tutorialworks

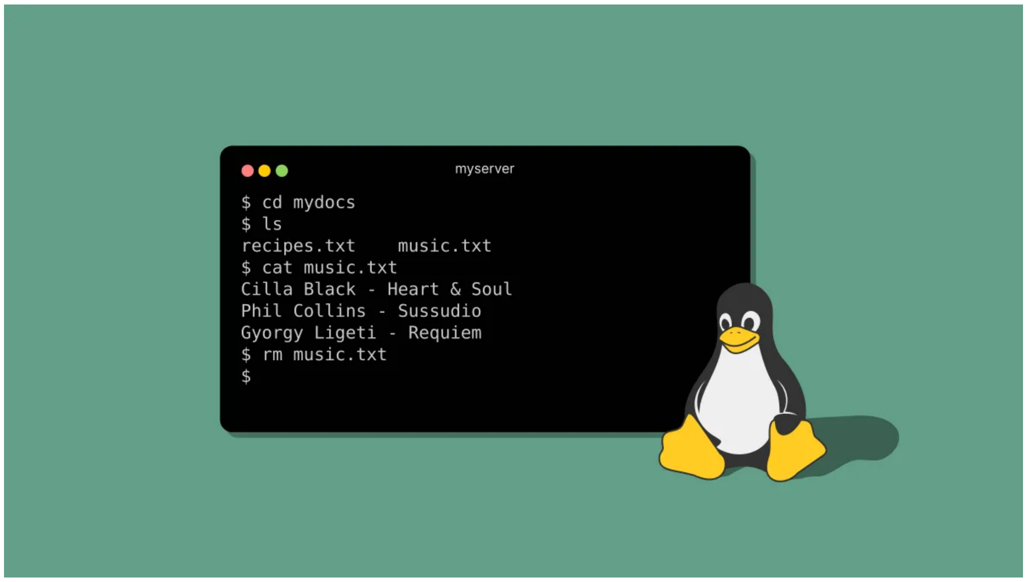
LINUX

45 Essential Linux Commands (with Examples)

From 'cd' to 'vim', here are some of the most useful commands and programs in Linux.

Written by Tom Donohue
Updated: 21 April 2022
0 Comments

Overwhelmed by all the Linux commands you think you need to know? Perplexed by `pwd`? Well, you can rest easy now, you've found the right page. Here are 45 of the most useful Linux commands, all in one place.



```
$ cd mydocs
$ ls
recipes.txt    music.txt
$ cat music.txt
Cilla Black - Heart & Soul
Phil Collins - Sussudio
Gyorgy Ligeti - Requiem
$ rm music.txt
$
```

Let's do the penguin

When you're learning Linux, it can seem like you need to memorise a bunch of stuff. How could you possibly remember every single command in Linux?

Well, let me tell you a little secret: most Linux users don't remember every single command. (We're human, after all!)

But, there are some commands which you'll probably use more than others. And that's inspired me to create this big list!

Before we dive into the list, don't forget to check out our other Linux articles:

MORE LINUX GUIDES TO CHECK OUT...

- **Linux for DevOps**: What you need to know
- **'top' command explained**: How to monitor system usage

All the commands

OK, so let's get started:

The table below contains a big list of all the Linux commands that we've covered in this guide.

For most commands we also give the reason behind the rather cryptic name, to help you remember it!

Click on a command name to jump straight to its entry, and see one or two examples, or just scroll down to read the whole lot.

COMMAND	DESCRIPTION	WHY THE NAME?
Help! (and WTF)		
man	Show help (the manual)	manual
which	Find out where a command is	which location?
--help	Find help on any command	help!
Ctrl+C	Stop the current program	
:q!	Quit the *vi* text editor without saving	quit!
Logging in and out		
logout	Exit the current login shell	
passwd	Change your password	password
sudo	Execute a command as a superuser	super-user do

COMMAND	DESCRIPTION	WHY THE NAME?
ssh	Create an SSH connection (a terminal) to a server	secure shell
📁 Move around the file system		
cd	Change the current directory (folder)	change directory
ls	List files in a directory	list
mkdir	Make/create a new directory	make directory
pwd	Print current directory	print working directory
cp	Copy files and directories	copy
rm	Delete files and directories	remove
📝 Find, view and edit files		
cat	Print the contents of a file	concatenate
chmod	Change the permissions of a file or directory	change mode
chown	Change the owner and group of a file or directory	change owner
diff	Show the difference between two files	difference
file	Show the type of a file	
less	Browse the contents of a file	opposite of 'more'
locate	Find files with names matching a pattern	
tail	Print the last few lines of a file	tail end (of something)
touch	Create a new file or update an existing one	
nano	An interactive file editor	Nano's ANOther Editor
vim	An interactive file editor (Intermediate level)	visual improved
📦 Install programs		
dnf	Install and manage packages in RHEL/Fedora	dandified yum (?!!)
apt	Install and manage packages in Ubuntu	Advanced Package Tool
📯 Networking		
curl	Get a web page or API	client URL

COMMAND	DESCRIPTION	WHY THE NAME?
dig	Look up DNS information	domain information gather
ip addr	Show your network configuration	IP address
ping	Check to see if a server is alive	

⏱️ Manage processes

COMMAND	DESCRIPTION	WHY THE NAME?
kill	Terminate a program (process)	
top	Show the top processes	table of processes
ps	List running processes	processes
Ctrl+Z	Suspend a process	
fg	Resume a process	foreground
jobs	Lists all active jobs (processes)	

⚙️ System management

COMMAND	DESCRIPTION	WHY THE NAME?
cat /etc/redhat-release	Show Linux distribution name & version	
date	Show the current date and time	
df	Show free disk space	disk free
du	Show disk space usage	disk used
shutdown	Halt or reboot your machine	
uname	Print system information	unix name
uptime	Show how long the system has been running	

> 📢 If this page seems like something that your Linux-loving friends would appreciate, then why not give it a share? It lets the algorithm know that it's not too bad. 😀

Learning Linux? Here are 45 top Linux commands that will power up your terminal sessions.

Click to Tweet Share on LinkedIn

Now we'll dive into each command, explain what it does and show an example or two.

Advertisements

🙋‍♀️ Help! (and WTF)

If you're going to memorise any commands, memorise these!

Why?

Because these first commands help you to find the documentation, look up help for a command, or find a command on your system. The Linux documentation is always available, even when you don't have an internet connection.

So, get familiar with these commands, and then you'll have help on hand whenever you need it.

- **man**: Show help (the manual)
- **which**: Find out where a command is
- **<command> --help**: Find help on any command
- **Ctrl+C**: Stop the current program
- **:q!**: Quit the *vi* text editor without saving

Advertisements

01. `man` — Show help (the manual)

Linux has a manual which you can read offline (yes, I know, retro!)

man lets you read the manual (sometimes called **manpages** or the **system reference manuals**).

Each command in Linux has its own page (or screen) in **man**.

When you want to call up the documentation for a particular command, just type `man` then the command name.

```
# Show the manual page for the `ls` command.
$ man ls

# Search for 'search'
$ man -k search
```

02. `which` — Find out where a command is

If a command exists on your system, `which` tell you!

`which` tells you where a command is located on your disk. That’s called the **path** to the command.

Alternatively, if you want to look for files, try locate.

```
# Shows the location of the command `ls` (all Linux systems have this!)
$ which ls
```

03. `<command> --help` — Find help on any command

Shows quick help on any command. It will show you the arguments, parameters that a command takes, so you can use the command in the correct way.

(Most commands recognise `--help`, but you might find a few commands which don't, or prefer `-h` instead.)

```
# Show quick help for the command `ls`
$ ls --help
```

04. `Ctrl+C` — Stop the current program

This key combination will **terminate** the current command and return you to the command prompt.

It actually sends a shutdown **signal** to the command, so you can sometimes expect the command to do a little tidying-up first before it exits.

05. `:q!` — Quit the *vi* text editor without saving

`vi` (the text editor) is one of the commands where `Ctrl+C` to exit doesn't work. So if you're planning to use vi (and you should, it's a life-enriching experience!), remember this key combination! This sequence of keys will let you quit the editor, without saving your changes.

Honestly I still have this command on a sticky note next to my laptop. :)

🧑‍💻 Logging in and out

When you're using the Linux terminal and you want to know how to end your session, or make a new one, you'll find these commands good to know!

- **logout**: Exit the current login shell
- **passwd**: Change your password
- **sudo**: Execute a command as a superuser
- **ssh**: Create an SSH connection (a terminal) to a server

06. `logout` — Exit the current login shell

Quit your current terminal session and return back to where you came from. If you're connected to a remote server, then it `logout` should return you back to your local terminal.

You can also usually use the keyboard shortcut `Ctrl+D` to logout.

07. `passwd` — Change your password

It's important to change your password once in a while! Always choose a secure password, or a passphrase, which is several words joined together.

(If you want help choosing a secure password, try this free tool from Bitwarden .)

```
# Change your password (you'll be prompted for your current password first)
$ passwd
```

08. `sudo` — Execute a command as a superuser

When you want to do something that only superusers can do, use this command. For example, editing a system file or running a powerful command.

This can sometimes be a bit difficult to understand for beginners, so if you want to dive deeper into this then check out this blog from Red Hat .

```
# Edit the file `/etc/motd` (which only superusers have write access to)
$ sudo vi /etc/motd
```

09. `ssh` — Create an SSH connection (a terminal) to a server

When you connect to another server, you usually use **SSH** to do it.

This command creates a new session and then prompts you to log in.

```
# Opens an SSH connection to _example.com_ and tries to log in as user _dave_.
$ ssh dave@example.com
```

🗂️ Move around the file system

When you want to open files and directories (folders), you'll usually use these commands.

I got lost in the filesystem once. Never again with these commands!

- **cd**: Change the current directory (folder)

- **ls**: List files in a directory
- **mkdir**: Make/create a new directory
- **pwd**: Print current directory
- **cp**: Copy files and directories
- **rm**: Delete files and directories

10. `cd` — Change the current directory (folder)

To move between different folders, use `cd`.

```
# Change directory to `mydocs`
$ cd mydocs

# Change to the parent directory (the directory above the current one)
$ cd ..

# Change to your user home directory (note it's just `cd` without any arguments!)
$ cd
```

11. `ls` — List files in a directory

List files and directories. You can add different options to the command, to change the output or sort it.

This is the closest equivalent to the `dir` command on Windows.

```
# List files in the current directory (to show your current directory, type [pwd](#p
$ ls

# List files (including hidden files) and also show their permissions and who owns t
$ ls -al

# List files (including hidden), ordering by most recent at the bottom.
$ ls -alrt
```

12. `mkdir` — Make/create a new directory

Directories work just like folders. Use this command to create a new one.

```
# Make a directory called `songs`
$ mkdir songs
```

13. `pwd` — Print current directory

This shows the your **working directory** (also called the current directory).

14. `cp` — Copy files and directories

This command lets you create a copy of a file or a directory and its contents. It's mega-useful for creating backups of files when you're editing something important and you want to keep a copy!

```
# Copy the contents of the file `money.txt` into a file `money.bak`
$ cp money.txt money.bak

# Copy the directory `songs` (and all of its contents) into `songs2`
$ cp -rp songs songs2
```

15. `rm` — Delete files and directories

There's no recycle bin in Linux so when you use the `rm` command, your files will be deleted permanently! To delete all files in a folder, you can add the `-r` ("recursive") option.

```
# Delete the file `myfile.txt`
$ rm myfile.txt

# Delete the directory `songs` and all files in it, without prompting! (warning, use
$ rm -rf songs
```

📝 Find, view and edit files

Now we're getting to the really useful commands! These are the commands that you'll use to manipulate files and create new ones. Most things in Linux are configured using files, so it's good to know these commands.

When you want to edit a text file, you'll use a *text editor*. The three most popular text editors are **vi**, **nano** and **emacs**.

Personally, I started out with nano and then moved onto vi!

- **cat**: Print the contents of a file
- **chmod**: Change the permissions of a file or directory
- **chown**: Change the owner and group of a file or directory
- **diff**: Show the difference between two files
- **file**: Show the type of a file
- **less**: Browse the contents of a file

- **locate**: Find files with names matching a pattern
- **tail**: Print the last few lines of a file
- **touch**: Create a new file or update an existing one
- **nano**: An interactive file editor
- **vim**: An interactive file editor (Intermediate level)

16. `cat` — Print the contents of a file

When you want to see the contents of a file, this is the simplest command to use!

The only problem is it just dumps the entire file to your terminal. So if it's a big file and you want to navigate around it, use the less command instead.

```
# Print the contents of `songs.txt`
$ cat songs.txt
```

17. `chmod` — Change the permissions of a file or directory

Each file in Linux has permission settings for three different groups: **user** (owner) permission, **group** permission, and **other** permission.

This command helps you set permissions for each of those three groups.

```
# Grant all permissions on `myscript.sh` to the user, and execute-only to _group_ an
$ chmod 755 myscript.sh
```

```
# Allow the user to execute `myscript.sh`, and perform no other changes.
$ chmod u+x myscript.sh
```

18. `chown` — Change the owner and group of a file or directory

As well as permissions, a file in Linux can have an owner and be assigned to a group.

This command lets you change the owner or group of a file.

```
# Change the owner to _dave_ and the group to _staff_ on the file _accounts.txt_.
$ chown dave:staff accounts.txt
```

19. `diff` — Show the difference between two files

Sometimes you want to see which changes were made to a file, or find out if two files are exactly the same. This command helps with that!

```
# Print the differences between `nyc.txt` and `london.txt`
$ diff nyc.txt london.txt
```

20. `file` — Show the type of a file

Attempts to detect the type of a file (e.g. image, mp3, Java program). It can also detect some binary files too.

This is useful if you want to know what a file is used for, or if the file has been saved without an **extension** (e.g. like .mp3)

It can usually detect the correct file type, even if the filename has the wrong extension.

```
# Show the file type of `cilla.mp3`
$ file cilla.mp3
```

21. `less` — Browse the contents of a file

Shows the contents of a file as a set of pages in your terminal. You can use the arrow keys or PageUp/PageDown to scroll through the file.

Very handy for big files!

```
# Show and scroll the file `songs.txt`. Press `q` to exit.
$ less songs.txt
```

22. `locate` — Find files with names matching a pattern

Searches a database of files on disk, to find a file matching a given pattern.

The database is only updated about once per day, so this is more useful for finding system files.

This is different from which, which looks for commands.

```
# Find the location of any file called `httpd.conf`
$ locate httpd.conf
```

23. `tail` — Print the last few lines of a file

Very handy if you're looking at log files, and you want to see the most recent lines in the log.

You can also add `-f` to "follow" the file, which means that any new lines will be printed to the screen too.

```
# Print end of file and follow new lines in file `server.log`
$ tail -f server.log
```

24. `touch` — Create a new file or update an existing one

This command is often used to do one of two things: either create a new, empty file, or update the "modified date" on an existing file.

Personally I find it the most useful for creating empty files.

```
# Create an empty file called `newfile.txt`
$ touch newfile.txt
```

25. `nano` — An interactive file editor

This is the easiest file editor for beginners, because you can type anywhere, and use the arrow keys to move around.

Most of the commands in nano start with `^`, which is the `Ctrl` key. To save a file, see the example below!

```
# Edit the file `customers.txt` in _nano_
$ nano customers.txt

# Save a file and then exit.
$ Ctrl+O Ctrl+X
```

26. `vim` — An interactive file editor (Intermediate level)

vim means “vi improved”. It’s an improvement on the original “vi” editor.

Everything in *vim* is controlled with keyboard commands, which can take some time to learn. But once you’ve learned it, you can be very productive!

If you want an easier editor, try nano.

```
# Edit the file `customers.txt` in _vim_
$ vim customers.txt
```

📦 Install programs

How do you install software in Linux?

In Linux, you usually install software from a **repository**. A repository is basically a central place which stores programs and is usually run by the Linux distribution (e.g. Ubuntu or Red Hat).

Programs that are installed like this, are usually called **packages**. Repositories contain **thousands** of packages! You can usually install them with a single command. You don't need to mess around with downloading a file from a web site, extracting it and so on.

You might still need to install a program manually if it's not in the repositories, but for those you should probably follow the instructions included with the program.

- **dnf**: Install and manage packages in RHEL/Fedora
- **apt**: Install and manage packages in Ubuntu

27. `dnf` — Install and manage packages in RHEL/Fedora

This is the new way to install software in Red Hat Enterprise Linux (RHEL) and its related distros, like Fedora and CentOS.

Previously, in older versions of RHEL/Fedora linux, the package manager was called yum .

🔐 Note you'll usually need to run this command with sudo.

```
# Search the online <i>repositories</i> for a package named `curl`
$ dnf search curl
```

```
# Install a package called `curl` from the online repositories
$ dnf install curl

# List all packages installed on your system
$ dnf list installed

# Upgrade all packages on your system to the latest compatible versions
$ dnf upgrade
```

28. `apt` — Install and manage packages in Ubuntu

In Ubuntu, you use this command to install and manage packages.

Note you'll usually need to run this command with sudo.

```
# Install a package called `curl` from the online repositories
$ sudo apt install curl

# Uninstall the package called `curl` from your system
$ sudo apt remove curl
```

Networking

Networking is one of the hardest things to understand, because there are so many layers and protocols! For most of us, it takes some time to understand it all.

But these tools make things a little easier, because they let you create network requests, look up information and call APIs.

These commands are very useful for testing, or figuring out why something isn't working....

- **curl**: Get a web page or API
- **dig**: Look up DNS information
- **ip addr**: Show your network configuration
- **ping**: Check to see if a server is alive

29. `curl` — Get a web page or API

This is an extremely powerful tool for making HTTP (web) requests. You can use it to hit APIs, fetch a web site, download files from websites and more.

Another solid alternative to this command is **wget**, but I prefer curl.

```
# Fetch the Google home page and print the contents
$ curl www.google.com

# Fetch the Google home page and save the HTML to a file
$ curl -o index.html www.google.com
```

30. `dig` — Look up DNS information

DNS is how you translate domain names into IP addresses, and much more.

Using dig can be really useful if you want to find out things like the IP address of a web site, how email is processed (with MX records), and lots more.

```
# Show the IP addresses for domain _reddit.com_
$ dig +short A reddit.com
```

31. `ip addr` — Show your network configuration

Prints your IP addresses. You'll usually have multiple IP addresses on your Linux server, depending on how it's configured.

```
# Prints a list of all network interfaces, their IP addresses and config
$ ip addr
```

32. `ping` — Check to see if a server is alive

A "ping" is something specific in networking terminology. It means sending a small packet of data to a server and waiting for it to respond.

You'll sometimes use this if you want to see if a server is alive and responding to requests.

```
# Send a ping to _www.tutorialworks.com_ to see if it is responding
$ ping www.tutorialworks.com
```

Advertisements

⌚ Manage processes

Programs running in Linux are usually called **processes**.

Sometimes you’ll get a program which has stopped responding, or you need to troubleshoot it.

These commands help with that. You can list running processes, stop them, and more.

- **kill**: Terminate a program (process)
- **top**: Show the top processes
- **ps**: List running processes
- **Ctrl+Z**: Suspend a process
- **fg**: Resume a process
- **jobs**: Lists all active jobs (processes)

33. `kill` — Terminate a program (process)

When you want to end a process in Linux, you send it a **signal**. There are different types of signals, the common ones being “TERM” and “KILL”.

“TERM” means terminate, is like a polite request for the command to stop. “KILL” is a bit more forceful - you can use this when a program really needs to terminate **now**.

```
# Terminate process number 12345
$ kill 12345

# Forcefully terminate (kill!) process number 12345
$ kill -9 12345
```

34. `top` — Show the top processes

A bit like Task Manager in Windows, this command shows you all of the programs that are running, their process IDs and how much resources they're using, such as memory and CPU.

We've also got a full article about the top command if you want to find out more!

35. `ps` — List running processes

Prints a list of the current processes, with their IDs, and the command that was used to start each process.

A good way of finding out exactly which programs you're currently running!

This command produces a list of processes, so you can filter the list using another tool like grep.

```
# List all your processes
$ ps

# List all processes matching the string `java`
$ ps -ef | grep java
```

36. `Ctrl+Z` — Suspend a process

This isn't a command, but it's a good keyboard shortcut to know.

This will pause a program, if it's running in the foreground, so you can run something else, or perform another task.

To resume the process again, use the command fg.

37. `fg` — Resume a process

Un-pauses or resumes a process which you've paused using Ctrl+Z.

38. `jobs` — Lists all active jobs (processes)

Lists all of the jobs. This is primarily used when you've paused/suspended a few processes and you want to see a list of them.

System management

When you want to find out more about your system, you can use these commands.

Advertisements

- **cat /etc/redhat-release**: Show Linux distribution name & version
- **date**: Show the current date and time

- **df**: Show free disk space
- **du**: Show disk space usage
- **shutdown**: Halt or reboot your machine
- **uname**: Print system information
- **uptime**: Show how long the system has been running

39. `cat /etc/redhat-release` — Show Linux distribution name & version

Show the current version of your Linux distribution.

You'll notice that this is actually a file! This file is created when you install Linux. Here we're just using cat to show the file's contents.

```
# Show the Linux distribution name and version
$ cat /etc/redhat-release
```

40. `date` — Show the current date and time

Your server has a date and time set. It also has a time zone too. This will be used whenever the server writes files.

So it's good to confirm that your server is set with the correct time and timezone.

41. `df` — Show free disk space

Want to find out how much disk space you have left? This command does it.

You can change the output in a few different ways to make it more human-readable (see the example below).

```
# Show free disk space with human-readable sizes
$ df -h
```

42. `du` — Show disk space usage

This is the opposite of the df command. It shows how much space your files and folders are taking up.

It can be handy to figure out where all your space is disappearing to!

```
# Show disk usage (in human-readable sizes) of all top-level directories
$ du -h -d 1 /
```

43. `shutdown` — Halt or reboot your machine

Getting ready to shut down your server? Use this command.

Be careful that nobody else is using the server at the same time. :-)

🔐 Note you'll usually need to run this command with sudo.

```
# Reboot the system now
$ shutdown -r now
```

44. `uname` — Print system information

Shows information about the Linux kernel.

```
# Show all info about the Linux kernel, including version number.
$ uname -a
```

45. `uptime` — Show how long the system has been running

Use this to check whether your system has been running without problems, or whether it’s been restarted at some point recently.

Phew!

And that’s your lot for today. We’ve covered a wide range of Linux commands, from the everyday commands, to keyboard combinations, and more.

(And, of course, that all-important key-combination you need to quit *vim*.)

Feel free to come back here any time you need a friendly refresher on Linux commands.

Want to suggest any Linux commands for this list? Pop your thoughts in the comments section below!

Happy Linuxing!

Acknowledgements: The fantastic Emoji in the social thumbnail of this post was designed by https://www.iconfinder.com/design-oni

By **Tom Donohue, Editor** | Twitter | LinkedIn

Tom is the founder of *Tutorial Works*. He's an engineer and open source advocate. He uses the blog as a vehicle for sharing tutorials, writing about technology and talking about himself in the third person. His very first computer was an Acorn Electron.

Join the discussion

Got some thoughts on what you've just read? Want to know what other people think? Or is there anything technically wrong with the article? (We'd love to know so that we can correct it!) **Join the conversation and leave a comment.**

Comments are moderated.

No Comments Yet

Type Comment Here (at least 3 chars)

Name | E-mail (optional) | Website (optional) | Preview | Submit

Want more? Read these articles next...

You've found the end of another article! Keep on learning by checking out one of these articles:

- **Run a web server in a Linux VM with Vagrant [Learning Project]**: Learn Linux and virtualisation basics by deploying a website in this tutorial.
- **The Best Places to Learn & Try Kubernetes Online**: Learning Kubernetes can seem challenging. But fear not! Here's a boatload of resources that will help you get there.
- **7 power features of the Linux shell you really need to know**: Here are some power features that you can use to get things done quicker and easier
- **DevOps Project Ideas**: A career in DevOps is all about building a broad skill base and understanding. Use these project ideas to invest in yourself and get that...
- **Linux for DevOps: What You Need to Know**: Worried about touching a Linux terminal? Don't know where to start? Find out what you need to know about Linux for DevOps.

Advertisements

Tutorial Works is a website to help you navigate the world of IT, and grow your tech career, with tips, tutorials, guides, and *real opinions*.

Thanks for being here today! 👋

About this website

Privacy policy

Contact us

TOPICS

Linux

Containers

DevOps

Kubernetes

Java

FREE DOWNLOADS

DevOps Roadmap