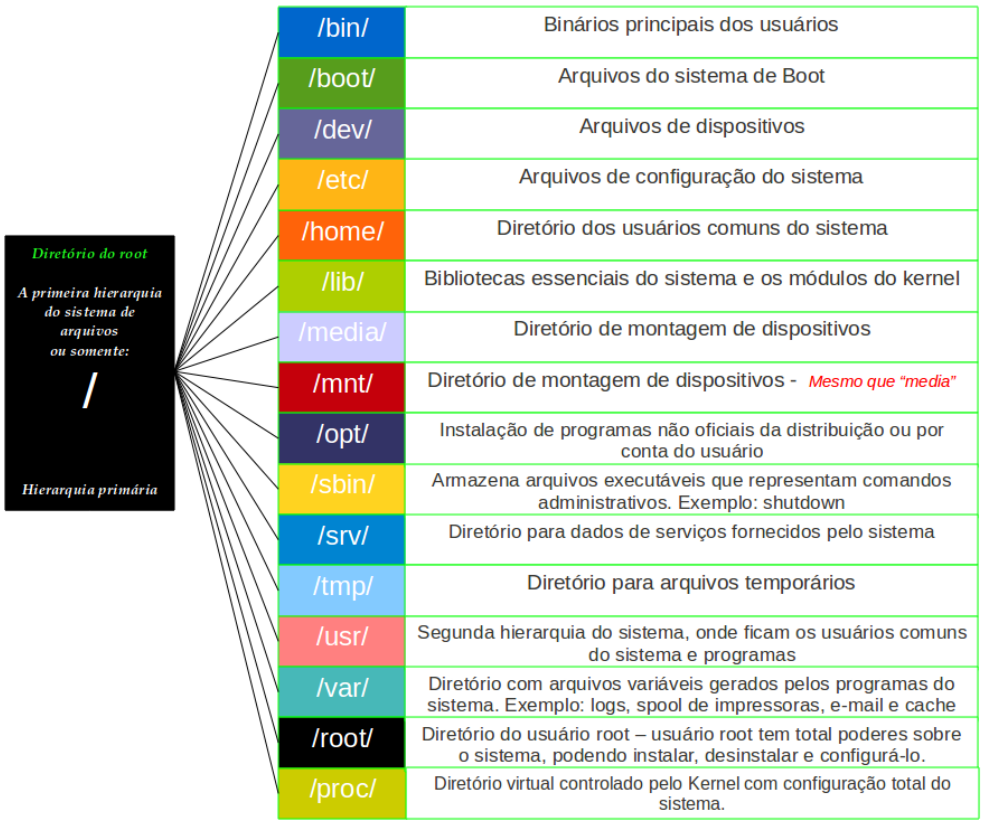
Neste e-book você aprenderá ou recapitulará de forma simplificada e otimizados conceitos de programação feito por ela.

let's share

SUMMARY

1.8 ESTRUTURA DE DIRETÓRIO





HTML Content

A Estrutura de Diretórios do Linux

Postado Em 30/10/2018[data do post] por **Fábio dos Reis**[autor do post] em **Linux**[categoria do post]

A Estrutura de Diretórios do Linux

A estrutura de diretórios do Linux pode parecer um pouco estranha a princípio para quem vem do ambiente Windows. Por exemplo, não temos letras de unidade como C: e D:, e as partições são numeradas, entre outras diferenças.

A estrutura dos sistemas de arquivos do Linux é definida por um padrão denominado ***Filesystem Hierarchy Standard***(Padrão de Hierarquia do Sistema de Arquivos), a qual define também as estruturas de outros sistemas como o BSD, por exemplo.

O Filesystem Hierarchy Standard (FHS) evoluiu a partir de padrões históricos originados de versões mais antigas do UNIX, como a Berkeley Software Distribution (BSD) e outras. O FHS fornece aos desenvolvedores Linux e administradores de sistemas uma estrutura de diretórios padrão para o sistema de arquivos, trazendo consistência entre sistemas e distribuições.

No site **http://www.pathname.com/fhs/**podemos acessar o documento padrão FHS e baixá-los para consulta e estudo.

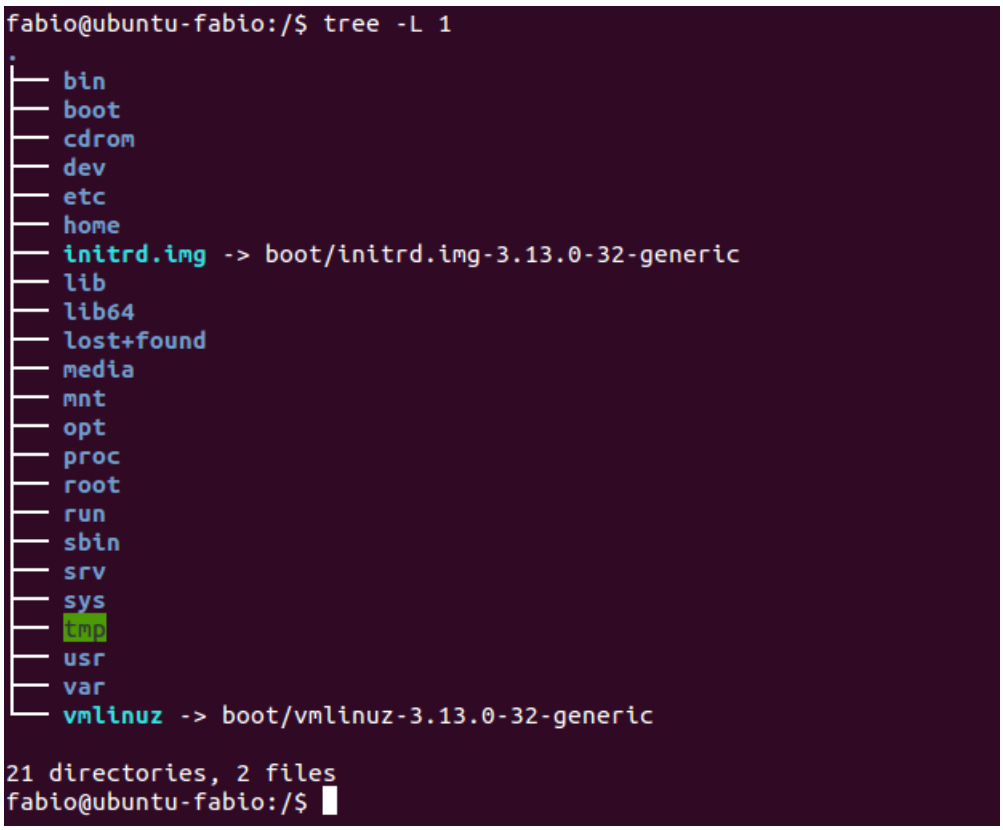
A página oficial do projeto é **http://www.linuxfoundation.org/collaborate/workgroups/lsb/fhs**

Vejamos um resumo sobre a estrutura de diretórios no Linux e uma breve descrição dos principais diretórios:

/ – O diretório Root (Raiz)

Tudo o que há no seu sistema Linux fica localizado dentro do diretório raiz, representado por **/**. É como se fosse um “C:\” do Windows, porém outras partições e discos também se localizam sob o diretório raiz no Linux, enquanto no Windows cada partição teria sua própria letra de unidade separada. No Linux, as demais partições se encontram “montadas” em pastas dentro da hierarquia de diretórios, sob a raiz (/).

Veja na figura abaixo o diretório raiz do Linux e seu conteúdo básico, listado com o comando **tree -L 1**:



/bin – Binários essenciais dos usuários

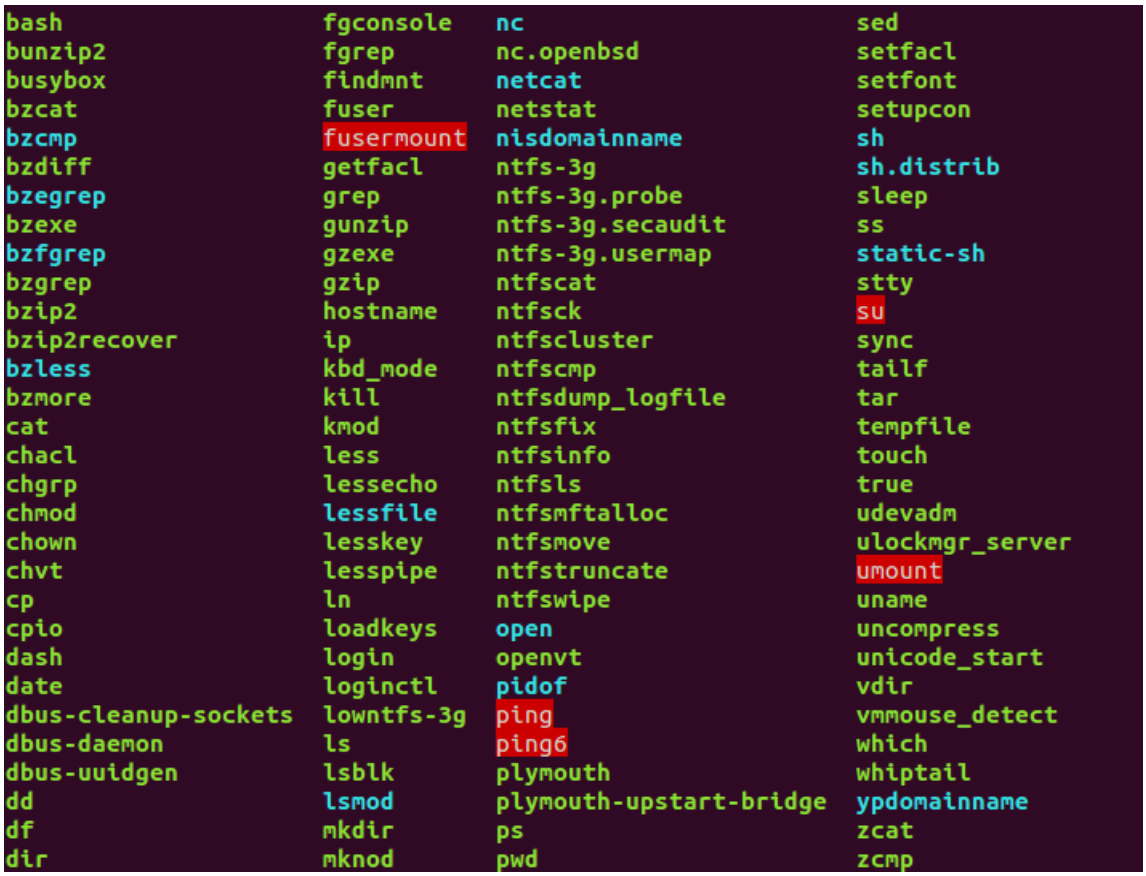
O diretório **/bin**contém binários essenciais aos usuários – ou seja, programas – que precisam estar presentes quando o sistema é inicializado no modo monousuário. Aplicativos comuns,como navegadores e jogos geralmente se localizam no diretório **/usr/bin**, ao passo

que programas e utilitários importantes são armazenados no diretório /bin.

O diretório /bin contém binários executáveis, comandos essenciais que são utilizados quando em modo monousuário e também muitos comandos essenciais que são requeridos por todos os usuários do sistema, tais como ls, rmdir e date.

Já os comandos que não são essenciais para o sistema quando em modo monousuário são colocados no diretório /usr/bin , ao passo que o diretório **/sbin**é usado para armazenar binários essenciais que tem relação com a administração do sistema.

Veja abaixo uma figura mostrando o conteúdo básico do diretório /bin:



/boot – Arquivos estáticos de inicialização

O diretório /boot contém arquivos necessários para inicializar o sistema, como os arquivos do carregador de inicialização **GRUB**e o kernel (ou kernels) do Linux. Alguns arquivos de configuração se localizam no diretório /etc.

/cdrom – Ponto de montagem para drives ópticos, como CD-ROMs

Este diretório não faz parte do padrão de hierarquia FHS, porém ele ainda é encontrado em algumas distribuições Linux. Ele é usado como local temporário para CDs e DVDs inseridos no drive – porém, o local padrão para essas mídias é o diretório **/media**.

/dev – Arquivos de Dispositivos

No Linux os dispositivos (hardware e software) são representados como arquivos, e esse diretório contém uma grande quantidade de arquivos especiais que representam esses dispositivos.

Estes arquivos de dispositivos não são arquivos comuns, e seu conteúdo não é legível – tente rodar o comando **cat /dev/sda1**para ver o que aparece.

Na verdade, o **/dev/sda**representa o primeiro disco SATA instalado no sistema. Para trabalhar com esse disco, é necessário usar um utilitário especial como o fdisk ou o mkfs, por exemplo.

Neste diretório também encontramos os chamados “pseudo-dispositivos”, que são dispositivos virtuais que não correspondem a hardware real da máquina. Como exemplo podemos citar o dispositivo especial /dev/null que não produz saída nenhuma e automaticamente descarta toda a entrada direcionada a ele – é como um “buraco negro” de dados.

O diretório /dev contém nós de dispositivos, que são uma espécie de pseudo-arquivos usados por muitos dispositivos de hardware (e de software) com a exceção de alguns dispositivos de rede. O diretório /dev/está sempre vazio quando não for montado, e se for montado conterá entradas que são criadas pelo sistema udev, o qual cria e também gerencia os nós no Linux, criando-os de forma dinâmica quando os dispositivos são encontrados.

Este diretório é interessante pois mostra uma característica marcante do sistema operacional Linux: no Linux, tudo é um arquivo ou diretório. Usamos esses arquivos para configurar e acessar vários dispositivos de hardware. Veremos com mais detalhes essa teoria e o funcionamento do diretório /dev em outro tópico. Abaixo, a definição de alguns dos arquivos encontrados dentro de /dev:

Arquivo em /dev	Definição
/dev/dsp	Digital Signal Processor; interface entre o software que produz um som e a placa de som
/dev/fd0	Primeiro drive de 3 ½
/dev/hda	Drive Master na controladora primária IDE
/dev/hdb	Drive Slave na controladora primária IDE
/dev/hdc e /dev/hdd	Drives master e slave na controladora secundária IDE
/dev/ht0	Primeiro drive de fita IDE
/dev/lp0	Primeiro dispositivo de impressora paralela
/dev/psaux	Porta de mouse PS/2
/dev/loop0	Primeiro dispositivo de loopback. Usado para montar sistemas de arquivos não localizados em outros dispositivos de blocos, como uma imagem ISO9660 sem gravá-la em mídia
/dev/null	“Balde de bits”; Buraco negro para onde dados podem ser enviados – e nunca mais vistos
/dev/random	Gerador de números aleatórios do kernel

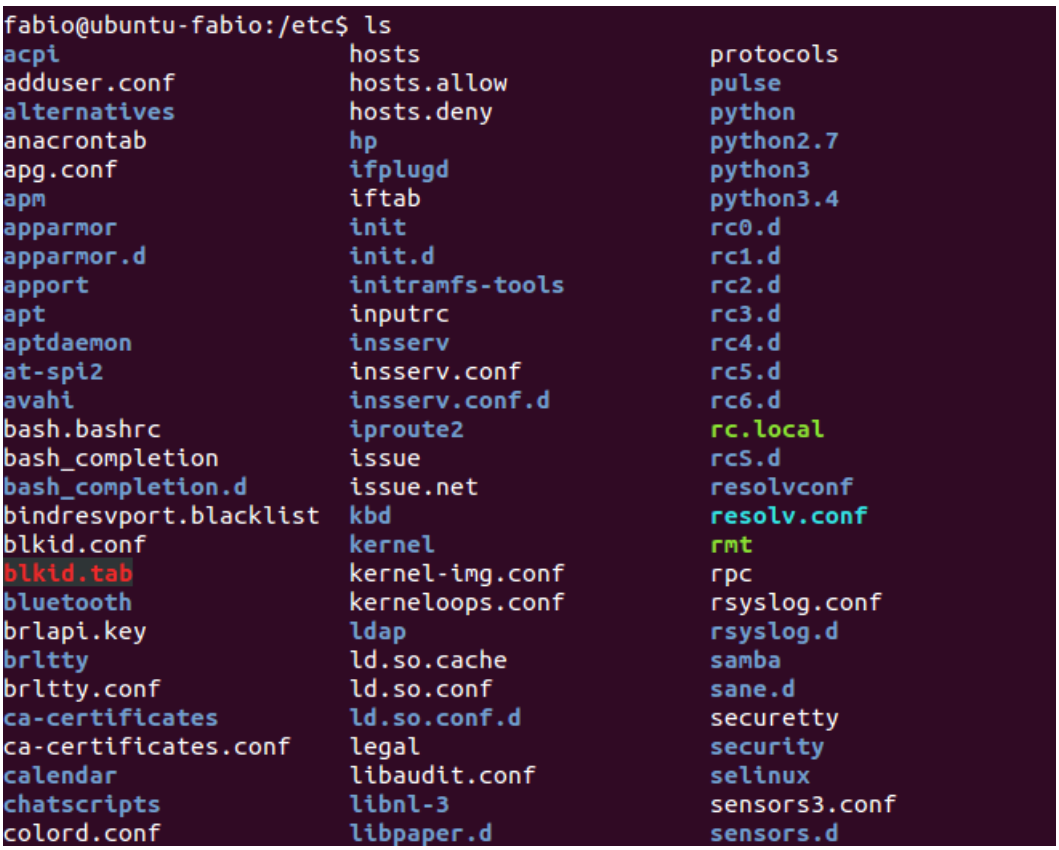
Arquivo em /dev	Definição
/dev/sda	Primeiro dispositivo SCSI ou SATA
/dev/sdb	Segundo dispositivo SCSI ou SATA
/dev/ttySO	Primeira porta serial
/dev/zero	Muitos zeros – retorna o valor 0

Uma listagem bem completa de arquivos de dispositivos pode ser encontrada em **http://www.lanana.org/docs/device-list/devices-2.6+.txt**

/etc – Arquivos de configuração diversos

O diretório /etc contém muitos arquivos de configuração do sistema, os quais podem geralmente ser editados manualmente usando-se um editor de textos, como o vi ou o emacs.

O diretório /etc não contém programas binários, contendo apenas scripts executáveis. Veja abaixo uma parte do conteúdo do diretório /etc:



Como exemplo na figura vemos o arquivo hosts que possui mapeamentos entre endereços IP e nomes de computadores, e o arquivo resolv.conf, que possui informações sobre resolução de nomes (servidores DNS).

/home – Diretórios Home dos usuários

O diretório **/home**contém um diretório padrão (de perfil) para cada usuário. Se o nome de seu usuário é fabio, então você encontrará um diretório de nome fabio dentro de /home, portando /home/fabio. Este diretório contém arquivos do usuário fabio e arquivos de configuração

específicos dessa conta de usuário. Os usuários possuem permissão de gravação apenas em seu próprio diretório padrão, e apenas permissão de leitura em outros diretórios do sistema (em alguns casos, permissão nenhuma).

Veja abaixo a listagem de um diretório /home, que contém três diretórios pertencentes às contas dos usuários fabio, mariana e daniela:

```
fabio@ubuntu-fabio:/home$ ls -l
total 12
drwxr-xr-x  2 daniela daniela 4096 Ago 21 21:02 daniela
drwxr-xr-x 21 fabio   fabio   4096 Ago 21 20:52 fabio
drwxr-xr-x  2 mariana mariana 4096 Ago 21 21:02 mariana
fabio@ubuntu-fabio:/home$
```

/lib – Bibliotecas compartilhadas essenciais do sistema

O diretório **/lib**contém bibliotecas que são necessárias aos programas localizados nos diretórios /bin /sbin.

Já as bibliotecas usadas pelos programas do diretório /usr/bin se localizam no diretório /usr/lib.

As bibliotecas são códigos de programas que são compartilhados entre aplicações e são necessários para que elas funcionem. Essas bibliotecas geralmente tem nomes que se iniciam com ld or lib.

```
fabio@ubuntu-fabio:/lib$ ls
apparmor                          libip6tc.so.0          plymouth
brltty                            libip6tc.so.0.1.0      recovery-mode
cpp                               libiptc.so.0           resolvconf
crda                              libiptc.so.0.0.0       security
firmware                          libxtables.so.10       systemd
hdparm                            libxtables.so.10.0.0   terminfo
ifupdown                          linux-sound-base       udev
init                              lsb                    ufw
klibc-P2s_k-gf23VtrGgO2_4pGkQgwMY.so  modprobe.d         x86_64-linux-gnu
libip4tc.so.0                     modules                xtables
libip4tc.so.0.1.0                 modules-load.d
fabio@ubuntu-fabio:/lib$
```

Algumas distribuições Linux possuem o diretório /lib64, o qual contém bibliotecas de 64 bits, ao passo que o diretório /lib armazena as bibliotecas de 32 bits. Em sua maioria essas bibliotecas são chamadas de dlls, “dynamically loaded libraries”, ou ainda shared libraries (bibliotecas compartilhadas) ou também Shared Objects (SO).

Neste diretório também costumamos encontrar os módulos do Kernel, que são códigos do kernel, geralmente drivers de dispositivos carregáveis e descarregáveis sem que seja necessária a reinicialização do sistema. Esses módulos são encontrados no subdiretório /lib/modules/<kernel-version-number.

/lost+found – Arquivos recuperados (“perdidos+encontrados”)

Caso ocorra um travamento no sistema, uma verificação do sistema de arquivos será realizada na próxima reinicialização. Se forem encontrados arquivos corrompidos eles serão colocados no diretório lost+found, permitindo assim que recuperemos esses dados, ou ao menos boa parte deles.

/media – Mídias Removíveis

O diretório **/media**contém subdiretórios onde são montados dispositivos de mídias removíveis inseridas no computador, como por exemplo um CD inserido no drive de CD/DVD, o qual será montado em um diretório criado automaticamente dentro de /media, nos permitindo acessar o conteúdo da mídia.

/mnt – Pontos de montagem temporários

O diretório **/mnt**é onde sistemas de arquivos temporários podem ser montados pelos administradores de sistemas durante seu uso. Por exemplo, podemos montar uma partição do Windows nesse diretório para efetuarmos tarefas de recuperação de arquivos, criando o diretório /mnt/windows.

Na verdade, podemos montar esses sistemas de arquivos em vários locais do sistema, não necessariamente em /mnt.

/opt – Pacotes opcionais

O diretório **/opt**é um contêiner para pacotes de software opcionais. Muito usado por determinados softwares proprietários que não obedecem ao FHS.

Muitos sistemas unix armazenam software compilado localmente no diretório /opt, e se esse for o caso, é interessante mantê-lo em uma partição separada.

Neste caso também é interessante criar um link simbólico do diretório /usr/local para o diretório /opt.

Algumas distribuições instalam softwares pre-compilados em outros diretórios, como o /usr/bin.

/proc – Arquivos de Processos e de Kernel

O diretório **/proc**contém arquivos especiais que representam informações sobre processos e sobre o sistema.

O sistema de arquivos virtual proc

O sistema de arquivos montado no diretório **/proc**é também conhecido como pseudo sistema de arquivos, pois seus arquivos não existem fisicamente no disco rígido do computador.

O sistema de arquivos /proc contém arquivos virtuais, que são arquivos que existem apenas na memória RAM da máquina, os quais permitem visualizar dados do kernel que mudam constantemente. Não há arquivos reais aqui, mas informações em tempo real do sistema, como configurações de hardware e memória física. Podemos citar alguns arquivos presentes no diretório /proc:

/proc/cpuinfo– Informações sobre a CPU

/proc/interrupts– Informações sobre IRQs

/proc/meminfo– Informações sobre a memória do sistema

/proc/mounts– Informações sobre dispositivos e pontos de montagem

/proc/partitions– Informações sobre as partições dos discos

/proc/version– Versão do Kernel do Linux e do compilador gcc.

E o diretório /proc também possui alguns subdiretórios, como o diretório /proc/<ID_processo>, que se trata de um diretório que contém informações sobre um processo rodando no sistema. Há um diretório desses para cada processo em execução.

Também existe o subdiretório **/proc/sys**, o qual possui muitas informações sobre o sistema, como configurações e dados sobre o hardware.

/root – Diretório home do usuário root

Este diretório é o diretório padrão do usuário root. Veja que o root não tem seu diretório home como subdiretório de /ho,e, cmo todos os demais usuários.

Não confunda o diretório /root com o diretório **/**(root), que é o diretório raiz do sistema.

/run – Arquivos de Estado de Aplicações

O diretório **/run**fornece às aplicações um local para armazenamento de arquivos transientes que elas necessitem usar, como por exemplo sockets.

Apesar desses arquivos serem transientes, eles não são armazenados em /tmp pois lá os arquivos poderiam ser excluídos causando problemas às aplicações que os utilizam.

/sbin – Binários para Administração do Sistema

O diretório **/sbin**é muito parecido com o diretório /bin. Ele possui muitos programas binários essenciais que são geralmente utilizados pelo administrador do sistema em suas tarefas de gerenciamento.

```
fabio@ubuntu-fabio:/sbin$ ls
acpi_available   fstrim-all         mkfs              restart
agetty           gdisk              mkfs.bfs          rmmod
alsa             getcap             mkfs.cramfs       route
apm_available    getpcaps           mkfs.ext2         rtacct
apparmor_parser  getty              mkfs.ext3         rtmon
badblocks        halt               mkfs.ext4         runlevel
blkid            hdparm             mkfs.ext4dev      setcap
blockdev         hwclock            mkfs.fat          setvtrgb
bridge           ifconfig           mkfs.minix        sfdisk
brltty           ifdown             mkfs.msdos        sgdisk
brltty-setup     ifquery            mkfs.ntfs         shadowconfig
capsh            ifup               mkfs.vfat         shutdown
cfdisk           init               mkhomedir_helper  slattach
cgdisk           initctl            mkntfs            start
crda             insmod             mkswap            startpar
ctrlaltdel       installkernel      mntctl            startpar-upstart-inject
debugfs          ip                 modinfo           start-stop-daemon
depmod           ip6tables          modprobe          status
dhclient         ip6tables-apply    mountall          stop
dhclient-script  ip6tables-restore  mount.fuse        sulogin
dmsetup          ip6tables-save     mount.lowntfs-3g  swaplabel
dosfsck          ipmaddr            mount.ntfs        swapoff
dosfslabel       iptables           mount.ntfs-3g     swapon
dumpe2fs         iptables-apply     mount.vboxsf      switch_root
e2fsck           iptables-restore   nameif            sysctl
e2image          iptables-save      ntfsclone         tc
e2label          iptunnel           ntfscp            telinit
```

Como exemplo podemos citar os programas formatadores de partições **mkfs.ext3**e **mkfs.vfat**, e os comandos **shutdown**e **runlevel**.

/srv – Dados de Serviços

O diretório **/srv**possui dados que são utilizados por serviços Como exemplo podemos citar o web server Apache, que usa o diretório /srv (na verdade, um subdiretório dentro dele) para armazenamento das páginas (arquivos) de um website.

/sys – Informações sobre o sistema e hardware

Sistema de arquivos pseudo-virtual que fornece informações sobre o sistema e o hardware do computador. Pode ser usado para alterar parâmetros do sistema e também para tarefas de debuggung.

Função: facilitar a troca de informações entre os programas que rodam no espaço do kernel, como os drivers, com os programas que rodam no espaço do usuário (aplicações).

Quando um dispositivo é adicionado ao sistema, o kernel cria um nome de dispositivo em /sys, e notifica o utilitário udev, o qual gerencia os nomes de dispositivos dinamicamente, criando então um arquivo de dispositivo, geralmente em /dev (ou o remove)

Alguns subdiretórios de /sys e suas respectivas funções:

/sys/bus– Barramentos de dados do sistema

/sys/module– Módulos carregados no kernel

/sys/devices– Todos os dispositivos conectados

/sys/block– Dispositivos de bloco, como HDs

/tmp – Arquivos temporários

As aplicações podem armazenar arquivos temporários no diretório **/tmp**. Estes arquivos são geralmente excluídos quando o sistema é reiniciado e podem ser excluídos a qualquer momento por utilitários como o tmpwatch.

O diretório /tmp é onde os usuários, assim como os programas armazenam arquivos de forma temporária.

É interessante mantê-lo em uma partição separada pois seu conteúdo pode crescer muito e interferir com o restante do sistema.

/usr – Binários dos usuários e Dados Somente-Leitura

O diretório /usr contém aplicações e arquivos utilizados pelos usuários comuns do sistema, ao contrário das aplicações e arquivos que são usados pelo sistema em si. Por exemplo, aplicações não-essenciais estão localizadas dentro do diretório /usr/bin em vez do diretório /bin e binários de administração do sistema ficam localizados no diretório /usr/sbin em vez do diretório /sbin. Bibliotecas estão locadas dentro do diretório /usr/lib.
O diretório /usr também contém outros diretórios – por exemplo, arquivos que não dependem de arquitetura como gráficos, que se localizam no diretório /usr/share.

Já o diretório **/usr/local**é onde aplicações compiladas localmente são instaladas por padrão – isso as impedem de bagunçar o resto do sistema.

Também é interessante manter esse diretório em uma partição separada, pois pacotes são adicionados costumeiramente ao sistema.

Subdiretórios localizados em /usr

O diretório /usr possui ao menos os seguintes subdiretórios:

Diretório	Uso
/usr/include	Arquivos de cabeçalho usados para compilar aplicações
/usr/lib	Bibliotecas para programas nos diretórios /usr/bin e /usr/sbin.
/usr/lib64	Bibliotecas de 64 bits para programas de 64 bits nos diretórios /usr/bin e /usr/sbin.
/usr/sbin	Binários de sistema não essenciais, tais como *daemons*do sistema.

Diretório	Uso
/usr/share	Dados compartilhados usados por aplicações, geralmente não-dependentes de arquitetura.
/usr/src	Código-fonte, normalmente para o kernel do Linux
/usr/X11R6	Arquivos de configuração do sistema X Window. Geralmente, obsoletos.
/usr/local	Dados e programas específicos da máquina local. Alguns subdiretórios incluem bin, sbin, lib, share, include, entre outros.
/usr/bin	Este é o diretório primário dos comandos executáveis no sistema.

/var – Arquivos de Dados Variáveis

O diretório **/var**é a contraparte com permissão de escrita do diretório /usr, o qual deve ser somente-leitura quando em operação normal. Arquivos de log e tudo o mais que normalmente seria escrito em /usr durante a operação normal são escritos no diretório /var. Por exemplo, encontramos arquivos de log no diretório /var/log.

O diretório /var é onde o sistema armazena seus arquivos de spool, como o spool de impressão, filas de entrada e saída de email e arquivos de log do sistema, entre outros.

Por conta disso, esses arquivos podem aumentar e diminuir de tamanho drasticamente e sem aviso.

Ele também contém arquivos cujo tamanho e conteúdo podem mudar enquanto o sistema está em execução, como as entradas nos diretórios a seguir:

- Arquivos de log do sistema: /var/log
- Pacotes e arquivos de banco de dados: /var/lib
- Filas de impressão: /var/spool
- Arquivos temporários: /var/tmp

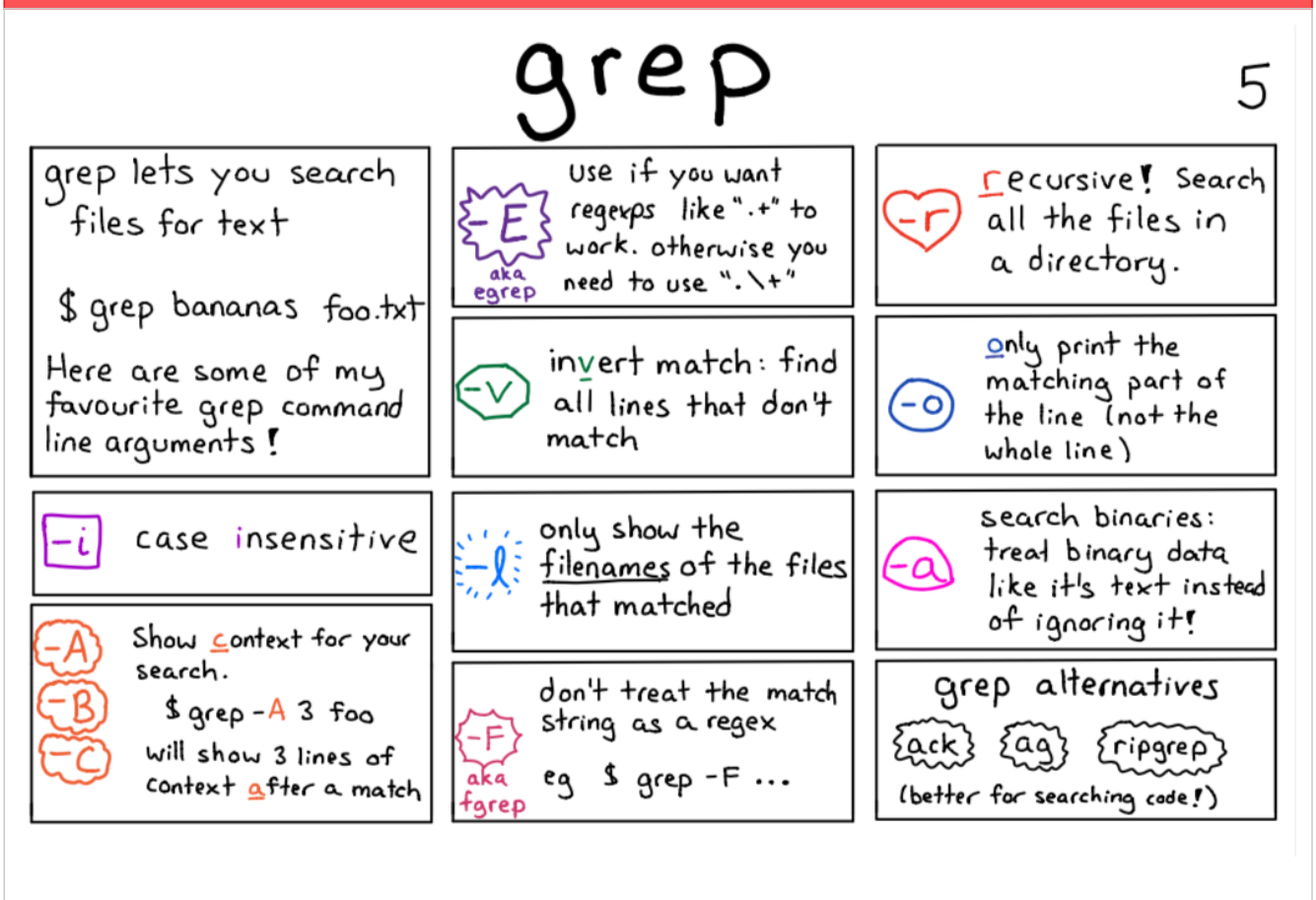
O diretório /var pode ser colocado em seu próprio sistema de arquivos de modo que o crescimento dos arquivos possa ser acomodado e os tamanhos dos arquivos não afetem o sistema de forma grave. Diretórios de serviços de rede como **/var/ftp**(serviço de FTP) e **/var/www**(serviço web HTTP) também são encontrados dentro de /var.

http://www.bosontreinamentos.com.br/linux/a-estrutura-de-diretorios-do-linux/

2.0 LINHAS DE COMANDO

greep, find, xargs, awk, sed, bask tricks, disk usage, tar, ps, top, sort&uniq, misc commands, head & tail, less, kill, cat, lsof

greep



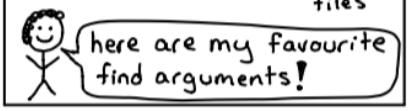
find

find 6

find searches a directory for files

```
find /tmp -type d -print
```

directory to search | which files | action to do with the files



-name/-iname

case insensitive

the filename! eg

```
-name '*.txt'
```

-type [TYPE]

f: regular file
d: directory
l: symlink
+ more!

-path / -ipath

search the full path!

```
-path '/home/*/*.go'
```

-maxdepth NUM

only descend NUM levels when searching a directory

-size 0

find empty files!
Useful to find files you created by accident

-print0

print null-separated filenames
Use with xargs -0!

locate

The locate command searches a database of every file on your system.

good: faster than find
bad: can get out of date

```
$sudo updatedb
```

updates the database

-exec COMMAND

action: run COMMAND on every file found

-delete

action: delete all files found

xargs

xargs 7

xargs takes whitespace separated strings from stdin and converts them into command-line arguments

```
$echo "/home /tmp" | xargs ls
```

will run

```
ls /home /tmp
```

this is useful when you want to run the same command on a list of files!

- → delete (xargs rm)
- → combine (xargs cat)
- → search (xargs grep)
- → replace (xargs sed)

how to replace "foo" with "bar" in all .txt files:

```
find . -name '*.txt' | xargs sed -i s/foo/bar/g
```

how to lint every Python file in your Git repo:

```
git ls-files | grep .py | xargs pep8
```

if there are spaces in your filenames "my day.txt"

xargs will think it's 2 files "my" and "day.txt"

fix it like this:

```
find . -print0 | xargs -0 COMMAND
```

more useful xargs options

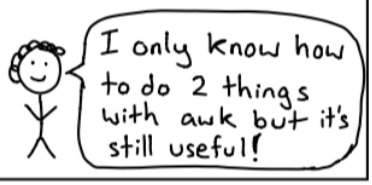
`-n 1` (max-args) makes xargs run a separate process for each input.

`-P` (capital P, max-procs) is the max number of parallel processes xargs will start

awk

awk 8

awk is a tiny programming language for manipulating columns of data



basic awk program structure

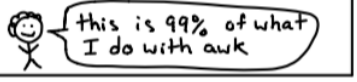
```
BEGIN { ... }
CONDITION {action}
CONDITION {action}
END {...}
```

do action on lines matching CONDITION

extract a column of text with awk

```
awk -F, '{print $5}'
```

column separator; single quotes!; print the 5th column



SO MANY unix commands print columns of text (ps! ls!)

so being able to get the column you want with awk is GREAT

awk program example:
Sum the numbers in the 3rd column

```
{s += $3};
END {print s}'
```

action

at the end, print the sum!

awk program example:
print every line over 80 characters

```
length($0) > 80
```

condition

(there's an implicit {print} as the action)

sed

sed 9

sed is most often used for replacing text in a file

```
$ sed s/cat/dog/g file.txt
```

can be a regular expression

change a file in place with -i

some more sed incantations...

sed -n 12 p
print 12th line

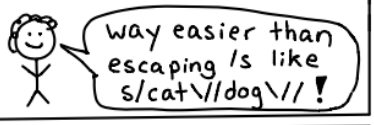
-n suppresses output so only what you print with 'p' gets printed

sed 5d
delete 5th line

sed /cat/d
delete lines matching /cat/

sed -n 5,30 p
print lines 5-30

sed s+cat/+dog/+
+ can be any character
use + as a regex delimeter



sed -n s/cat/dog/p
only print changed lines

sed G
double space a file (good for long error lines)

sed '/cat/a dog'
append 'dog' after lines containing 'cat'

sed '17i panda'
insert "panda" on line 17

bask tricks

bash tricks 10

★ ctrl + r ★

search your history!

I use this ♥ constantly ♥ to rerun commands

★ magical braces ★

```
$ convert file.{jpg,png}
```

expands to

```
$ convert file.jpg file.png
```

{1..5} expands to 1 2 3 4 5

(for i in {1..100} ...)

!!

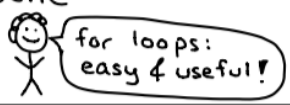
expands to the last command run

```
$ sudo !!
```

commands that start with a space don't go in your history. good if there's a 🔒 password 🔒

loops

```
for i in *.png
do
  convert $i $i.jpg
done
```



$( )

gives the output of a command

```
$ touch file-$(date -I)
```

create a file named file-2018-05-25

more keyboard shortcuts

ctrl a	beginning of line
ctrl+e	end of line
ctrl+l	clear the screen

+ lots more emacs shortcuts too!

more bash tricks 11

cd -

changes to the directory you were last in

pushd & popd let you keep a stack

ctrl+z

suspends (SIGTSTP) the running program

bg

starts suspended program & backgrounds it (use after ctrl+z)

fg

brings backgrounded/suspended program to the foreground

♡ shellcheck ♡

shell script linter! helps spot common mistakes.

<( )

process substitution

treat process output like a file (no more temp files!)

eg:

```
$ diff <(ls) <(ls -a)
```

fc

"fix command"

open the last command you ran in an editor

then run the edited version

type

tells you if something is a builtin, program, or alias

try running type on

{time} [ping] (pushd)

(they're all different types)

disk usage

disk usage 12

du

tells you how much disk space files / directories take up

-s summary: total size of all files in a directory

-h human readable sizes

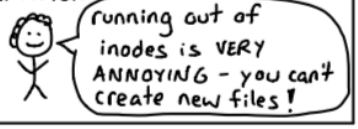
df

tells you how much free space each partition has. -h for human-readable sizes

```
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda3        18G     G  2.5G  86% /
udev            483M  4.0K  483M   1% /dev
tmpfs            99M  1.4M   97M   2% /run
/dev/sda4       167G  157G  9.9G  95% /home
```

df -i

instead of % disk free, report how many inodes are used / free on each partition



ncdu

see what's using disk space navigate with arrow keys

```
17.5 GiB [####] /music
 3.2 GiB [##  ] /pictures
 5.7 MiB [    ] /text
 2.0 MiB [    ] file.pdf
```

iostat

get statistics about disk reads / writes

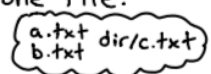
interval to report at

```
# iostat 5
Device:  kB_read/s  kB_wrtn/s
sda        2190.21     652.87
sdb           6.00       0.00
```

tar

tar 13

The .tar file format combines many files into one file.

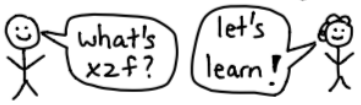


.tar files aren't compressed by themselves. Usually you gzip them: .tar.gz or .tgz!

Usually when you use the 'tar' command, you'll run some incantation

To unpack a tar.gz, use:

```
tar -xzf file.tar.gz
```



-x is for extract

into the current directory by default (change with -C)

-c is for create

makes a new tar file!

-t is for list

lists the contents of a tar archive

-f is for file

which tar file to create or unpack

tar can compress / decompress

-z gzip format (.gz)

-j bzip2 format (.bz2)

-J xz format (.xz)

& more! see the man page ☺

putting it together

list contents of a .tar.bz2:

```
$ tar -tjvf file.tar.bz2
```

(v: verbose)

create a .tar.gz

```
$ tar -czf file.tar.gz dir/
```

(dir/: files to go in the archive)

ps

ps

ps shows which processes are running

I usually run ps like this:

$ ps aux

u means include username column

a+x together show all process

(ps -ef works too)

w

is for wide. ps auxwww will show all the command line args for each process

e

is for environment. ps auxe will show the environment vars!

wchan

you can choose which columns to show with ps (ps -eo ...)

One cool column is 'wchan' which tells you the name of the Kernel function if the process is sleeping

try it:

ps -eo user,pid,wchan,cmd

★ process state ★

Here's what the letters in ps's STATE column mean:

R: running
S/D: asleep
Z: zombie
l: multithreaded
+: in the foreground

f

is for "forest" 🌲. ps auxf will show you an ASCII art process tree!

pstree can display a process tree too

ps has 3 different sets of command line arguments 💔

1. UNIX (1 dash)
2. BSD (no dash)
3. GNU (2 dashes)

you can write monstrosities like:

$ ps f -f

forest (BSD)

full format (UNIX)

top

top

a live-updating summary of the top users of your system's resources

let's explain some numbers in top!

load average

3 numbers that roughly reflect demand for your CPUs on the system in the last 1, 5, and 15 minutes

if it's higher than the # of CPUs you have, that's often bad

memory

4 numbers:

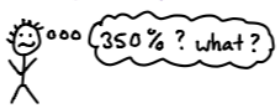
total / free / used / cached

One perhaps unexpected thing: total is not free + used!

total = free + used + cached

filesystem cache

% CPU



this column is given as % of a single core. If you have 4 cores, this can go up to 400%!

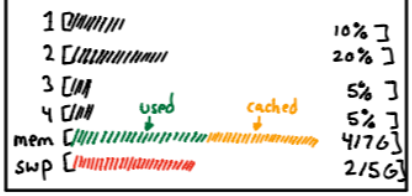
RES

this column is the "resident set size" aka how much RAM your process is using.

SHR is how much of the RES is shared with other processes

htop

a prettier & more interactive version of top ★



sort&uniq

sort & uniq 16

sort sorts its input

```
$ sort names.txt
```

the default sort is alphabetical.

sort -n
numeric sort

'sort' order	'sort -n' order
12	12
15000	48
48	96
6020	6020
96	15000

sort -h: human sort

'sort -n' order	'sort -h' order
15G	45K
30M	30M
45K	15G
200G	200G

useful example:
`du -sh * | sort -h`

uniq removes duplicates

```
a
b
b   =>   a b a c   (notice there are still 2 'a's! uniq only uniquifies adjacent matching lines)
a
c
c
```

sort + uniq = ♥

Pipe something to 'sort | uniq' and you'll get a deduplicated list of lines! `sort -u` does the same thing.

```
b
a   | sort -u =>  a b
b      (or sort | uniq)
a
```

uniq -c
counts each line it saw.

Recipe: get the top 10 most common lines in a file:

```
$ sort foo.txt
  | uniq -c
  | sort -n
  | tail -n 10
```

misc

misc commands ♥ 17

rlwrap
adds history & ctrl support to REPLs that don't already have them (rl stands for readline)

```
$ rlwrap python
```

watch
rerun a command every 2 seconds

pv
"pipe viewer", gives you stats on data going through a pipe

file
figures out what kind of file (png? pdf?) a file is

cal
a tiny calendar ♡

ts
add a timestamp in front of every input line

ncdu
figure out what's using all your disk space

diff
diff 2 files. Run with '-U 8' for context.

comm
find lines 2 sorted files have in common

column
format input into columns

xsel / xclip
copy/paste from system clipboard.
(pbcopy/pbpaste on Mac)

commands

misc commands ♥ 17

rlwrap

adds history & ctrl support to REPLs that don't already have them (rl stands for readline)

$ rlwrap python

watch

rerun a command every 2 seconds

pv

"pipe viewer", gives you stats on data going through a pipe

file

figures out what kind of file (png? pdf?) a file is

cal

a tiny calendar ♡

ts

add a timestamp in front of every input line

ncdu

figure out what's using all your disk space

diff

diff 2 files. Run with '-U 8' for context.

comm

find lines 2 sorted files have in common

column

format input into columns

xsel / xclip

copy/paste from system clipboard. (pbcopy/pbpaste on Mac)

head&tail

head & tail 18

head

shows you the first 10 lines of a file

if you pipe a program's output to head, the program will stop after printing 10 lines (it gets sent SIGPIPE)

tail

tail shows the last 10 lines!

`tail -f FILE` will follow:

print any new lines added to the end of FILE. Super useful for log files!

-n NUM

-n NUM (either head or tail) will change the # lines shown

head -n -NUM } show all
tail -n +NUM } but the last / first NUM lines

-c NUM

show the first/last NUM bytes of the file

head -c 1K

tail --retry

keep trying to open file if it's inaccesible

tail --pid PID

tail --follow=name

Usually tail -f will follow a file descriptor.

tail --follow=name FILENAME
Will keep following the same

will show the first 1024 bytes

stop when process PID stops running (with -f)

filename, eg if the file descriptor is rotated.

less

less 19

less is a pager

that means it lets you view (not edit) text files or piped in text

man uses your pager (usually less) to display man pages

many vim shortcuts work in less

/	search
n / N	next / prev match
j / k	down / up a line
m / '	mark / return to line
g / G	beginning / end of file

(gg in vim)

less -r

displays bash escape codes as colours

try ls --color | less -r

with -r	without -r
a.txt	a.txt
a.txt.gz (red, bold)	ESC[0m ESC [01;31m a.txt.gz ESC[0m

ugh

q

quit

V ← lowercase

edit file in your $EDITOR

arrow keys, Home/End, PgUp, PgDn work in less

F

press F to keep reading from the file as it's updated (like tail -f)

press Ctrl+C to stop reading updates

+

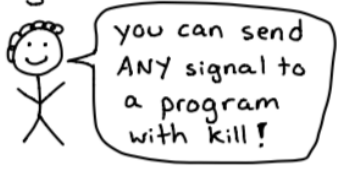
+ runs a command when less starts

less +F	follow updates
less +G	start at end of file
less +20%	start 20% into file
less +/foo	search for 'foo' right away

kill

kill 20

kill doesn't just kill programs



kill -SIGNAL PID

name or number

which signal kill sends

		name	num
kill	=>	SIGTERM	15
kill -9 / kill -KILL	=>	SIGKILL	9
kill -HUP	=>	SIGHUP	1
kill -STOP	=>	SIGSTOP	19

kill -l lists all signals

1 HUP	2 INT	3 QUIT	4 ILL
5 TRAP	6 ABRT	7 BUS	8 FPE
9 KILL	10 USR1	11 SEGV	12 USR2
13 PIPE	14 ALRM	15 TERM	16 STKFLT
17 CHLD	18 CONT	19 STOP	20 TSTP
21 TTIN	22 TTOU	23 URG	24 XCPU
25 XFSZ	26 VTALRM	27 PROF	28 WINCH
29 POLL	30 PWR	31 SYS	

killall -SIGNAL NAME

signals all processes called NAME for example

$ killall firefox

pgrep

prints PIDs of matching running programs

pgrep fire matches firefox firebird

pkill

same as pgrep, but signals PIDs found. ex:

pkill -f firefox

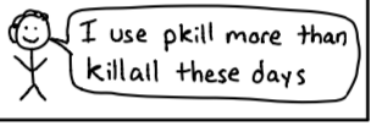
useful flags:

-w wait for all signaled processes to die

-i ask before signalling

NOT bash firefox.sh

To search the whole command line (eg bash firefox.sh) use pgrep -f



cat

cat & friends 21

cat concatenates files

$ cat myfile.txt
prints contents of myfile.txt

$ cat *.txt
prints all .txt files put together!

you can use cat as an EXTREMELY BASIC text editor:

① Run $ cat > file.txt
② type the contents (don't make mistakes ☺)
③ press ctrl+d to finish

cat -n

prints out the file with line numbers!

1 Once upon a midnight..
2 Over many a quaint.
3 While I nodded, nearly

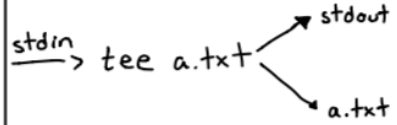
zcat

cats a gzipped file!

Actually just a 1-line shell script that runs 'gzip -cd', but easier to remember.

tee

'tee file.txt' will write its stdin to both stdout and file.txt



how to redirect to a file owned by root

$ sudo echo "hi" >> x.txt

this will open x.txt as your user, not as root, so it fails!

$ echo "hi" | sudo tee -a x.txt
will open x.txt as root ☺

lsof

lsof 22

lsof

stands for list open files

what lsof tells you

for each open file:
- pid
- file type (regular? directory? FIFO? socket?)
- file descriptor (FD column)
- user
- filename / socket address

-p PID

list the files PID has open

lsof /some/dir

list just the open files in /some/dir

-i

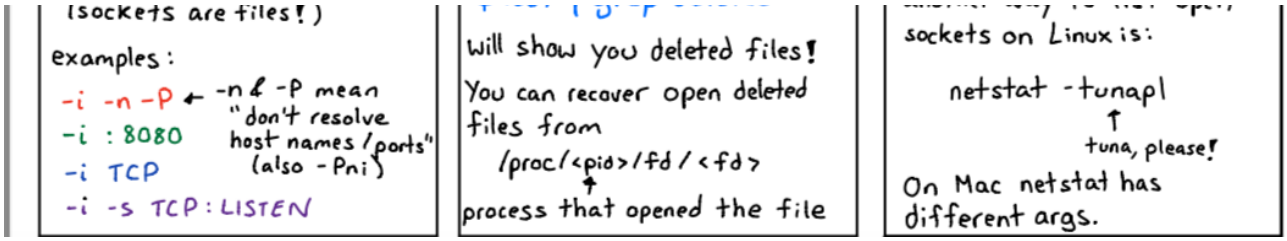
list open network sockets

find deleted files

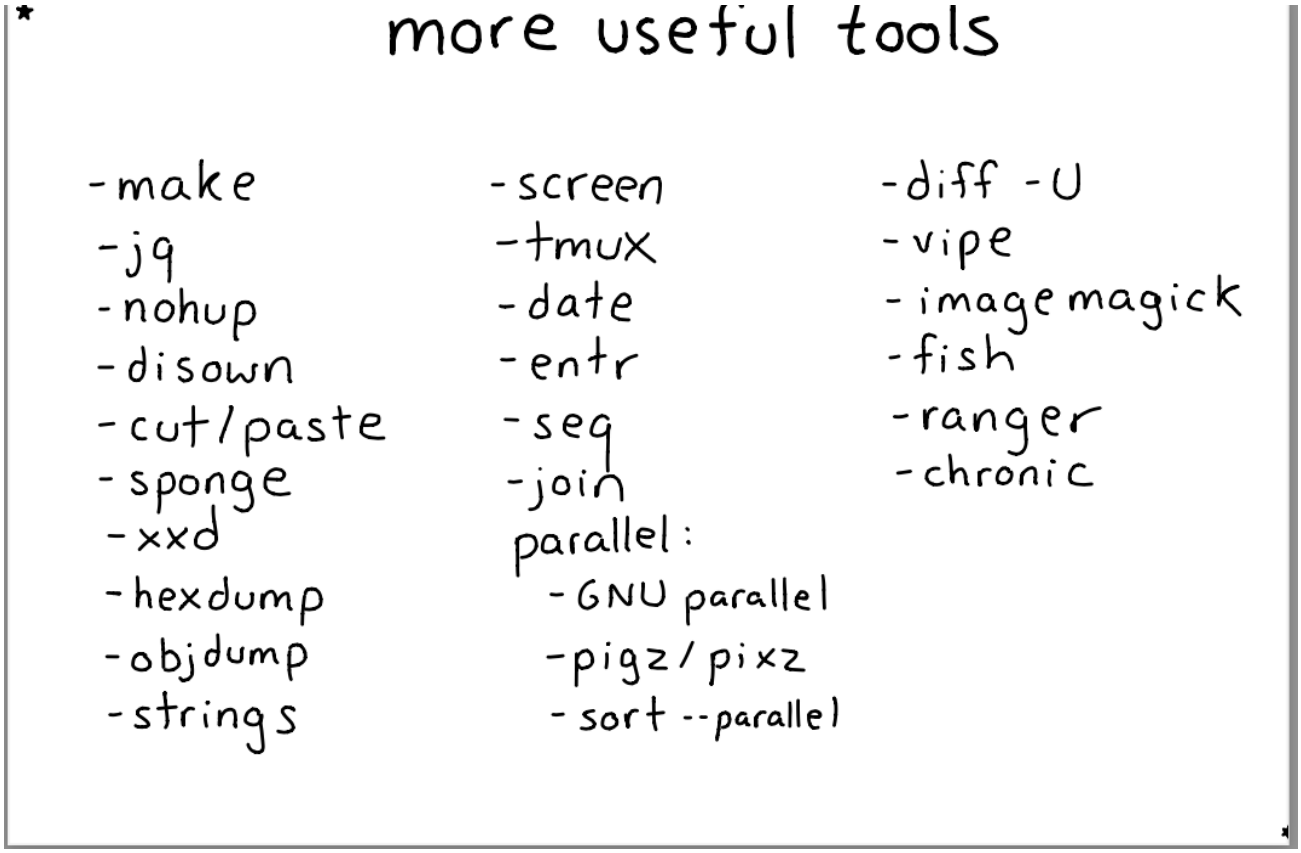
$ lsof | grep deleted

netstat

another way to list open



more



pwd
cp
chmod
diff
more
-mv
-rm
-mkdir
help
history
pico
-ps
defaults
say
alloc
cal
contrab

-clear
dc
curl
cut
ditto
last
lpr
mdfind
open
passwd
-ping

tar

ps

top

sort & unid

misc commands

head&tail

less

kill

cat

lsof

HTML Content

A folha de dicas dos comandos do terminal Mac

POR RAHUL SAIGAL

PUBLICADO EM 07 DE DEZEMBRO DE 2019

Nossa mega folha de dicas de comandos do terminal Mac fornece uma ótima referência para todos os comandos importantes que você deve saber.

mac-terminal-cheatsheet

macOS é um sistema operacional intuitivo, então você não precisa perder muito tempo aprendendo o básico; Sabendo disso, por que você deveria aprender e aproveitar as vantagens da linha de comando Unix disponível em seu Mac? Temos quatro boas razões:

1. Existem dezenas de aplicativos baseados em Unix de código aberto e disponíveis gratuitamente. Você não precisa gastar dinheiro com isso.
2. Quando estiver tendo dificuldade para pesquisar arquivos no Spotlight, você pode recorrer às ferramentas de pesquisa do Unix. Eles são muito mais poderosos do que o Spotlight.

3. Você pode gerenciar arquivos, pastas e arquivamentos de arquivos de maneira automatizada. A configuração de um cron job resolverá isso automaticamente.
4. Dá a você mais poder e controle sobre seu sistema.

Com tantos comandos do Mac, muitas vezes é difícil lembrar e usar todos eles. Estamos aqui para ajudar com uma folha de dicas detalhada dos comandos do Terminal Mac que você pode usar para desbloquear a produtividade aprimorada em seu sistema.

Inicie o aplicativo Terminalem **Aplicativos> Utilitários**ou pesquise por meio do Spotlight. Então você pode começar com alguns dos comandos poderosos abaixo.

DOWNLOAD GRATUITO:esta folha de dicas está disponível como um **PDF para download**em nosso parceiro de distribuição, TradePub. Você terá que preencher um pequeno formulário para acessá-lo apenas pela primeira vez. Baixe a **folha de dicas dos comandos do Mac Terminal**.

A folha de dicas dos comandos do terminal Mac

Comando	Açao
Atalhos	
Aba	Autocompletar nomes de arquivos e pastas

Comando	Açao
Ctrl + A	Vai para o início da linha em que você está digitando
Ctrl + E	Vá para o final da linha em que você está digitando
Ctrl + U	Limpe a linha antes do cursor
Ctrl + K	Limpe a linha após o cursor
Ctrl + W	Exclua a palavra antes do cursor
Ctrl + T	Troque os dois últimos caracteres antes do cursor
Esc + T	Troque as duas últimas palavras antes do cursor
Ctrl + L	Limpar a tela
Ctrl + C	Mate tudo o que você está executando
Ctrl + D	Sai do shell atual
Opção + →	Mova o cursor uma palavra para a frente
Opção + ←	Mova o cursor uma palavra para trás
Ctrl + F	Mova o cursor um caractere para a frente
Ctrl + B	Move o cursor um caractere para trás
Ctrl + Y	Cole o que foi cortado pelo último comando
Ctrl + Z	Coloca tudo o que você está executando em um processo suspenso em segundo plano
Ctrl + _	Desfaz o último comando
Fundamentos	

Comando	Açao
/ (Barra para frente)	Diretório de nível superior
. (Período Único)	Diretório atual
.. (período duplo)	Diretório Parental
~ (Til)	Diretório inicial
sudo [comando]	Execute o comando com os privilégios de segurança do superusuário
nano [arquivo]	Abre o editor do Terminal
abrir arquivo]	Abre um arquivo
[comando] -h	Obtenha ajuda sobre um comando
man [comando]	Mostra o manual de ajuda do comando
Alterar diretório	
CD	Diretório inicial
cd [pasta]	Mudar de diretório, por exemplo, Documentos de cd
cd ~	Diretório inicial
CD/	Raiz da unidade
CD -	Diretório ou pasta anterior em que você navegou pela última vez
pwd	Mostre seu diretório de trabalho
CD..	Mover para o diretório pai
CD../..	Suba dois níveis
Listar conteúdo do diretório	

Comando	Açao
ls	Mostra o nome dos arquivos e subdiretórios do diretório
ls -C	Forçar saída de várias colunas da listagem
ls -a	Liste todas as entradas, incluindo aquelas com. (Ponto final) e .. (ponto final duplo)
ls -1	Produz a lista de arquivos em uma entrada por formato de linha
ls -F	Exibe uma / (barra) imediatamente após cada caminho que é um diretório, * (asterisco) após programas executáveis ou scripts e @ após um link simbólico
ls -S	Classifique arquivos ou entradas por tamanho
ls -l	Liste em um formato longo. Inclui modo de arquivo, nome do proprietário e do grupo, data e hora em que o arquivo foi modificado, nome do caminho e muito mais
ls -lt	Liste os arquivos classificados por tempo de modificação (mais recentes primeiro)
ls -lh	Longa listagem com tamanhos de arquivo legíveis por humanos em KB, MB ou GB
ls -lo	Liste os nomes dos arquivos com tamanho, proprietário e sinalizadores
ls -la	Lista o conteúdo detalhado do diretório, incluindo arquivos ocultos

Comando	Açao
Tamanho do arquivo e espaço em disco	
du	Liste o uso de cada subdiretório e seu conteúdo
du -sh [pasta]	Saída legível por humanos de todos os arquivos em um diretório
du -s	Mostra uma entrada para cada arquivo especificado
du -sk * l sort -nr	Liste arquivos e pastas, totalizando o tamanho, incluindo as subpastas. Substitua sk * por sm * para listar os diretórios em MB
df -h	Calcule o espaço livre em disco do seu sistema
df -H	Calcule o espaço livre em disco em potências de 1.000 (em oposição a 1.024)
Gerenciamento de arquivos e diretórios	
mkdir <dir>	Crie uma nova pasta chamada <dir>
mkdir -p <dir> / <dir>	Crie pastas aninhadas
mkdir <dir1> <dir2> <dir3>	Crie várias pastas de uma vez
mkdir "<dir>"	Crie uma pasta com um espaço no nome do arquivo
rmdir <dir>	Excluir uma pasta (funciona apenas em pastas vazias)
rm -R <dir>	Exclua uma pasta e seu conteúdo
toque em <arquivo>	Crie um novo arquivo sem qualquer extensão

Comando	Açao
cp <file> <dir>	Copie um arquivo para a pasta
cp <file> <newfile>	Copie um arquivo para a pasta atual
cp <file> ~ / <dir> / <newfile>	Copie um arquivo para a pasta e renomeie o arquivo copiado
cp -R <dir> <"novo dir">	Copie uma pasta para uma nova pasta com espaços no nome do arquivo
cp -i <file> <dir>	Avisa antes de copiar um arquivo com uma mensagem de aviso de substituição
cp <file1> <file2> <file3> / Users / <dir>	Copie vários arquivos para uma pasta
rm <arquivo>	Excluir um arquivo (isso exclui o arquivo permanentemente; use com cuidado.)
rm -i <arquivo>	Exclua um arquivo apenas quando você der a confirmação
rm -f <arquivo>	Forçar remoção sem confirmação
rm <arquivo1> <arquivo2> <arquivo3>	Exclua vários arquivos sem qualquer confirmação
mv <file> <newfilename>	Mover / renomear
mv <arquivo> <dir>	Mova um arquivo para a pasta, possivelmente substituindo um arquivo existente
mv -i <file> <dir>	Opcional -i sinalizador para avisá-lo antes de sobrescrever o arquivo

Comando	Açao
mv * .png ~ / <dir>	Mova todos os arquivos PNG da pasta atual para uma pasta diferente
Histórico de Comandos	
Ctrl + R	Pesquise os comandos usados anteriormente
história n	Mostra os comandos anteriores que você digitou. Adicione um número para limitar os últimos n itens
![valor]	Execute o último comando digitado que começa com um valor
!!	Execute o último comando digitado
Permissões	
ls -ld	Exibir a permissão padrão para um diretório home
ls -ld / <dir>	Exibir a permissão de leitura, gravação e acesso de uma pasta específica
chmod 755 <file>	Altere a permissão de um arquivo para 755
chmod -R 600 <dir>	Altere a permissão de uma pasta (e seu conteúdo) para 600
chown <usuário>: <grupo> <arquivo>	Altere a propriedade de um arquivo para usuário e grupo. Adicione -R para incluir o conteúdo da pasta
Processos	
ps -ax	Saída de processos atualmente em execução. Aqui, a mostra os processos de todos os usuários ex mostra os processos que não estão conectados ao Terminal

Comando	Açao
ps -aux	Mostra todos os processos com% cpu,% mem, entrada de página, PID e comando
principal	Exibir informações ao vivo sobre os processos em execução
top -ocpu -s 5	Exibir processos classificados por uso de CPU, atualizando a cada 5 segundos
top -o rsize	Classifique os principais por uso de memória
matar PID	Saia do processo com ID <PID>. Você verá o PID como uma coluna no Monitor de Atividade
ps -ax l grep <appname>	Encontre um processo por nome ou PID
Rede	
ping <host>	Fazer ping do host e exibir o status
whois <domain>	Informações de saída de whois para um domínio
curl -O <url / para / arquivo>	Baixe o arquivo via HTTP, HTTPS ou FTP
ssh <nome de usuário> @ <host>	Estabeleça conexão SSH para <host> com o usuário <username>
scp <file> <user> @ <host>: / remote / path	Copie <file> para um <host> remoto
Homebrew	

Comando	Açao
doutor de cerveja	Verifique a mistura em busca de problemas em potencial
brew install <formula>	Instale uma fórmula
brew uninstall <formula>	Desinstalar uma fórmula
lista de cerveja	Liste todas as fórmulas instaladas
busca de cerveja	Exibir fórmulas disponíveis para cerveja
atualização de cerveja	Atualize todas as cervejas desatualizadas e não fixadas
atualização de cerveja	Obtenha a versão mais recente do homebrew e da fórmula
limpeza de cerveja	Remova a versão mais antiga da fórmula instalada
torneira de fermentação homebrew / barril	Toque no repositório do barril do GitHub
lista de barris de cerveja	Liste todos os barris instalados
brew cask install <cask>	Instale o barril dado
desinstalação do brew cask <cask>	Desinstale o barril fornecido
Procurar	

Comando	Açao
find <dir> -name <"file">	Encontre todos os arquivos com o nome <arquivo> dentro de <dir>. Use curingas (*) para pesquisar partes de nomes de arquivos
grep "<text>" <file>	Exibir todas as ocorrências de <texto> dentro de <arquivo> (adicionar -i para não diferenciar maiúsculas de minúsculas)
grep -rl "<texto>" <dir>	Pesquise todos os arquivos contendo <texto> dentro de <dir>
Saída	
cat <file>	Produza o conteúdo de <file>
menos <file>	Produza o conteúdo de <file> usando o comando less que suporta paginação e mais
head <file>	Produza as primeiras 10 linhas de <file>
<cmd>>> <arquivo>	Anexa a saída de <cmd> a <file>
<cmd>> <arquivo>	Direcione a saída de <cmd> para <file>
<cmd1> l <cmd2>	Direcione a saída de <cmd1> para <cmd2>

Em seguida, personalize o terminal

Existem muitos comandos nesta folha de dicas. Mas você não precisa aprender todos eles de uma vez! Escolha alguns que se integrem bem ao seu fluxo de trabalho e economizem mais tempo. Depois de

dominar esses comandos, ainda há mais para aprender sobre o Terminal para aprimorar sua experiência com ele.

Para ler mais, vimos **como personalizar o Terminal Mac e torná-lo mais útil**.

HTML Content

The Mac Terminal Commands Cheat Sheet

BY RAHUL SAIGAL

PUBLISHED DEC 07, 2019

Our mega cheat sheet of Mac terminal commands provides a great reference for all the important commands you should know.

mac-terminal-cheatsheet

macOS is an intuitive operating system, so you don't have to spend lot of time learning the basics; Knowing this, why should you learn and take advantage of the Unix command line available on your Mac? We have four good reasons:

1. There are dozens of open source and freely available Unix-based apps. You don't have to spend money on these.
2. When you're having difficulty searching for files in Spotlight, you can turn to Unix search tools. They're way more powerful than Spotlight.

3. You can manage files, folders, and file archives in an automated manner. Setting up a cron job will handle this automatically.
4. It gives you more power and control over your system.

With so many Mac commands, it's often difficult to remember and use them all. We're here to help with a detailed cheat sheet of Mac Terminal commands you can use to unlock enhanced productivity on your system.

Launch the Terminal appfrom**Applications > Utilities**or search for it via Spotlight. Then you can get started with some of the powerful commands below.

FREE DOWNLOAD: This cheat sheet is available as a **downloadable PDF**from our distribution partner, TradePub. You will have to complete a short form to access it for the first time only. Download **The Mac Terminal Commands Cheat Sheet**.

The Mac Terminal Commands Cheat Sheet

Command	Action
Shortcuts	
Tab	Auto-complete file and folder names
Ctrl + A	Go to the beginning of the line you're currently typing on

Command	Action
Ctrl + E	Go to the end of the line you're currently typing on
Ctrl + U	Clear the line before the cursor
Ctrl + K	Clear the line after the cursor
Ctrl + W	Delete the word before the cursor
Ctrl + T	Swap the last two characters before the cursor
Esc + T	Swap the last two words before the cursor
Ctrl + L	Clear the screen
Ctrl + C	Kill whatever you're running
Ctrl + D	Exit the current shell
Option + →	Move cursor one word forward
Option + ←	Move cursor one word backward
Ctrl + F	Move cursor one character forward
Ctrl + B	Move cursor one character backward
Ctrl + Y	Paste whatever was cut by the last command

Command	Action
Ctrl + Z	Puts whatever you're running into a suspended background process
Ctrl + _	Undo the last command
Basics	
/ (Forward Slash)	Top level directory
. (Single Period)	Current directory
.. (Double Period)	Parent directory
~ (Tilde)	Home directory
sudo [command]	Run command with the security privileges of the super user
nano [file]	Opens the Terminal editor
open [file]	Opens a file
[command] -h	Get help about a command
man [command]	Show the help manual of the command
Change Directory	
cd	Home directory
cd [folder]	Change directory, e.g. cd Documents
cd ~	Home directory
cd/	Root of the drive
cd -	Previous directory or folder you last browsed

Command	Action
pwd	Show your working directory
cd..	Move up to the parent directory
cd../..	Move up two levels
List Directory Contents	
ls	Display the name of files and subdirectories in the directory
ls -C	Force multi-column output of the listing
ls -a	List all entries including those with .(period) and ..(double period)
ls -1	Output the list of files in one entry per line format
ls -F	Display a / (slash) immediately after each path that is a directory, * (asterisk) after executable programs or scripts, and @ after a symbolic link
ls -S	Sort files or entries by size
ls -l	List in a long format. Includes file mode, owner and group name, date and time file was modified, pathname, and more

Command	Action
ls -lt	List the files sorted by time modified (most recent first)
ls -lh	Long listing with human readable file sizes in KB, MB, or GB
ls -lo	List the file names with size, owner, and flags
ls -la	List detailed directory contents, including hidden files
File Size and Disk Space	
du	List usage for each subdirectory and its contents
du -sh [folder]	Human readable output of all files in a directory
du -s	Display an entry for each specified file
du -sk* | sort -nr	List files and folders, totaling the size including the subfolders. Replace sk* with sm* to list directories in MB
df -h	Calculate your system's free disk space
df -H	Calculate free disk space in powers of 1,000 (as opposed to 1,024)
File and Directory Management	
mkdir <dir>	Create new folder named <dir>

Command	Action
mkdir -p <dir>/<dir>	Create nested folders
mkdir <dir1> <dir2> <dir3>	Create several folders at once
mkdir "<dir>"	Create a folder with a space in the filename
rmdir <dir>	Delete a folder (only works on empty folders)
rm -R <dir>	Delete a folder and its contents
touch <file>	Create a new file without any extension
cp <file> <dir>	Copy a file to the folder
cp <file> <newfile>	Copy a file to the current folder
cp <file>~/<dir>/<newfile>	Copy a file to the folder and rename the copied file
cp -R <dir> <"new dir">	Copy a folder to a new folder with spaces in the filename
cp -i <file><dir>	Prompts you before copying a file with a warning overwrite message
cp <file1> <file2> <file3>/Users/<dir>	Copy multiple files to a folder
rm <file>	Delete a file (This deletes the file permanently; use with caution.)
rm -i <file>	Delete a file only when you give confirmation

Command	Action
rm -f <file>	Force removal without confirmation
rm <file1> <file2> <file3>	Delete multiple files without any confirmation
mv <file> <newfilename>	Move/rename
mv <file> <dir>	Move a file to the folder, possibly by overwriting an existing file
mv -i <file> <dir>	Optional -i flag to warn you before overwriting the file
mv *.png ~/<dir>	Move all PNG files from current folder to a different folder
Command History	
Ctrl + R	Search through previously used commands
history n	Shows the previous commands you've typed. Add a number to limit to the last n items
![value]	Execute the last command typed that starts with a value
!!	Execute the last command typed
Permissions	
ls -ld	Display the default permission for a home directory

Command	Action
ls -ld/<dir>	Display the read, write, and access permission of a particular folder
chmod 755 <file>	Change the permission of a file to 755
chmod -R 600 <dir>	Change the permission of a folder (and its contents) to 600
chown <user>:<group> <file>	Change the ownership of a file to user and group. Add -R to include folder contents
Processes	
ps -ax	Output currently running processes. Here, a shows processes from all users and x shows processes that are not connected with the Terminal
ps -aux	Shows all the processes with %cpu, %mem, page in, PID, and command
top	Display live information about currently running processes
top -ocpu -s 5	Display processes sorted by CPU usage, updating every 5 seconds
top -o rsize	Sort top by memory usage

Command	Action
kill PID	Quit process with ID <PID>. You'll see PID as a column in the Activity Monitor
ps -ax | grep <appname>	Find a process by name or PID
Network	
ping <host>	Ping host and display status
whois <domain>	Output whois info for a domain
curl -O <url/to/file>	Download file via HTTP, HTTPS, or FTP
ssh <username>@<host>	Establish SSH connection to <host> with user <username>
scp <file> <user>@<host>:/remote/path	Copy <file> to a remote <host>
Homebrew	
brew doctor	Check brew for potential problems
brew install <formula>	Install a formula
brew uninstall <formula>	Uninstall a formula
brew list	List all the installed formulas
brew search	Display available formulas for brewing
brew upgrade	Upgrade all outdated and unpinned brews
brew update	Fetch latest version of homebrew and formula

Command	Action
brew cleanup	Remove older version of installed formula
brew tap homebrew/cask	Tap the cask repository from GitHub
brew cask list	List all installed casks
brew cask install <cask>	Install the given cask
brew cask uninstall <cask>	Uninstall the given cask
Search	
find <dir> -name <"file">	Find all files named <file> inside <dir>. Use wildcards (*) to search for parts of filenames
grep "<text>" <file>	Output all occurrences of <text> inside <file> (add -i for case insensitivity)
grep -rl "<text>" <dir>	Search for all files containing <text> inside <dir>
Output	
cat <file>	Output the content of <file>
less <file>	Output the contents of <file> using the less command that supports pagination and more
head <file>	Output the first 10 lines of <file>
<cmd> > > <file>	Appends the output of <cmd> to <file>
<cmd> > <file>	Direct the output of <cmd> into <file>

Command	Action
<cmd1> | <cmd2>	Direct the output of <cmd1> to <cmd2>

Next, Customize the Terminal

There are lot of commands in this cheat sheet. But you don't have to learn all of them at once! Pick a few that integrate well with your workflow and save you the most time. Once you've mastered these commands, there's still more to learn about the Terminal to enhance your experience with it.

2.1 LINHAS DE COMANDOS PRINCIPAIS

Bite Size Command Line.pdf 22 MB

A folha de dicas dos comandos do terminal Mac

AÇÃO	COMANDO
FUNDAMENTOS	
ATALHOS	
/ (Barra para frente)	Diretório de nível superior
Aba	Auto-completar nomes de arquivos e pastas
. (Período Único)	Diretório atual
Ctrl + A	Vai para o início da linha em que você está digitando
.. (período duplo)	Diretório Parental
Ctrl + E	Vá para o final da linha em que você está digitando
~ (Til)	Diretório inicial
Ctrl + U	Limpe a linha antes do cursor
sudo [comando]	Execute o comando com os privilégios de segurança do super usuário
Ctrl + K	Limpe a linha após o cursor
nano [arquivo]	Abre o editor do Terminal
Ctrl + W	Exclua a palavra antes do cursor
abrir arquivo]	Abre um arquivo
Ctrl + T	Troque os dois últimos caracteres antes do cursor
[comando] -h	Obtenha ajuda sobre um comando
Esc + T	Troque as duas últimas palavras antes do cursor
man [comando]	Mostra o manual de ajuda do comando
Ctrl + L	Limpar a tela
ALTERAR DIRETÓRIO	
Ctrl + C	Mate tudo o que você está executando
CD	Diretório inicial
Ctrl + D	Sai do shell atual

cd [pasta]	Mudar de diretório, por exemplo, Documentos de cd
Opção + →	Mova o cursor uma palavra para a frente
cd ~	Diretório inicial
Opção + ←	Mova o cursor uma palavra para trás
CD/	Raiz da unidade
Ctrl + F	Mova o cursor um caractere para a frente
CD -	Diretório ou pasta anterior em que você navegou pela última vez
Ctrl + B	Move o cursor um caractere para trás
pwd	Mostre seu diretório de trabalho
Ctrl + Y	Cole o que foi cortado pelo último comando
CD..	Mover para o diretório pai
Ctrl + Z	Coloca tudo o que você está executando em um processo suspenso em segundo plano
CD../..	Suba dois níveis
Ctrl + _	Desfaz o último comando
LISTAR CONTEÚDO DO DIRETÓRIO	
GERENCIAMENTO DE ARQUIVOS E DIRETÓRIOS	
ls	Mostra o nome dos arquivos e subdiretórios do diretório
mkdir <dir>	Crie uma nova pasta chamada <dir>
ls -C	Forçar saída de várias colunas da listagem
mkdir -p <dir> / <dir>	Crie pastas aninhadas
ls -a	Liste todas as entradas, incluindo aquelas com. (Ponto final) e .. (ponto final duplo)
mkdir <dir1> <dir2> <dir3>	Crie várias pastas de uma vez
ls -1	Produz a lista de arquivos em uma entrada por formato de linha
mkdir "<dir>"	Crie uma pasta com um espaço no nome do arquivo
ls -F	Exibe uma / (barra) imediatamente após cada caminho que é um diretório, * (asterisco) após programas executáveis ou scripts e @ após um link simbólico
rmdir <dir>	Excluir uma pasta (funciona apenas em pastas vazias)
ls -S	Classifique arquivos ou entradas por tamanho
rm -R <dir>	Exclua uma pasta e seu conteúdo

ls -l	Liste em um formato longo. Inclui modo de arquivo, nome do proprietário e do grupo, data e hora em que o arquivo foi modificado, nome do caminho e muito mais
touch <file>	Crie um novo arquivo sem qualquer extensão
ls -lt	Liste os arquivos classificados por tempo de modificação (mais recentes primeiro)
cp <file> <dir>	Copie um arquivo para a pasta
ls -lh	Longa listagem com tamanhos de arquivo legíveis por humanos em KB, MB ou GB
cp <file> <newfile>	Copie um arquivo para a pasta atual
ls -lo	Liste os nomes dos arquivos com tamanho, proprietário e sinalizadores
cp <file> ~ / <dir> / <newfile>	Copie um arquivo para a pasta e renomeie o arquivo copiado
ls -la	Lista o conteúdo detalhado do diretório, incluindo arquivos ocultos
cp -R <dir> <"novo dir">	Copie uma pasta para uma nova pasta com espaços no nome do arquivo
TAMANHO DO ARQUIVO E ESPAÇO EM DISCO	
cp -i <file> <dir>	Avisa antes de copiar um arquivo com uma mensagem de aviso de substituição
du	Liste o uso de cada subdiretório e seu conteúdo
cp <file1> <file2> <file3> / Users / <dir>	Copie vários arquivos para uma pasta
du -sh [pasta]	Saída legível por humanos de todos os arquivos em um diretório
rm <file>	Excluir um arquivo (isso exclui o arquivo permanentemente; use com cuidado.)
du -s	Mostra uma entrada para cada arquivo especificado
rm -i <file>	Exclua um arquivo apenas quando você der a confirmação
du -sk * | sort -nr	Liste arquivos e pastas, totalizando o tamanho, incluindo as subpastas. Substitua sk * por sm * para listar os diretórios em MB
rm -f <file>	Forçar remoção sem confirmação
df -h	Calcule o espaço livre em disco do seu sistema
rm <file> <file> <file3>	Exclua vários arquivos sem qualquer confirmação
df -H	Calcule o espaço livre em disco em potências de 1.000 (em oposição a 1.024)
mv <file> <newfilename>	Mover / renomear
HISTÓRICO DE COMANDOS	
mv <file> <dir>	Mova um arquivo para a pasta, possivelmente substituindo um arquivo

	existente
Ctrl + R	Pesquise os comandos usados anteriormente
mv -i <file> <dir>	Opcional -i sinalizador para avisá-lo antes de sobrescrever o arquivo
history n	Mostra os comandos anteriores que você digitou. Adicione um número para limitar os últimos n itens
mv * .png ~ / <dir>	Mova todos os arquivos PNG da pasta atual para uma pasta diferente
![value]	Execute o último comando digitado que começa com um valor
HOMEBREW	
!!	Execute o último comando digitado
brew doctor	Verifique a mistura em busca de problemas em potencial
PERMISSÕES	
brew install <formula>	Instale uma fórmula
ls -ld	Exibir a permissão padrão para um diretório home
brew uninstall <formula>	Desinstalar uma fórmula
ls -ld / <dir>	Exibir a permissão de leitura, gravação e acesso de uma pasta específica
brew list	Liste todas as fórmulas instaladas
chmod 755 <file>	Altere a permissão de um arquivo para 755
brew search	Exibir fórmulas disponíveis para brew
chmod -R 600 <dir>	Altere a permissão de uma pasta (e seu conteúdo) para 600
brew upgrade	Atualize todas as brew desatualizadas e não fixadas
chown <user>: <group> <file>	Altere a propriedade de um arquivo para usuário e grupo. Adicione -R para incluir o conteúdo da pasta
brew update	Obtenha a versão mais recente do homebrew e da fórmula
PROCESSOS	
brew cleanup	Remova a versão mais antiga da fórmula instalada
ps -ax	Saída de processos atualmente em execução. Aqui, a mostra os processos de todos os usuários ex mostra os processos que não estão conectados ao Terminal
brew tap homebrew/cask	Toque no repositório do cask do GitHub

ps -aux	Mostra todos os processos com% cpu,% mem, entrada de página, PID e comando
brew cask list	Liste todos os cask instalados
top	Exibir informações ao vivo sobre os processos em execução
brew cask install <cask>	Instale o cask dado
top -ocpu -s 5	Exibir processos classificados por uso de CPU, atualizando a cada 5 segundos
desinstalação do brew cask <cask>	Desinstale o cask fornecido
top -o rsize	Classifique os principais por uso de memória
PROCURAR	
kill PID	Saia do processo com ID <PID>. Você verá o PID como uma coluna no Monitor de Atividade
find <dir> -name <"file">	Encontre todos os arquivos com o nome <arquivo> dentro de <dir>. Use curingas (*) para pesquisar partes de nomes de arquivos
ps -ax | grep <appname>	Encontre um processo por nome ou PID
grep "<text>" <file>	Exibir todas as ocorrências de <texto> dentro de <arquivo> (adicionar -i para não diferenciar maiúsculas de minúsculas)
REDE	
grep -rl "<texto>" <dir>	Pesquise todos os arquivos contendo <texto> dentro de <dir>
ping <host>	Fazer ping do host e exibir o status
SAÍDA	
whois <domain>	Informações de saída de whois para um domínio
cat <file>	Produza o conteúdo de <file>
curl -O <url / to / file>	Baixe o arquivo via HTTP, HTTPS ou FTP
less <file>	Produza o conteúdo de <file> usando o comando less que suporta paginação e mais
ssh <username> @ <host>	Estabeleça conexão SSH para <host> com o usuário <username>
head <file>	Produza as primeiras 10 linhas de <file>
scp <file> <user> @ <host>: / remote / path	Copie <file> para um <host> remoto
<cmd> > > <arquivo>	Anexa a saída de <cmd> a <file>
<cmd>> <arquivo>	Direcione a saída de <cmd> para <file>
<cmd1> | <cmd2>	Direcione a saída de <cmd1> para <cmd2>

PRINCIPAIS LINHAS DE COMANDOS

tab

AUTO COMPLETAR

com as pastas que tem

tab + /	tab/

PWD

SABER ONDE ESTOU

pwd

LS

MOSTRAR ARQUIVOS DE ONDE ESTOU

qual pasta?

ls

↓

mostra onde estou

RM

REMOVER PASTA (vazia)

rm

↓

remover arquivo

REMOVER TUDO QUE ESTÁ NO ARQUIVO

rm -r

↓

remover TODO o arquivo

RM

CRIAR UM ATALHO COM LINHAS DE COMANDO

cria um atalho
Symbolic Link Symlinks

ln - s urers/brunna/outros

↓ ↓

p/ onde vai *pasta que vai*

exemplo 2

ln - s frango teste

↓ ↓

p/ onde vai *pasta que vai*

exemplo 3

ln - s caminho de destino + caminho do atalho

↓

↓

p/ onde vai

atalho a partir de onde vai sair

DE

ONDE VAI SAIR

quando não especifica

(base) brunna@MacBook-Pro-de-Brunna **Desktop** %

ln - s user/brunna/code/outros/frango portal_frangoso

↓

quando não especifica para onde vai, ele vai para a pasta ATUAL

→ **Desktop** ←

>

DENTRO DE: >

>

↓

dentro de...

ECHO

CRIANDO > echo < IMPRIMINDO

echo "é outro arquivo" **>** frango/outro.text

↓ ↓ ↓ ↓

criar arquivo 'dentro de' direção do arquivo

pronúncia: "eco"

CP

COPIAR

cp

cp + arquivo origem + arquivo destino

CD

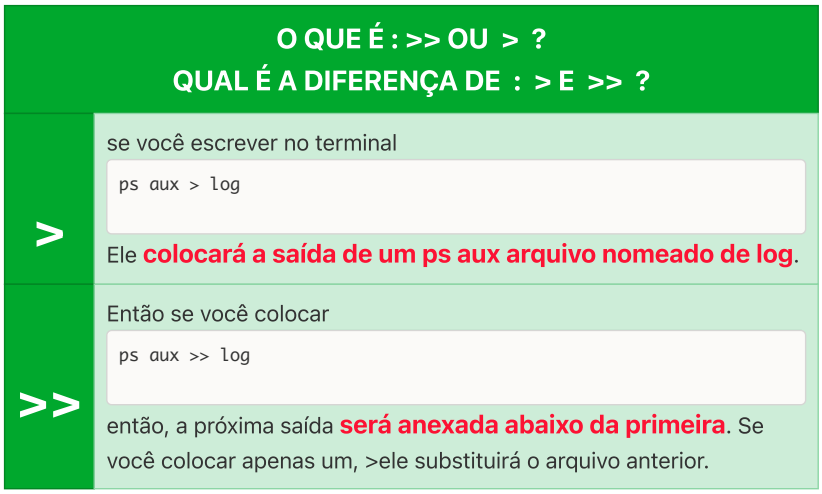
ENTRAR NO ARQUIVO/PASTA

cd

VOLTAR

cd ..

2.2 - > ou >>


O QUE É : >> OU > ?
QUAL É A DIFERENÇA DE : > E >> ?
>
se você escrever no terminal
ps aux > log
Ele colocará a saída de um ps aux arquivo nomeado de log.
>>
Então se você colocar
ps aux >> log
então, a próxima saída será anexada abaixo da primeira. Se você colocar apenas um, >ele substituirá o arquivo anterior.

CONTACT

Brunna Croches

Developer Full Stack

brunnacroches.dev

linkedin.com/brunnacroches

github.com/brunnacroches

@brunnacroches.dev

discord.com/brunnacroches

brunnacroches@gmail.com

let's share