bash basics…

<tab>	complete command
<up-arrow>	recalls the last command
;	ends command (optional)
#	comment
x &	runs command x in the background
x && y	runs x then y if x is successful
x | y	pipes data from x to y
./x	executes file x (in current directory)
`x` or $(x)	execute x in another shell
>x or >>x	overwrite or append data to file x

…bash basics…

cd	**c**hange **d**irectory
ls	**l**i**s**t directory contents
pwd	**p**rint **w**orking **d**irectory to stdout
apropos x	finds commands **apropos** of x
man x	displays the **man**ual for x
find x	lists all files/directories in x
less <x>	displays x (or stdin)
mkdir x	**m**a**k**es **dir**ectory x

…bash basics…

file x	displays type of **file** x
gzip x	compresses file x (creates x.gz)
gzip -d x.gz	uncompresses archive x.gz (deletes x.gz)
gzip -dc x.gz	uncompresses archive x.gz to stdout
xz x	compresses file x (creates x.xz)
xz -d x.gz	uncompresses archive x.xz (deletes x.xz)
xz -dc x.gz	uncompresses archive x.xz to stdout

…bash basics

tar xvzf x.tgz	uncompresses archive x.tgz
tar czvf x.tgz y	creates archive x.tgz from y
tar xvJf x.txz	uncompresses archive x.txz
tar cJvf x.txz y	creates archive x.txz from y
$PATH	the directories bash searches for files in
$HOME	the $USER’s default directory
sudo x	run command x with a user p**s**e**udo**nym

file permissions

each file has an owner

- only the owner (or their superior) can modify a file
- use chown to **ch**ange the **own**er (usually must be root)

each file has read, write, and execute permissions

- set for the owner, the group, and other users
- use chmod to **ch**ange the **mod**e
  - chmod [who] +/- what file
  - e.g. chmod u+x file; chmod ugo-rwx file

file permissions…

```
dpl10@phyto:~$ ls -lh
total 8.0K
-rw-rw-r-- 1 dpl10 dpl10  4 Sep  7 16:59 y.txt
-rwxrwxr-x 1 dpl10 dpl10 26 Sep  7 17:00 z.sh
```

…file permissions

chmod x y => **ch**ange **mod**e of file y to x

use octal notation for best results

0700 == -rwx------

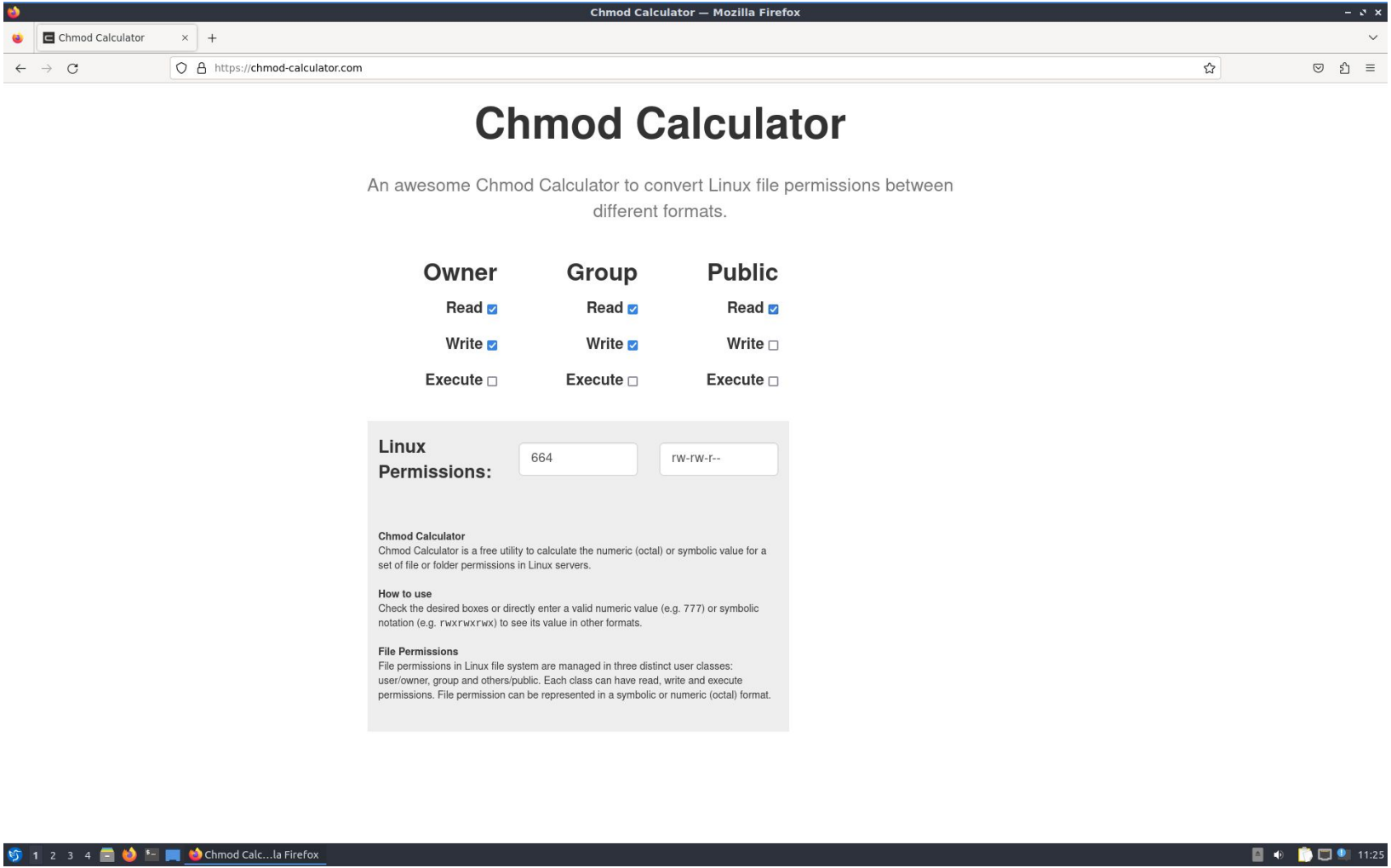
0666 == -rw-rw-rw-

0755 == -rwxrw-rw-

0600 == -rw-------

0644 == -rw-r--r--

chown x:y z => **ch**ange **own**er of file z to x user, y group


Chmod Calculator — Mozilla Firefox
Chmod Calculator
https://chmod-calculator.com
Chmod Calculator
An awesome Chmod Calculator to convert Linux file permissions between different formats.
Owner
Group
Public
Read
Write
Execute
Linux Permissions:
664
rw-rw-r--
Chmod Calculator
Chmod Calculator is a free utility to calculate the numeric (octal) or symbolic value for a set of file or folder permissions in Linux servers.
How to use
Check the desired boxes or directly enter a valid numeric value (e.g. 777) or symbolic notation (e.g. rwxrwxrwx) to see its value in other formats.
File Permissions
File permissions in Linux file system are managed in three distinct user classes: user/owner, group and others/public. Each class can have read, write and execute permissions. File permission can be represented in a symbolic or numeric (octal) format.
1 2 3 4
Chmod Calc...la Firefox
11:25

installing software

necessary to conduct most bioinformatic analyses

commercial software typically have (very bad) LINUX installers

open source software typically have (bad) LINUX installers

- often (very) poorly documented
- tested on a limited number of configurations

use the tested configuration if yours does not work

- dependencies are not always listed or ambiguously listed
- virtualization can be very useful for testing

one of the most frustrating things about POSIX

package managers

installs executable (usually binary) and configuration files

- greatly simplifies installation and upgrades

depends upon the (usually volunteer) package maintainers

apt

- the Debian wrapper for dpkg
- used to install, update, remove, and purge packages
- will install dependencies for the target package
- http://packages.ubuntu.com/
- if apt fails, try aptitude (the industrial strength version)

Ubuntu Packages Search

This site provides you with information about all the packages available in the Ubuntu Package archive.

Browse through the lists of packages:

- bionic (18.04LTS)
- bionic-updates
- bionic-backports
- focal (20.04LTS)
- focal-updates
- focal-backports
- jammy (22.04LTS)
- jammy-updates
- jammy-backports
- kinetic (22.10)
- kinetic-updates
- kinetic-backports
- lunar

There is also a list of packages recently added to lunar.

Old releases can be found at http://old-releases.ubuntu.com/.

Search

Search package directories

Keyword: Search Reset

Search on: Package names only Descriptions Source package names

Only show exact matches:

Distribution: kinetic Section: any

There are shortcuts for some searches available:

Search in specific suite: [bionic] [bionic-updates] [bionic-backports] [focal] [focal-updates] [focal-backports] [jammy] [jammy-updates] [jammy-backports] [kinetic] [kinetic-updates] [kinetic-backports] [lunar]
Search in all suites
Limit search to a specific architecture: [i386] [amd64] [powerpc] [arm64] [armhf] [ppc64el] [s390x]

Some results have not been displayed due to the search parameters.

You have searched for packages that names contain *xxhash* in suite(s) *jammy*, all sections, and all architectures. Found **8** matching packages.

Exact hits

Package xxhash

- jammy (22.04LTS) (utils): Extremely fast hash algorithm [**universe**]
  0.8.1-1: amd64 arm64 armhf i386 ppc64el s390x

Other hits

Package golang-github-cespare-xxhash-dev

- jammy (22.04LTS) (devel): implementation of the 64-bit xxHash algorithm (XXH64) [**universe**]
  2.1.1-2: all

Package golang-github-oneofone-xxhash-dev

- jammy (22.04LTS) (devel): native implementation of the excellent XXHash hashing algorithm [**universe**]
  1.2.4-1.1: all

Package golang-github-pierrec-xxhash-dev

- jammy (22.04LTS) (devel): pure Go implementation of xxHash (32 and 64 bits versions) [**universe**]
  0.1.1-4: all

Package librust-xxhash-rust-dev

- jammy (22.04LTS) (rust): Xxhash - Rust source code [**universe**]

Package: xxhash (0.8.1-1) [universe]

Extremely fast hash algorithm

Other Packages Related to xxhash

● depends ◆ recommends ■ suggests • enhances

- ● libc6 (>= 2.34)
  GNU C Library: Shared libraries
- ● libxxhash0 (>= 0.8.1-1)
  shared library for xxhash

Download xxhash

Architecture	Package Size	Installed Size	Files
amd64	41.1 kB	125.0 kB	[list of files]
arm64	32.7 kB	85.0 kB	[list of files]
armhf	38.0 kB	84.0 kB	[list of files]
i386	56.1 kB	176.0 kB	[list of files]
ppc64el	64.0 kB	153.0 kB	[list of files]
s390x	33.0 kB	93.0 kB	[list of files]

Links for xxhash

Ubuntu Resources:

- Bug Reports
- Ubuntu Changelog
- Copyright File

Download Source Package xxhash:

- [xxhash_0.8.1-1.dsc]
- [xxhash_0.8.1.orig.tar.gz]
- [xxhash_0.8.1-1.debian.tar.xz]

Maintainer:

- Ubuntu Core Developers (Mail Archive)

Please consider filing a bug or asking a question via Launchpad before contacting the maintainer directly.

Original Maintainer (usually from Debian):

File list of package *xxhash* in *jammy* of architecture *amd64*

```
/usr/bin/xxh128sum
/usr/bin/xxh32sum
/usr/bin/xxh64sum
/usr/bin/xxhsum
/usr/share/doc/xxhash/changelog.Debian.gz
/usr/share/doc/xxhash/copyright
/usr/share/man/man1/xxh128sum.1.gz
/usr/share/man/man1/xxh32sum.1.gz
/usr/share/man/man1/xxh64sum.1.gz
/usr/share/man/man1/xxhsum.1.gz
```

apt

sudo apt update	updates the package cache
sudo apt upgrade	installs (most) upgrades
sudo apt autoremove	removes unneeded packages
apt-cache search x	searches repository for x
apt-cache show x	information for package x
sudo apt install x	installs package x
sudo apt remove x	removes package x
sudo apt purge x	removes package x and its files

encapsulators and containers

install finicky things easily

allow multiple versions of the same program

(sometimes) safer

(usually) at the expense of storage/memory/speed

encapsulators (e.g. snap)

usually for GUI programs, distribution specific

containers (e.g. flatpak, docker)

usually for CLI programs, relatively generic

scripts (interpreted code)

simple scripts are placed in a directory listed in $PATH

- e.g. $HOME/bin

dependencies must be satisfied

- interpreter
- interpreter extensions/packages/modules
- external programs
- environmental variables

some interpreters (e.g. Perl, Python, JavaScript) have their own package management systems

compiling…

convert source code (text) into a (binary) executable file

./configure

- script that produces an appropriate MakeFile
- not required for all programs
- should indicate missing dependencies
- can set various options
  - specify the (non–standard) location of a dependency
  - set program specific options
    - e.g. compile a parallel version, set install location

…compiling…

make

run the MakeFile script (runs the compiler and linker)

-jx uses x processor cores (for a faster run)

CFLAGS="-march=native -O2 -static"

optimizes the code for your processor

makes static code

can improve performance

the binary is not ‘portable’

…compiling…

make check

tests the newly compiled software

not all MakeFiles have tests

<sudo> make install

installs the newly compiled software

not all MakeFiles have an install function

make clean

erases all of the newly compiled software

not all MakeFiles have a clean function

...compiling

code (sometimes) compiles differently with other compilers

- pass vs. fail
- memory leaks
- execution speed (options + compiler ability)
- different floating point values

gcc, clang (llvm), icc

- test to find the best combination of settings/compilers

moving files (locally)

cp x y	copies file x to y
mv x y	moves file x to y (renames x)
mv x y/	moves file x to y (moves x to directory y)
mv x y/z	moves file x to y/z (moves x to y as z)
rm x	removes file x
rm -R x	removes directory x

moving files (anywhere)

wget x	downloads from url x
sftp	transfers files to/from a remote server
scp	transfers files to/from a remote server
ssh	opens a terminal on a remote server
rsync	synchronizes two directories

reading files

cat x	copy file x to stdout
tac x	copy file x to stdout (backwards)
less x	read file x one screen at a time
head -n y x	copy the first y lines of file x to stdout
tail -n y x	copy the last y lines of file x to stdout

POSIX data streams

standard input (stdin; 0)

- from the keyboard (default)
- ignored by some programs (e.g. ls)

standard output (stdout; 1)

- to the terminal/screen (default)
- not produced by all programs

standard error (stderr; 2)

- to the screen (default)
- not produced by all programs

redirecting data streams...

use < to redirect stdin from a file

e.g. wc < file.txt

use > to redirect stdout* to a file (overwrite mode)

e.g. ls > file.txt [overwrites an existing file!]

use >> to redirect stdout* to a file (append mode)

e.g. ls >> file.txt

*stdout and stderr may be combined in some circumstances (e.g. when using nohup)

...redirecting data streams...

use 1> or 1>> to redirect stdout to a file

== ‘>’ or ‘>>’

use 2> or 2>> to redirect stderr to a file

use 2>&1 to redirect stderr to stdout

use &> or &>> to redirect both stderr and stdout to one file

> /dev/null to redirect to oblivion

…redirecting data streams…

x < y	stdout from y to x
x > y	stdout from x to y (overwrites y)
x 1> y	stdout from x to y (overwrites y)
x >> y	stdout from x to y (appends y)
x 1>> y	stdout from x to y (appends y)
x 2> y	stderr from x to y (overwrites y)
x 2>> y	stderr from x to y (appends y)
x &> y	stdout + stderr from x to y (overwrites y)
x &>> y	stdout + stderr from x to y (appends y)

…redirecting data streams

x 1> y 2> z	stdout from x to y, stderr from x to z
x 1>> y 2> z	stdout from x to y, stderr from x to z
x 1> y 2>> z	stdout from x to y, stderr from x to z
x 1>> y 2>> z	stdout from x to y, stderr from x to z
x 1> y 2>&1	stdout from x to y, stderr becomes stdout
x 1>> y 2>&1	stdout from x to y, stderr becomes stdout

pipes

a redirect with conversion

stdout from the process on the left passes through a pipe and becomes stdin for the process on the right

e.g. ls | wc -l == ls > file.txt; wc -l file.txt

can string many pipes together

e.g. cat file.txt | tr 'x' 'y' | sort | uniq -c

(never) pipe a sort [it can be slow]

can split streams using tee

LINUX text processing

stdin/stdout redirection (pipes) + small utilities = quantification and dissection of text files

Boolean logic & simple math answer most questions

build ‘queries’ one step at a time

    easy to check for programming errors

very fast for most datasets (i.e. small ones)

the only thing efficient enough for large datasets

can often be run in parallel via xargs

can work with compressed data

text processing utilities…

agrep	**a**pproximate **grep** (tre-agrep)
awk	a pattern scanning programming language
bc	a **b**asic **c**alculator language
bloom	a **Bloom** filter
cat	con**cat**enate files
datamash	an advanced calculator and table manipulation program
diff	find **diff**erences between two files line–by–line
grep	**g**lobally search a **r**egular **e**xpression and **p**rint
head	output the first part of a file
join	**join** lines of two files using a common field
perl	**p**ractical **e**xtraction and **r**eporting **l**anguage

…text processing utilities

paste	a column oriented concatenation text utility
python	a (text processing) programming language
sed	**s**tream **ed**itor for filtering and transforming files
shuf	**shuf**fle lines in a file (do not use)
sort	**sort** lines of files
split	**split** a file into pieces
tail	output the last part of a file
tr	**tr**ansliterate (or delete) characters
uniq	**uniq**ue (or not) lines
wc	**w**ord (and other things) **c**ount