Linux Mint 5 Day Essentials Introduction Course

Introduction

Why an introductory Linux admin, command line-oriented eBook in 2023 when GUIs are so powerful now and Windows is still the dominant desktop OS?

Well, lots of reasons, but for anyone looking for a career in IT, having some Linux skills is an obvious bonus and as it is a free OS that has programmes written for it that cover almost any use anyone would want. It is a cheap way into the IT industry for students on a budget if nothing else, as it runs on almost all hardware, but you can research what laptops or hardware - like wifi cards - it doesn't have built-in drivers for, on the web.

Linux has never been so popular, as it's a very stable, truly multitasking OS, and I hope to show you some of its incredibly low overhead versatility, running as it is on my 15-year-old, dual-core laptop, writing this text – from basic Desktop user admin, to software installation, file system security concepts, basic programming environments for Python, C and Java, and very importantly, backups. All in 5 "days", more or less... depending on you.

"96.3% of the top one million web servers are running Linux.

Among the websites that run on Linux, remarkable ones include Twitter, Yahoo, and eBay, speaking volumes about Linux's popularity. The internet is increasingly dependent on Linux — 96.3% of the top one million web servers use Linux, server statistics indicate. Windows (1.9%) and FreeBSD (1.8%) share the rest.

(ZDNet)"
https://truelist.co/blog/Linux-statistics/#:~:text=The%20internet%20is%20increasingly%20dependent,1.8%25)%20share%20the%20rest.

This eBook is designed to give an introductory to intermediate level insight into some immediately useful tasks that a Linux-based system offers, along with some technical insight to the topics under discussion, as relevant, to give an idea of some aspects of System Administration.

It is presumed the reader has a laptop or desktop available with an Ubuntu-based Linux distribution installed and has "root" (Administrator) level access, as the installation of a particular OS is not covered in this text. The examples shown are using a Linux Mint, Ubuntu-based installation.

You can download installation files for pen drives from Linuxmint.com

Depending on the reader's prior knowledge and ability, the material may need more than five days to cover and digest, copying the command examples into your own system Terminal.

The concept behind the eBook is to enable an interactive experience for the reader via copying and pasting the commands between this eBook and the Linux Terminal. This is best achieved via some

of the many Linux-based eBook readers, as the Kindle reader format text copy function is a bit awkward, but worse, only available for Windows and Apple systems, not Linux.
The Kindle web reader does not allow text copy due to Amazon's proprietary book formatting and to prevent book copyright infringement, so no use.

The material can be viewed as a basic introduction to System Administration for any Linux platform as well as Windows and Apple systems, from a principles viewpoint, even if the Terminal/Shell commands may be very different between systems.

Operating Systems have to run from a file system of some sort, which requires at least a basic understanding of file system permissions, however implemented, whether the file system is Windows NTFS, Linux Ext4, Apple File System (APFS) or one of many others.

These permissions are absolutely key to understanding how security and access to the file system is implemented for all users, so two days – days 3 and 4 – are dedicated to explaining key aspects of file permissions, with examples, to understand their impact on both local and network access to the filesystem.

The book format comprises GUI / Terminal examples and basic explanations of:

Day 1 - Software Installation, moving around folders, listing file attributes, "root" and sudo users.
Day 2 - Users/Groups, piping hashed password files, pattern searching, AWK column stripping.
Day 3 - File Permissions
Day 4 - File Permissions Continued and Network Access via Samba
Day 5 – Non-Linux Repository Software Package Installation, Programming
Scripts for Python/Java/C examples, Rsync Backup example.

In most cases I will show methods using the Graphical User Interface (GUI) then the equivalent Terminal command line (often much quicker and has system resource overhead benefits) to aid a deeper understanding and appreciation of the genius of the Unix based file system design that Linux is based on.
Commands are in blue. Output is in red. Other links and info are in green.

Contents

Day 1: Software Installs, Moving Around the Filesystem, File Info and Root User p4

Day 2: Admin, Users and Groups p17

Day 3: File and Directory Permissions p32

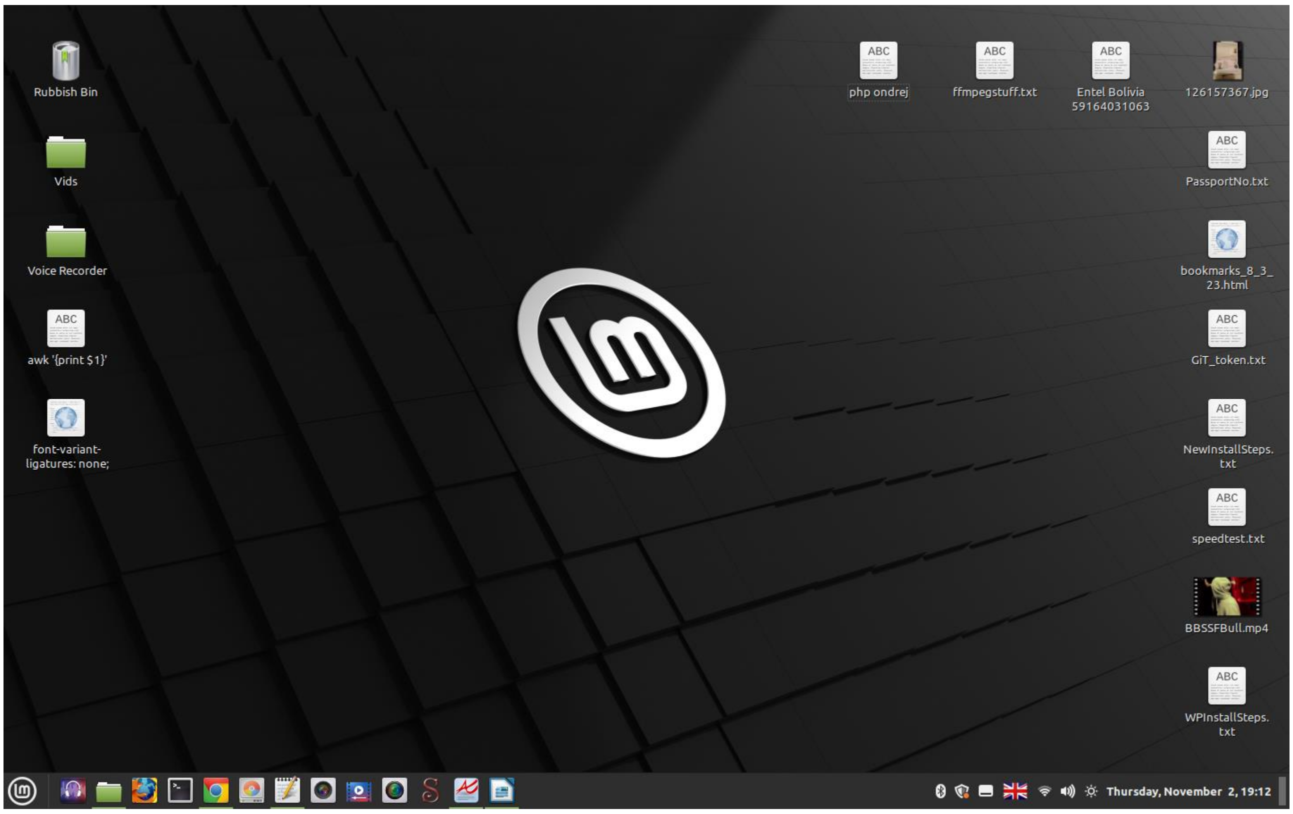
Day 4: File and Directory Permissions, Network Access via Samba p49

Day 5: Package Installation, Programming Environments, Backups p51

Day 1: Software Installs, Moving Around the Filesystem, File Info and Root User

The Desktop

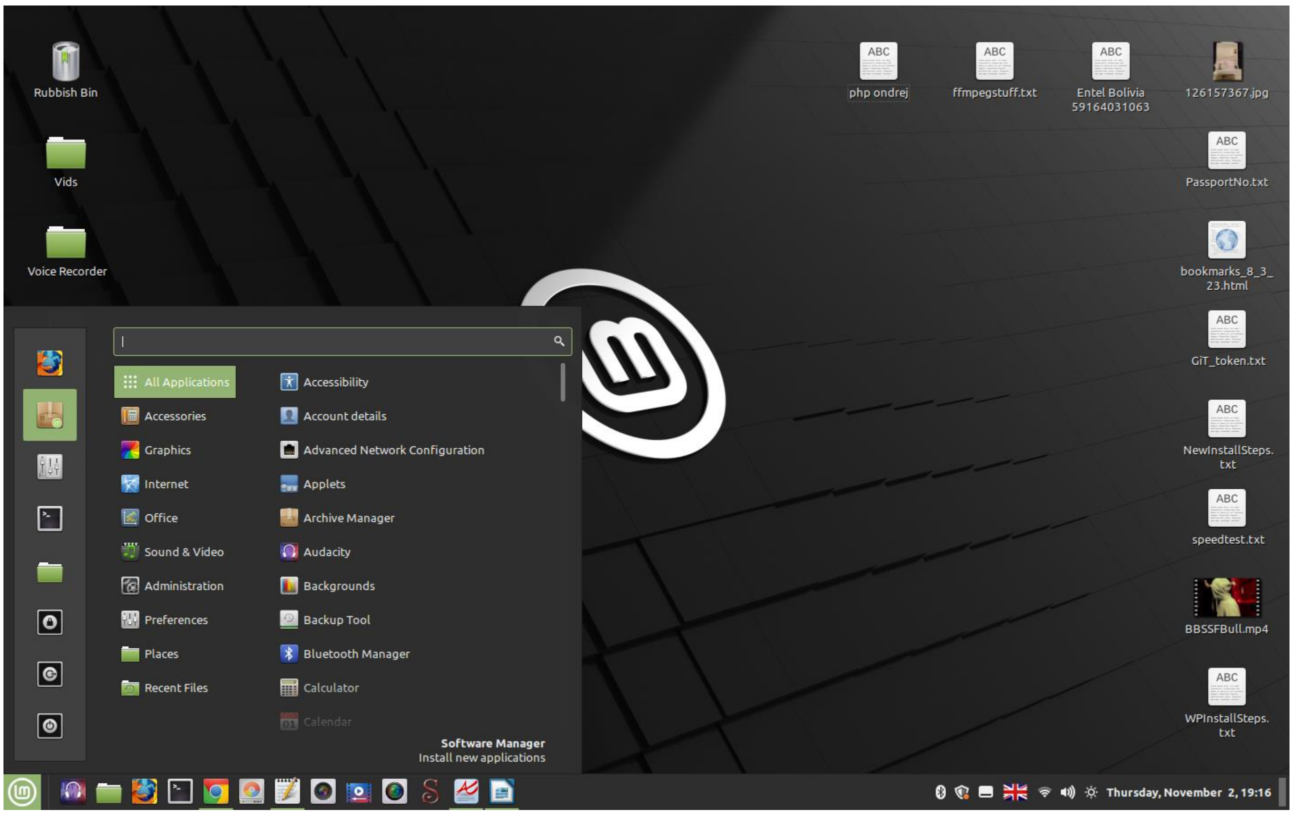
A typical Mint Desktop may look similar to:



Some key commands to learn to help most OS system usage:

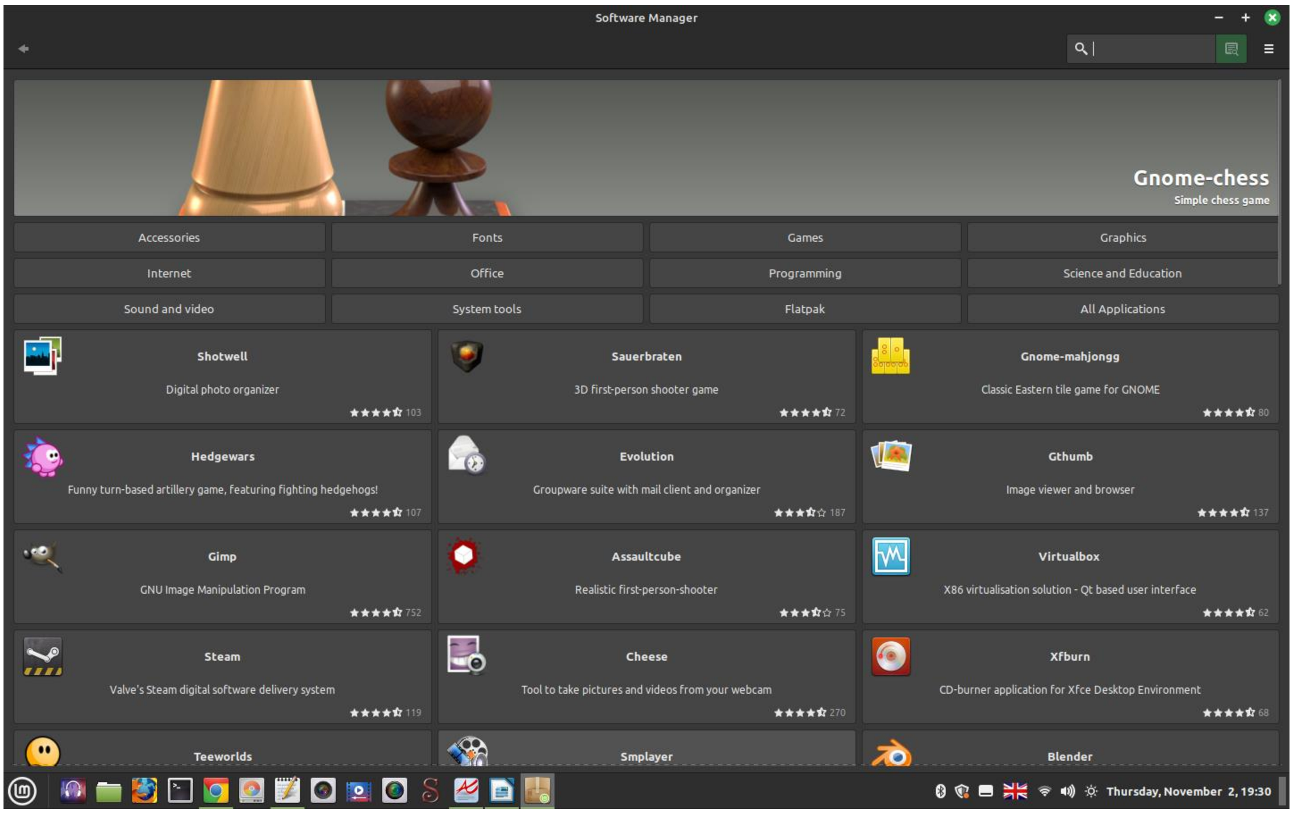
Tab Key - completes available command line options in consoles
F3 - find text in some apps and pages like this webpage!
F5 - browser and Konqueror based apps page refresh (or e.g. http://192.168.1.2/server-status?refresh=5)
Ctrl-C - stops nearly all running command line progs
CTl-Alt-BackSpace - kills and respawns X window on Ctrl-Alt-F8 window
Ctrl-F1 - App help?
CtrlZ - undo
CtrlX - cut
CtrlC - copy text etc (GUI/App/R click)
CtrlV - paste
Ctrl-S - stops fast scroll in Terminal
Ctrl-T for new tab in browser
Ctrl + or - to magnify/shrink text (Ctrl+wheel)
Ctrl A/E (move cursor to start/end of Terminal cmd line)

Ctrl U (deletes Terminal cmd line)
Ctrl Z (pause running Terminal cmd line job)
Ctrl-Alt-F1 to F7 Linux terminals and Dtop GUI (F8 usually)
Esc - close last open box etc.
Alt-F1 - Workspaces
Alt-F2 - GUI run Terminal cmd box
Alt-F3 - ?
Alt F4 - closes active window/app
Alt-F5 - reduces window size to pre max
Print Screen saves screenshot to Pictures in Mint
Alt-PrntScr - saves active window to Pictures in Mint
Ctrl-Alt-Delete - close session option from GUI or reboot from F1-7 terminal



Installing Software

The Menu button, bottom left, where you can search for installed programmes and others to install according to type using the graphical Software Manager:



Shotwell
Digital photo organizer

Sauerbraten
3D first-person shooter game

Gnome-mahjongg
Classic Eastern tile game for GNOME

Hedgewars
Funny turn-based artillery game, featuring fighting hedgehogs!

Evolution
Groupware suite with mail client and organizer

Gthumb
Image viewer and browser

Gimp
GNU Image Manipulation Program

Assaultcube
Realistic first-person-shooter

Virtualbox
X86 virtualisation solution - Qt based user interface

Steam
Valve's Steam digital software delivery system

Cheese
Tool to take pictures and videos from your webcam

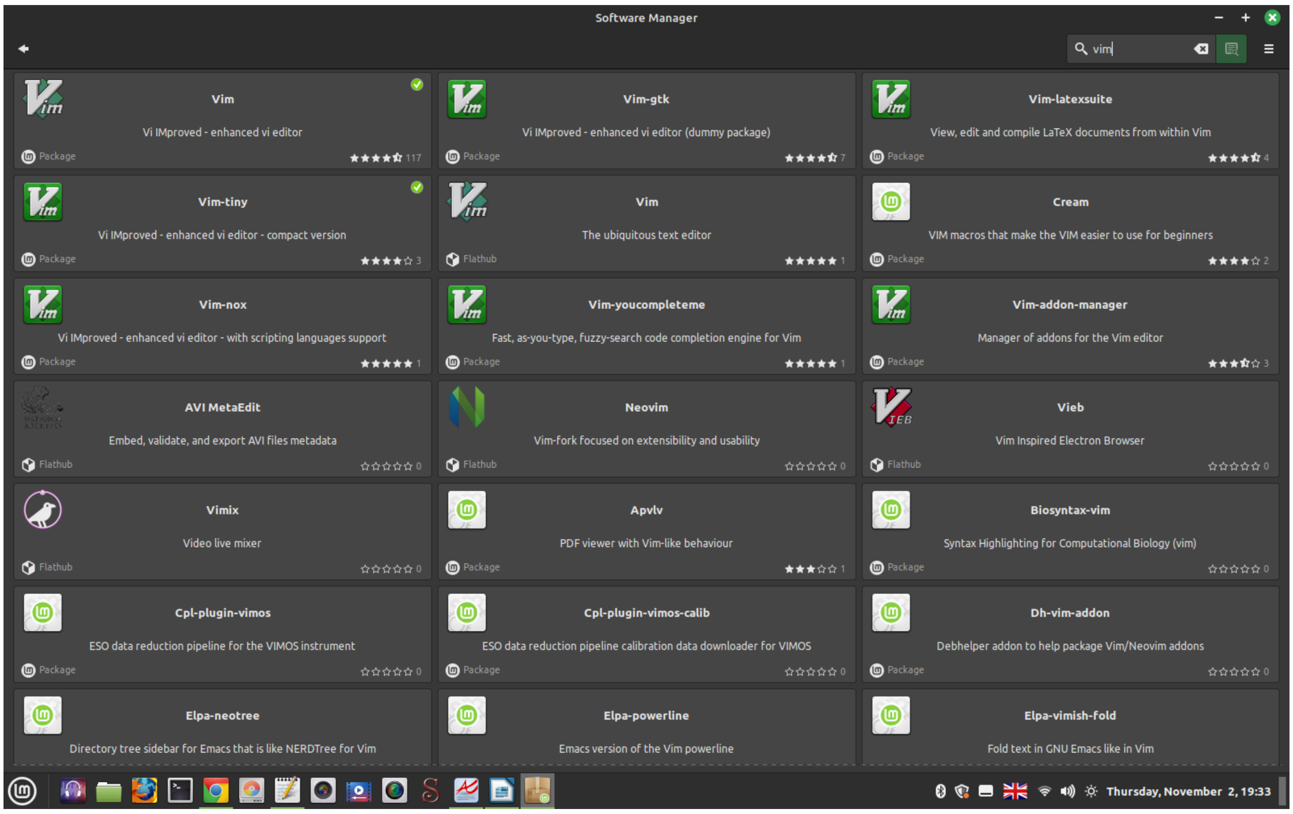
Xfburn
CD-burner application for Xfce Desktop Environment

Teeworlds

Smplayer

Blender

For example, if you searched for a well-known Terminal text editor called Vim, the software will show you any available items that match your search:



Vim
Vi IMproved - enhanced vi editor
Package

Vim-gtk
Vi IMproved - enhanced vi editor (dummy package)
Package

Vim-latexsuite
View, edit and compile LaTeX documents from within Vim
Package

Vim-tiny
Vi IMproved - enhanced vi editor - compact version
Package

Vim
The ubiquitous text editor
Flathub

Cream
VIM macros that make the VIM easier to use for beginners
Package

Vim-nox
Vi IMproved - enhanced vi editor - with scripting languages support
Package

Vim-youcompleteme
Fast, as-you-type, fuzzy-search code completion engine for Vim
Package

Vim-addon-manager
Manager of addons for the Vim editor
Package

AVI MetaEdit
Embed, validate, and export AVI files metadata
Flathub

Neovim
Vim-fork focused on extensibility and usability
Flathub

Vieb
Vim Inspired Electron Browser
Flathub

Vimix
Video live mixer
Flathub

Apvlv
PDF viewer with Vim-like behaviour
Package

Biosyntax-vim
Syntax Highlighting for Computational Biology (vim)
Package

Cpl-plugin-vimos
ESO data reduction pipeline for the VIMOS instrument
Package

Cpl-plugin-vimos-calib
ESO data reduction pipeline calibration data downloader for VIMOS
Package

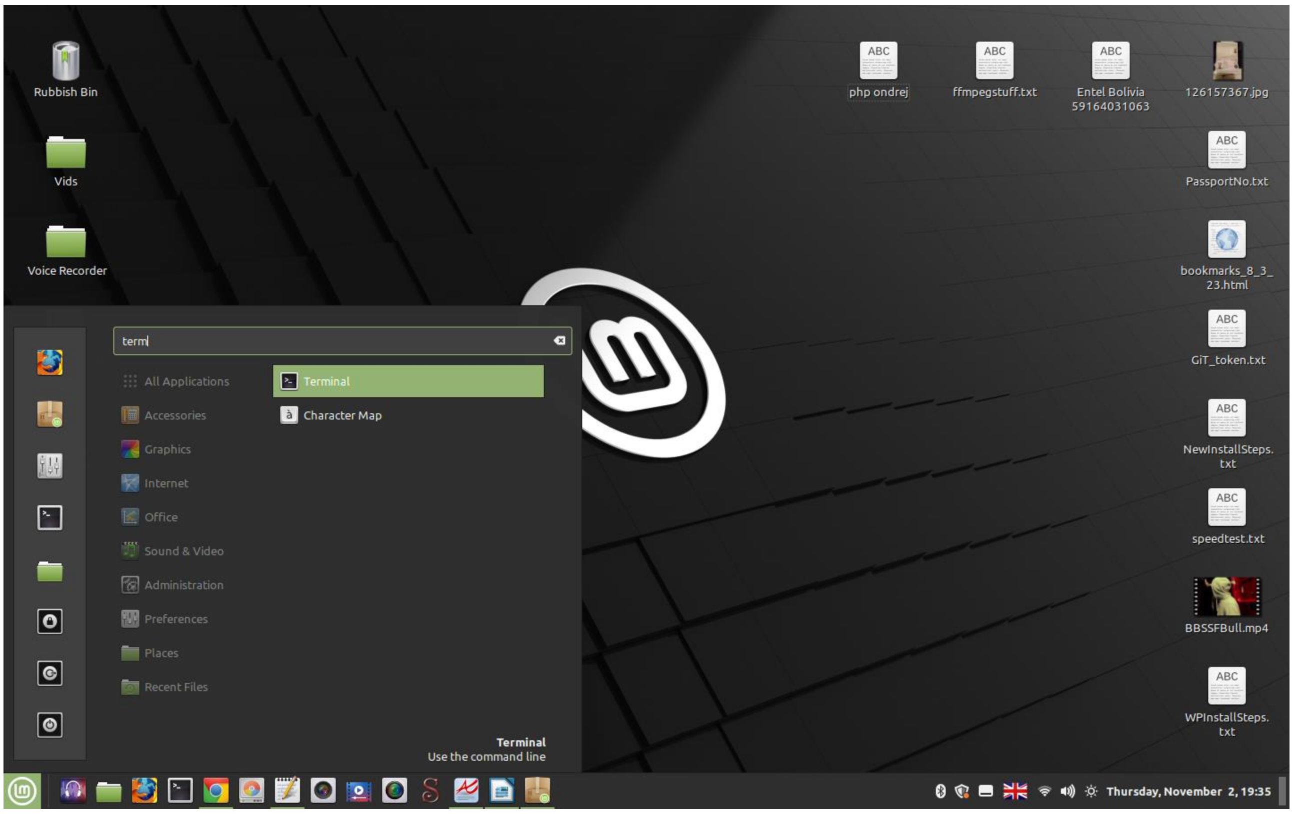
Dh-vim-addon
Debhelper addon to help package Vim/Neovim addons
Package

Elpa-neotree
Directory tree sidebar for Emacs that is like NERDTree for Vim

Elpa-powerline
Emacs version of the Vim powerline

Elpa-vimish-fold
Fold text in GNU Emacs like in Vim

To accomplish the same task in the Terminal command line, go to the Menu button and search Terminal, then click to open it:



When the terminal is open, return to this page and highlight the blue command below, then paste it into the terminal using the middle mouse button or wheel press to paste it into the Terminal.
Or, use slower right click/copy/paste commands:

```
sudo apt install vim
```

Enter your user password to complete the command:

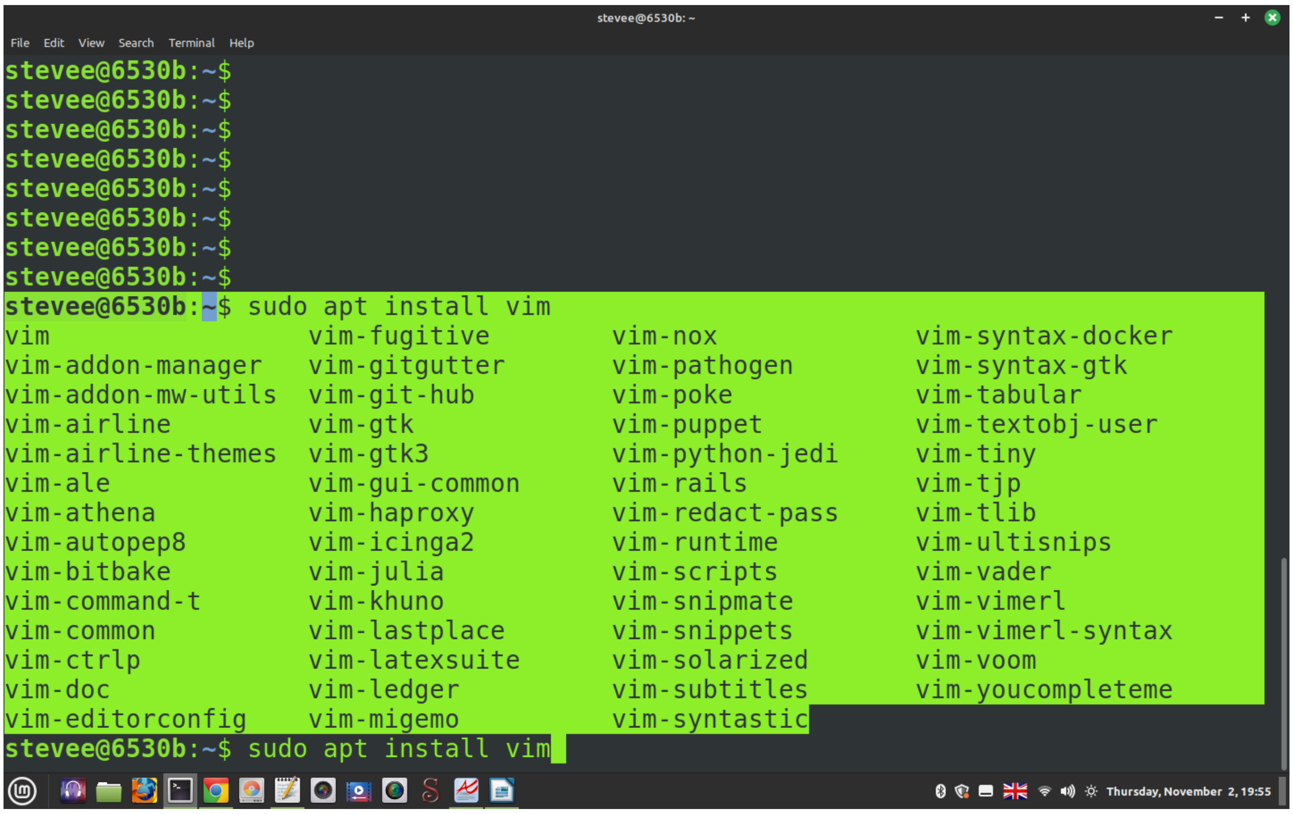
```
[sudo] password for stevee:
```

In my case, Vim is already installed so no software will be downloaded.

Tab Completion

A really useful aspect of the command line is called Tab Completion, often used to see what software is available should you not know the full name or version of a particular software item, or command, you want.

You hit the Tab key after the partial software name, command or directory name, for example; if you Tab when the Terminal cursor is at the end of the last command, vim, it will auto complete all possible options for vim related software that is available in the online Linux repositories:



stevee@6530b: ~
File Edit View Search Terminal Help
Thursday, November 2, 19:55

```
stevee@6530b:~$
stevee@6530b:~$
stevee@6530b:~$
stevee@6530b:~$
stevee@6530b:~$
stevee@6530b:~$
stevee@6530b:~$
stevee@6530b:~$
stevee@6530b:~$ sudo apt install vim
vim                   vim-fugitive          vim-nox               vim-syntax-docker
vim-addon-manager     vim-gitgutter         vim-pathogen          vim-syntax-gtk
vim-addon-mw-utils    vim-git-hub           vim-poke              vim-tabular
vim-airline           vim-gtk               vim-puppet            vim-textobj-user
vim-airline-themes    vim-gtk3              vim-python-jedi       vim-tiny
vim-ale               vim-gui-common        vim-rails             vim-tjp
vim-athena            vim-haproxy           vim-redact-pass       vim-tlib
vim-autopep8          vim-icinga2           vim-runtime           vim-ultisnips
vim-bitbake           vim-julia             vim-scripts           vim-vader
vim-command-t         vim-khuno             vim-snipmate          vim-vimerl
vim-common            vim-lastplace         vim-snippets          vim-vimerl-syntax
vim-ctrlp             vim-latexsuite        vim-solarized         vim-voom
vim-doc               vim-ledger            vim-subtitles         vim-youcompleteme
vim-editorconfig      vim-migemo            vim-syntastic
stevee@6530b:~$ sudo apt install vim
```

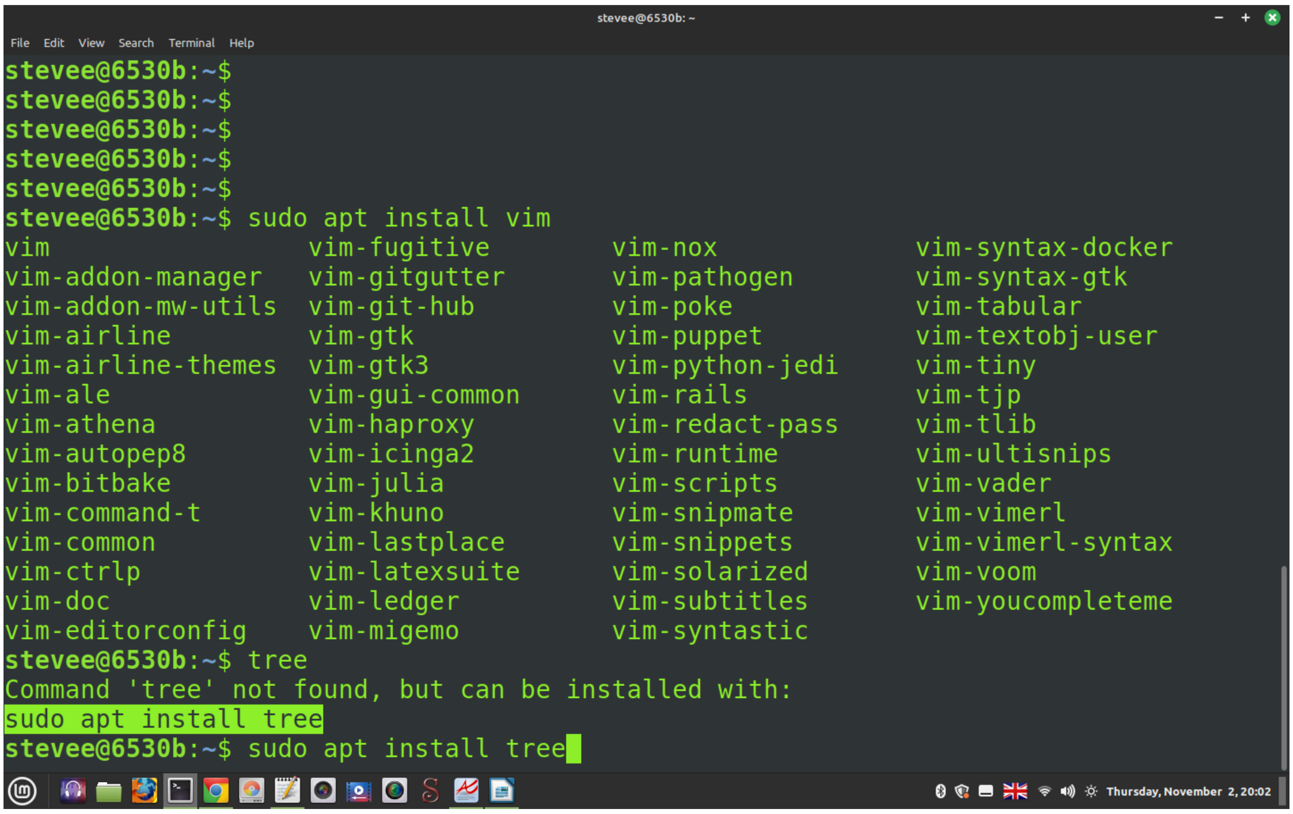
You can change Terminal background and text colours in the Edit/Preferences menu.

File System Orientation

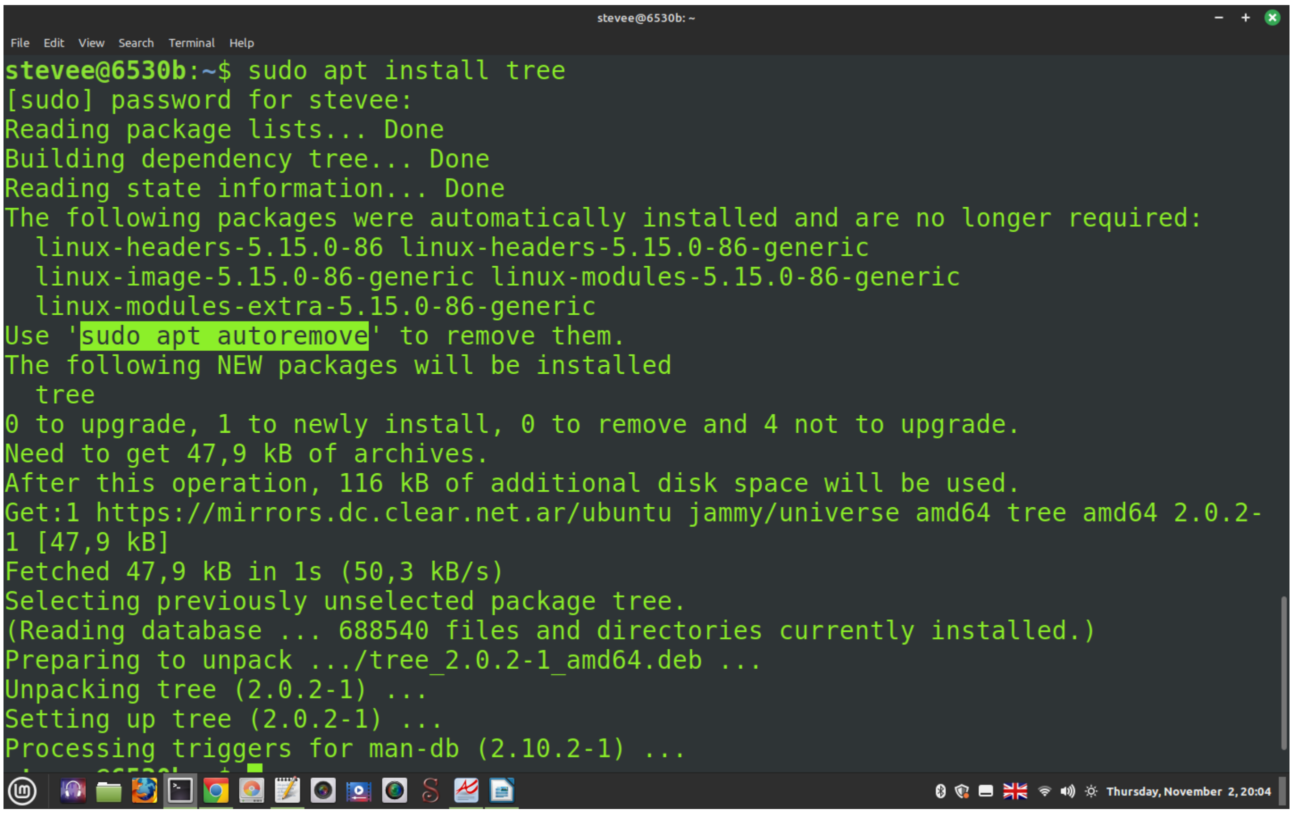
A filesystem is a root or tree structure, with the top of the tree being designated by the forward slash symbol (/) and can be viewed as such within the Terminal using the “tree” command (but first needs to be installed).

This can be accomplished in the terminal by using the Up arrow to bring the last vim install command back (from a command history file in the system: history), then backspacing to replace the word “vim” with “tree”:

```
sudo apt install tree
```


stevee@6530b: ~
File Edit View Search Terminal Help
Thursday, November 2, 20:02

```
stevee@6530b:~$
stevee@6530b:~$
stevee@6530b:~$
stevee@6530b:~$
stevee@6530b:~$
stevee@6530b:~$ sudo apt install vim
vim                 vim-fugitive        vim-nox             vim-syntax-docker
vim-addon-manager   vim-gitgutter       vim-pathogen        vim-syntax-gtk
vim-addon-mw-utils  vim-git-hub         vim-poke            vim-tabular
vim-airline         vim-gtk             vim-puppet          vim-textobj-user
vim-airline-themes  vim-gtk3            vim-python-jedi     vim-tiny
vim-ale             vim-gui-common      vim-rails           vim-tjp
vim-athena          vim-haproxy         vim-redact-pass     vim-tlib
vim-autopep8        vim-icinga2         vim-runtime         vim-ultisnips
vim-bitbake         vim-julia           vim-scripts         vim-vader
vim-command-t       vim-khuno           vim-snipmate        vim-vimerl
vim-common          vim-lastplace       vim-snippets        vim-vimerl-syntax
vim-ctrlp           vim-latexsuite      vim-solarized       vim-voom
vim-doc             vim-ledger          vim-subtitles       vim-youcompleteme
vim-editorconfig    vim-migemo          vim-syntastic
stevee@6530b:~$ tree
Command 'tree' not found, but can be installed with:
sudo apt install tree
stevee@6530b:~$ sudo apt install tree
```

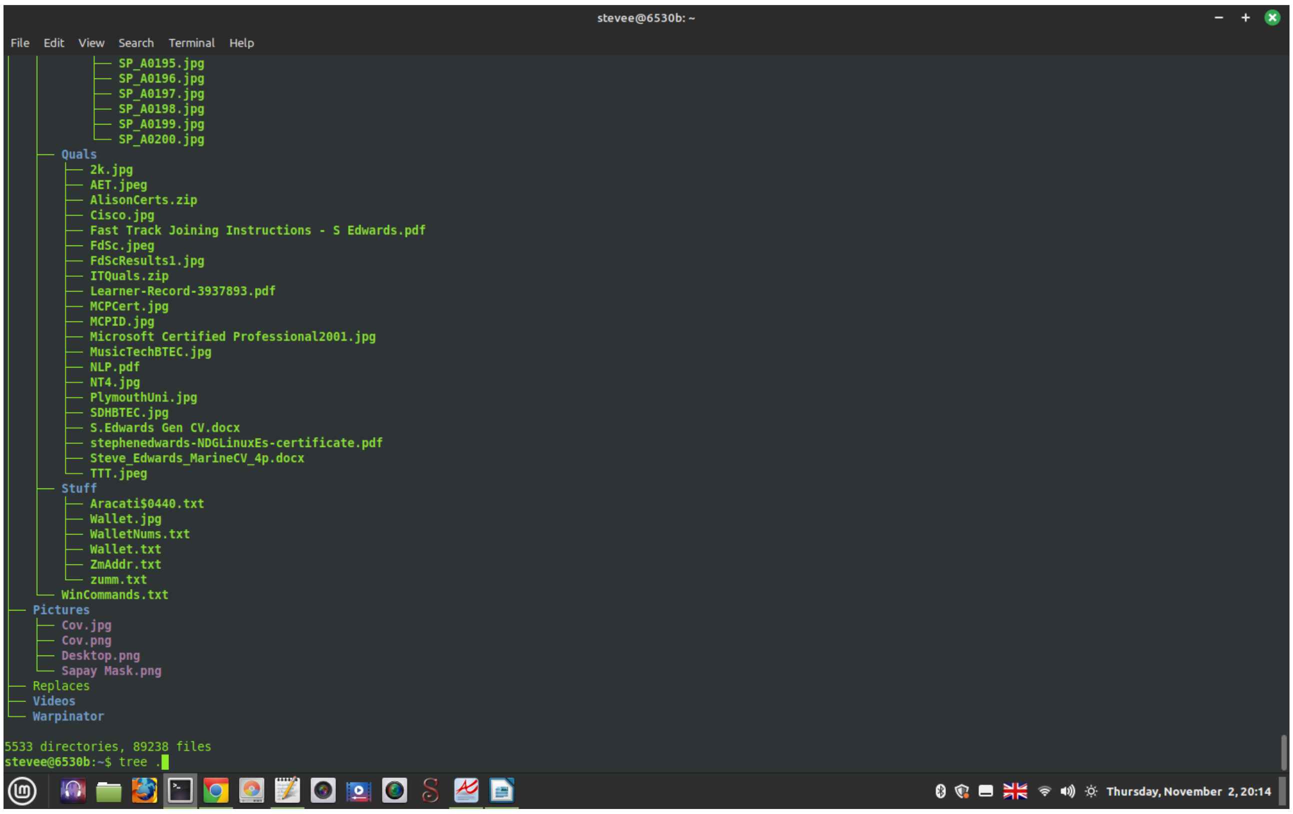

stevee@6530b: ~
File Edit View Search Terminal Help
Thursday, November 2, 20:04

```
stevee@6530b:~$ sudo apt install tree
[sudo] password for stevee:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-headers-5.15.0-86 linux-headers-5.15.0-86-generic
  linux-image-5.15.0-86-generic linux-modules-5.15.0-86-generic
  linux-modules-extra-5.15.0-86-generic
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed
  tree
0 to upgrade, 1 to newly install, 0 to remove and 4 not to upgrade.
Need to get 47,9 kB of archives.
After this operation, 116 kB of additional disk space will be used.
Get:1 https://mirrors.dc.clear.net.ar/ubuntu jammy/universe amd64 tree amd64 2.0.2-
1 [47,9 kB]
Fetched 47,9 kB in 1s (50,3 kB/s)
Selecting previously unselected package tree.
(Reading database ... 688540 files and directories currently installed.)
Preparing to unpack .../tree_2.0.2-1_amd64.deb ...
Unpacking tree (2.0.2-1) ...
Setting up tree (2.0.2-1) ...
Processing triggers for man-db (2.10.2-1) ...
```

Once the tree program is installed it can be run where all subdirectories of the root folder (/) are shown, but this will scroll for a long while as it will show ALL subdirectories of the whole OS! To limit this to just your home directory, we type a space and a full stop after the tree command:

```
tree .
```

This partial view shows the file structure subfolder/file relationship:


stevee@6530b: ~
File Edit View Search Terminal Help
Thursday, November 2, 20:14

```
│   │       ├── SP_A0195.jpg
│   │       ├── SP_A0196.jpg
│   │       ├── SP_A0197.jpg
│   │       ├── SP_A0198.jpg
│   │       ├── SP_A0199.jpg
│   │       └── SP_A0200.jpg
│   ├── Quals
│   │   ├── 2k.jpg
│   │   ├── AET.jpeg
│   │   ├── AlisonCerts.zip
│   │   ├── Cisco.jpg
│   │   ├── Fast Track Joining Instructions - S Edwards.pdf
│   │   ├── FdSc.jpeg
│   │   ├── FdScResults1.jpg
│   │   ├── ITQuals.zip
│   │   ├── Learner-Record-3937893.pdf
│   │   ├── MCPCert.jpg
│   │   ├── MCPID.jpg
│   │   ├── Microsoft Certified Professional2001.jpg
│   │   ├── MusicTechBTEC.jpg
│   │   ├── NLP.pdf
│   │   ├── NT4.jpg
│   │   ├── PlymouthUni.jpg
│   │   ├── SDHBTEC.jpg
│   │   ├── S.Edwards Gen CV.docx
│   │   ├── stephenedwards-NDGLinuxEs-certificate.pdf
│   │   ├── Steve_Edwards_MarineCV_4p.docx
│   │   └── TTT.jpeg
│   ├── Stuff
│   │   ├── Aracati$0440.txt
│   │   ├── Wallet.jpg
│   │   ├── WalletNums.txt
│   │   ├── Wallet.txt
│   │   ├── ZmAddr.txt
│   │   └── zumm.txt
│   └── WinCommands.txt
├── Pictures
│   ├── Cov.jpg
│   ├── Cov.png
│   ├── Desktop.png
│   └── Sapay Mask.png
├── Replaces
├── Videos
└── Warpinator

5533 directories, 89238 files
stevee@6530b:~$ tree .
```

The full stop after the command is a file system designator for a branch point (inode) in the filesystem two directories above where you are in the tree. So, in file system vertical terms, it can be visualised:

.
..
current working directory

This is why you can move up (cd = change directory) one directory from where you are, using:

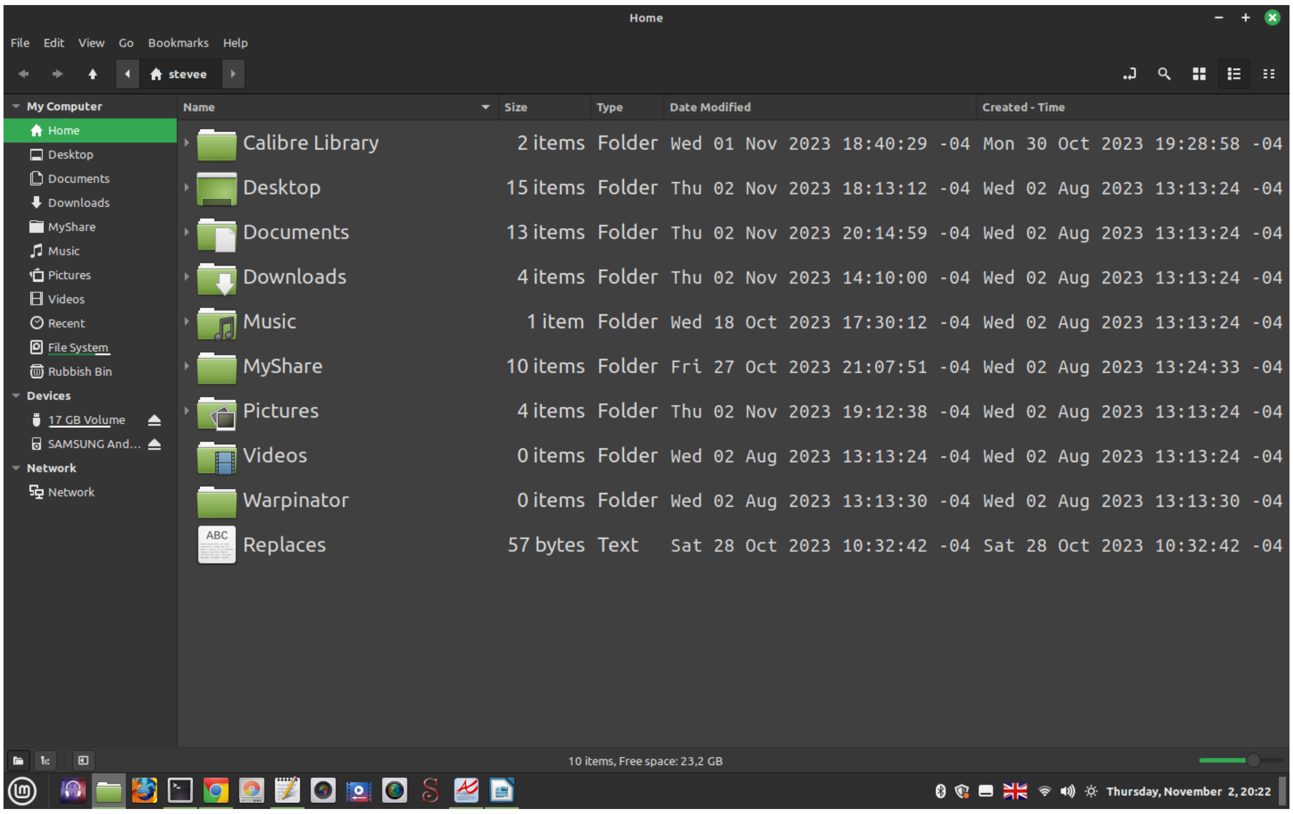
```
cd ..
```

The two dots (..) represent the directory above you.

So, how do you know where you are in a file system structure? Use the command pwd, or present working directory:

```
stevee@laptop:~$ pwd
/home/stevee
```

This tells me I am in my home directory, which in the GUI view is:



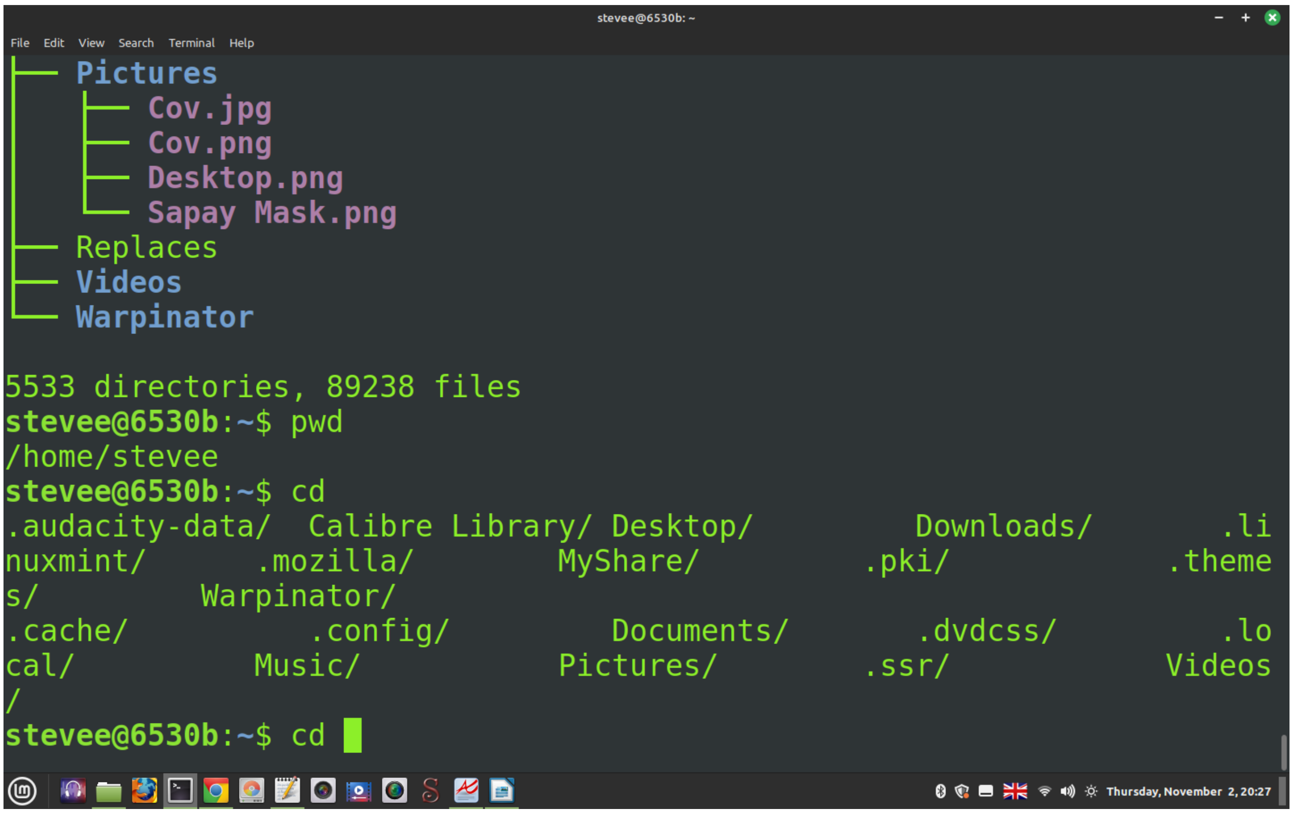
Name	Size	Type	Date Modified	Created - Time
Calibre Library	2 items	Folder	Wed 01 Nov 2023 18:40:29 -04	Mon 30 Oct 2023 19:28:58 -04
Desktop	15 items	Folder	Thu 02 Nov 2023 18:13:12 -04	Wed 02 Aug 2023 13:13:24 -04
Documents	13 items	Folder	Thu 02 Nov 2023 20:14:59 -04	Wed 02 Aug 2023 13:13:24 -04
Downloads	4 items	Folder	Thu 02 Nov 2023 14:10:00 -04	Wed 02 Aug 2023 13:13:24 -04
Music	1 item	Folder	Wed 18 Oct 2023 17:30:12 -04	Wed 02 Aug 2023 13:13:24 -04
MyShare	10 items	Folder	Fri 27 Oct 2023 21:07:51 -04	Wed 02 Aug 2023 13:24:33 -04
Pictures	4 items	Folder	Thu 02 Nov 2023 19:12:38 -04	Wed 02 Aug 2023 13:13:24 -04
Videos	0 items	Folder	Wed 02 Aug 2023 13:13:24 -04	Wed 02 Aug 2023 13:13:24 -04
Warpinator	0 items	Folder	Wed 02 Aug 2023 13:13:30 -04	Wed 02 Aug 2023 13:13:30 -04
Replaces	57 bytes	Text	Sat 28 Oct 2023 10:32:42 -04	Sat 28 Oct 2023 10:32:42 -04

10 items, Free space: 23,2 GB

Any time you lose track of where in the tree structure you are, you can use pwd to find yourself, or type in “cd” for change directory, which, with no further arguments after it will take you back to your Home directory. This is a key command for moving around the file system, IF you have the permissions to enter a particular folder as a particular user with particular privileges, but more on that in days 3-4. e.g.:

Move into your Documents folder (you can use the Tab key, with a space after the cd command, to see all available folder options):

cd (TAB key)

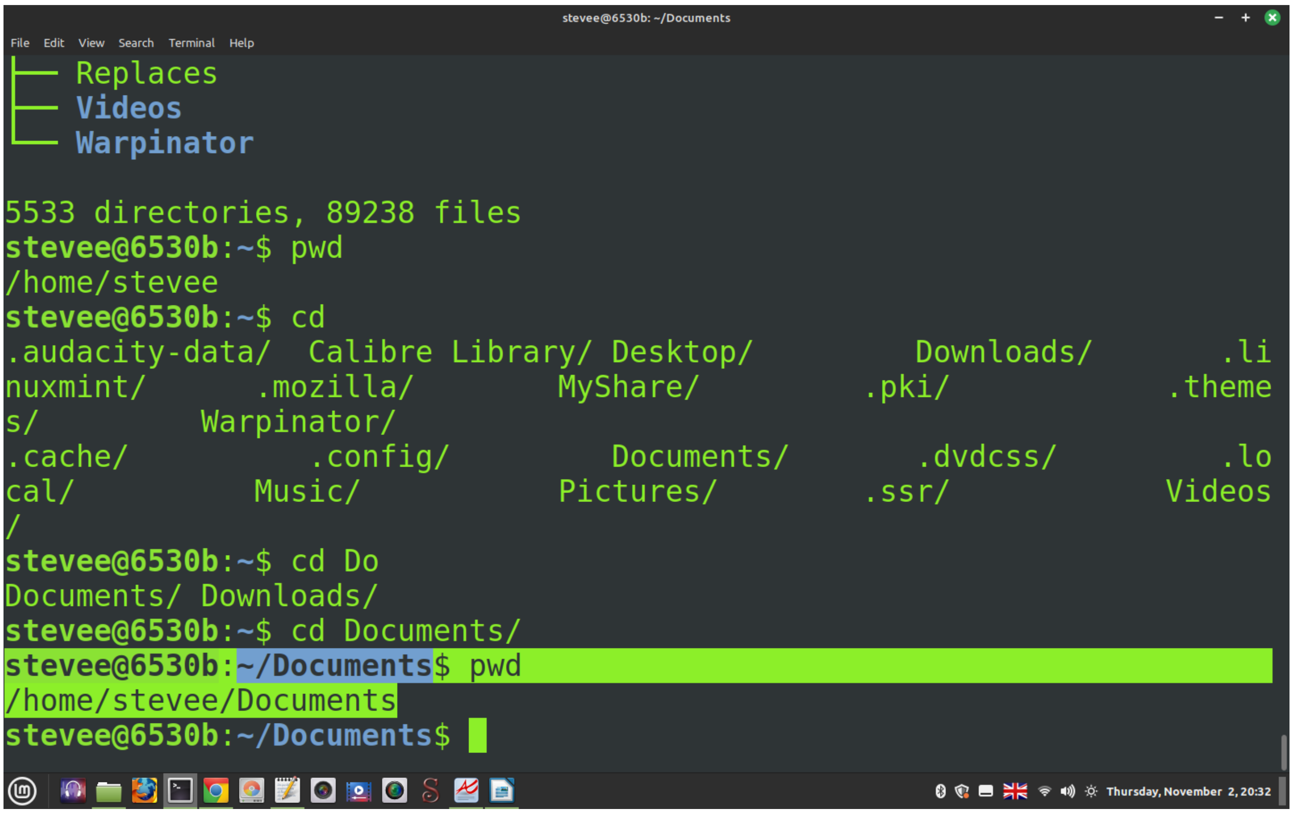


```
├── Pictures
│   ├── Cov.jpg
│   ├── Cov.png
│   ├── Desktop.png
│   └── Sapay Mask.png
├── Replaces
├── Videos
└── Warpinator

5533 directories, 89238 files
stevee@6530b:~$ pwd
/home/stevee
stevee@6530b:~$ cd
.audacity-data/  Calibre Library/ Desktop/         Downloads/        .li
nuxmint/      .mozilla/         MyShare/          .pki/             .theme
s/          Warpinator/
.cache/          .config/          Documents/        .dvdcss/          .lo
cal/          Music/            Pictures/         .ssr/             Videos
/
stevee@6530b:~$ cd 
```

The Documents folder is listed, so you can type:

cd Doc[TAB]

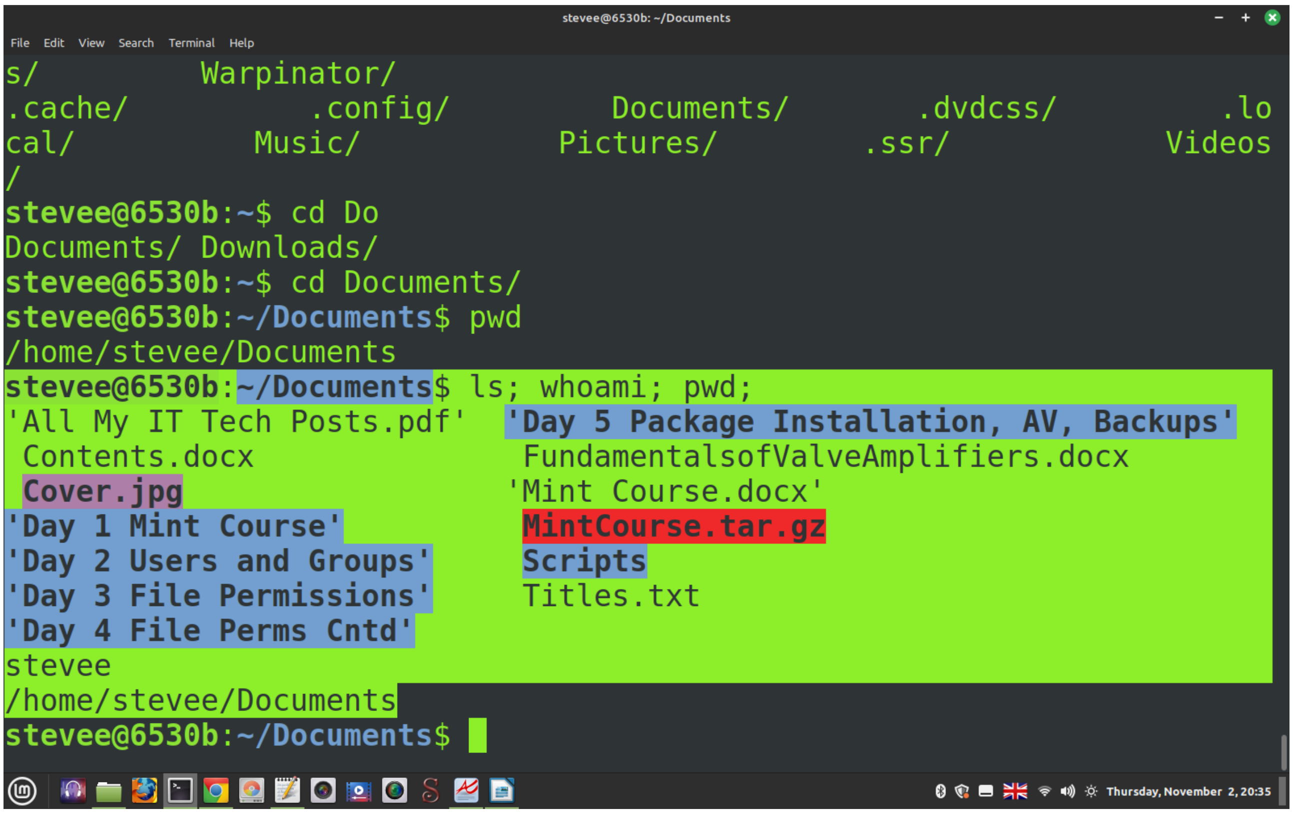
and Tab Completion will complete the only available folder name - Documents - for you.
On Enter, you will cd into that folder.
Use pwd to confirm for yourself by typing, or using the UP arrow to bring pwd back from the command history:


stevee@6530b: ~/Documents
File Edit View Search Terminal Help
Thursday, November 2, 20:32

```
├── Replaces
├── Videos
└── Warpinator

5533 directories, 89238 files
stevee@6530b:~$ pwd
/home/stevee
stevee@6530b:~$ cd
.audacity-data/  Calibre Library/ Desktop/          Downloads/        .li
nuxmint/      .mozilla/         MyShare/          .pki/             .theme
s/          Warpinator/
.cache/           .config/          Documents/        .dvdcss/          .lo
cal/          Music/            Pictures/         .ssr/             Videos
/
stevee@6530b:~$ cd Do
Documents/ Downloads/
stevee@6530b:~$ cd Documents/
stevee@6530b:~/Documents$ pwd
/home/stevee/Documents
stevee@6530b:~/Documents$ 
```

Commands in Linux can be run serially with the appropriate semicolon separators and will output in order:

ls; whoami; pwd;


stevee@6530b: ~/Documents
File Edit View Search Terminal Help
Thursday, November 2, 20:35

```
s/           Warpinator/
.cache/            .config/           Documents/        .dvdcss/          .lo
cal/           Music/            Pictures/         .ssr/             Videos
/
stevee@6530b:~$ cd Do
Documents/ Downloads/
stevee@6530b:~$ cd Documents/
stevee@6530b:~/Documents$ pwd
/home/stevee/Documents
stevee@6530b:~/Documents$ ls; whoami; pwd;
'All My IT Tech Posts.pdf'  'Day 5 Package Installation, AV, Backups'
 Contents.docx               FundamentalsofValveAmplifiers.docx
 Cover.jpg                  'Mint Course.docx'
'Day 1 Mint Course'          MintCourse.tar.gz
'Day 2 Users and Groups'     Scripts
'Day 3 File Permissions'     Titles.txt
'Day 4 File Perms Cntd'
stevee
/home/stevee/Documents
stevee@6530b:~/Documents$ 
```

From now on, as you have the idea of the Terminal view, I will mostly paste the output text to save image page space. The previous commands in order were:

```
stevee@laptop:~/Documents$ ls; whoami; pwd;
'All My IT Tech Posts.pdf'  'Day 5 Package Installation, AV, Backups'
 Contents.docx               FundamentalsofValveAmplifiers.docx
 Cover.jpg                  'Mint Course.docx'
'Day 1 Mint Course'          MintCourse.tar.gz
'Day 2 Users and Groups'     Scripts
'Day 3 File Permissions'     Titles.txt
'Day 4 File Perms Cntd'
stevee
/home/stevee/Documents
```

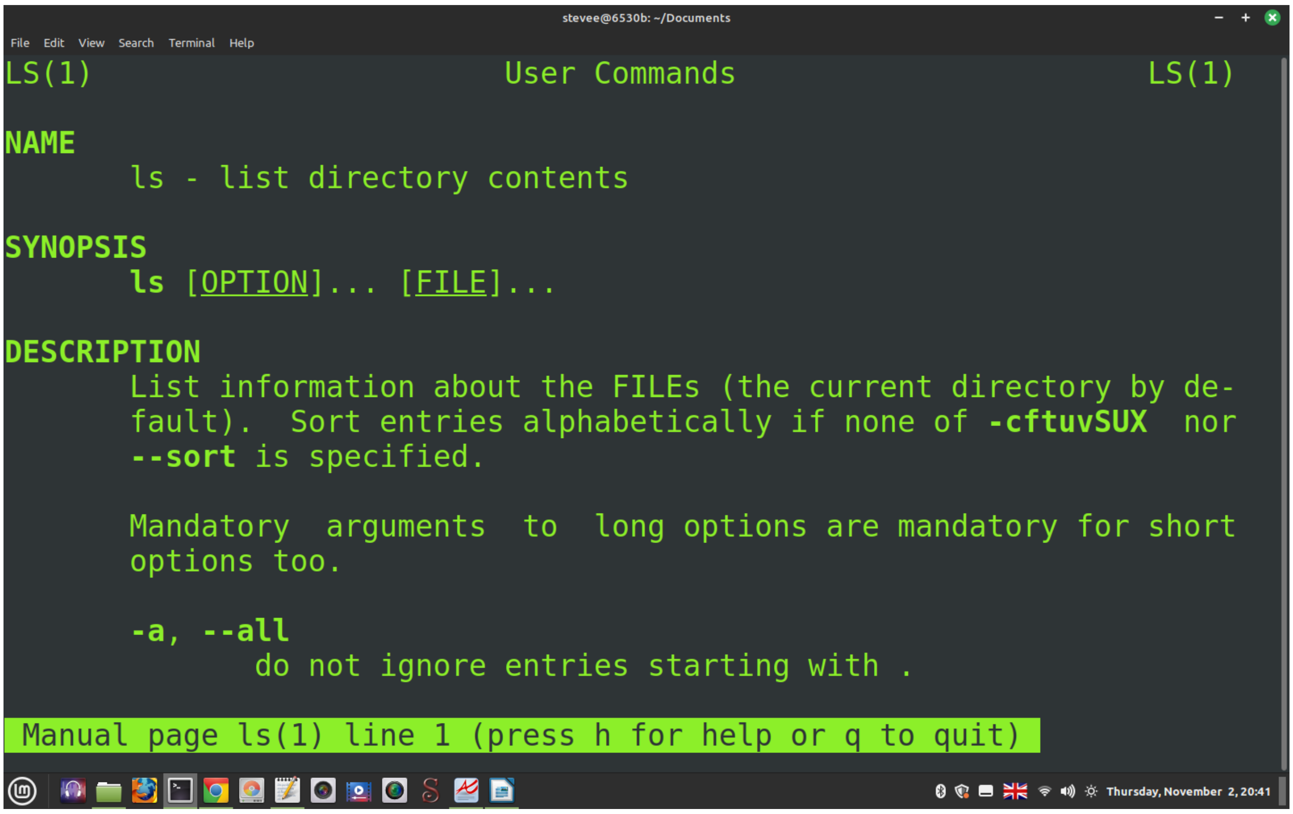
This shows the listing of all the files in the directory; who am logged in as (stevee); and the present working directory (/home/stevee/Documents).

The first command “ls” means “list” and can be followed by different “switches” to give more information about the files listed by it.

Most Linux commands can have a -h or --help switch to show what options are available for that command, or you can read the manual page for that command e.g.:

```
man ls
```

q to quit


stevee@6530b: ~/Documents
File Edit View Search Terminal Help
Thursday, November 2, 20:41

```
LS(1)                        User Commands                        LS(1)

NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List information about the FILEs (the current directory by de-
       fault).  Sort entries alphabetically if none of -cftuvSUX  nor
       --sort is specified.

       Mandatory  arguments  to  long options are mandatory for short
       options too.

       -a, --all
              do not ignore entries starting with .

 Manual page ls(1) line 1 (press h for help or q to quit)
```

So, for the ls command, we can view:

```
stevee@laptop:~$ cd; ls -l
total 40
drwxrwxr-x 2 stevee stevee 4096 Nov  1 18:40 'Calibre Library'
drwxr-xr-x 4 stevee stevee 4096 Nov  2 18:13  Desktop
drwxr-xr-x 8 stevee stevee 4096 Nov  2 20:42  Documents
drwxr-xr-x 3 stevee stevee 4096 Nov  2 14:10  Downloads
drwxr-xr-x 3 stevee stevee 4096 Oct 18 17:30  Music
drwxrwxr-x 9 stevee stevee 4096 Oct 27 21:07  MyShare
drwxr-xr-x 2 stevee stevee 4096 Nov  2 19:12  Pictures
-rw-rw-r-- 1 stevee stevee   57 Oct 28 10:32  Replaces
drwxr-xr-x 2 stevee stevee 4096 Aug  2 13:13  Videos
drwxrwxr-x 2 stevee stevee 4096 Aug  2 13:13  Warpinator
```

From left to right, the 7 columns mean:

File permissions: Replication factor: User: Group: bytes size on disk: File creation date/time: Folder/filename

I'll explain the drwxrwxr-x columns file permission blocks in detail on Days 3 and 4.

The Replication factor (default = 3) is the number of copies a file has in the filesystem to guard against corruption.

Root and Sudo Users

The second serial command was whoami. It simply tells you who you are logged in as at present – this is because the "root" user has system wide Administrator permissions so can log in as any other user at any time for Sysadmin purposes, so you need to know who you are at all times before issuing commands to prevent serious errors that may damage the system!

"With great power comes great responsibility!"

Beware root user command errors! Whole systems can be destroyed by incorrect command usage!

Don't EVER use the recursive "remove" command:

```
rm -vr /*
```

It will remove the whole root filesystem without warning!!!

```
stevee@laptop:~$ whoami
stevee
```

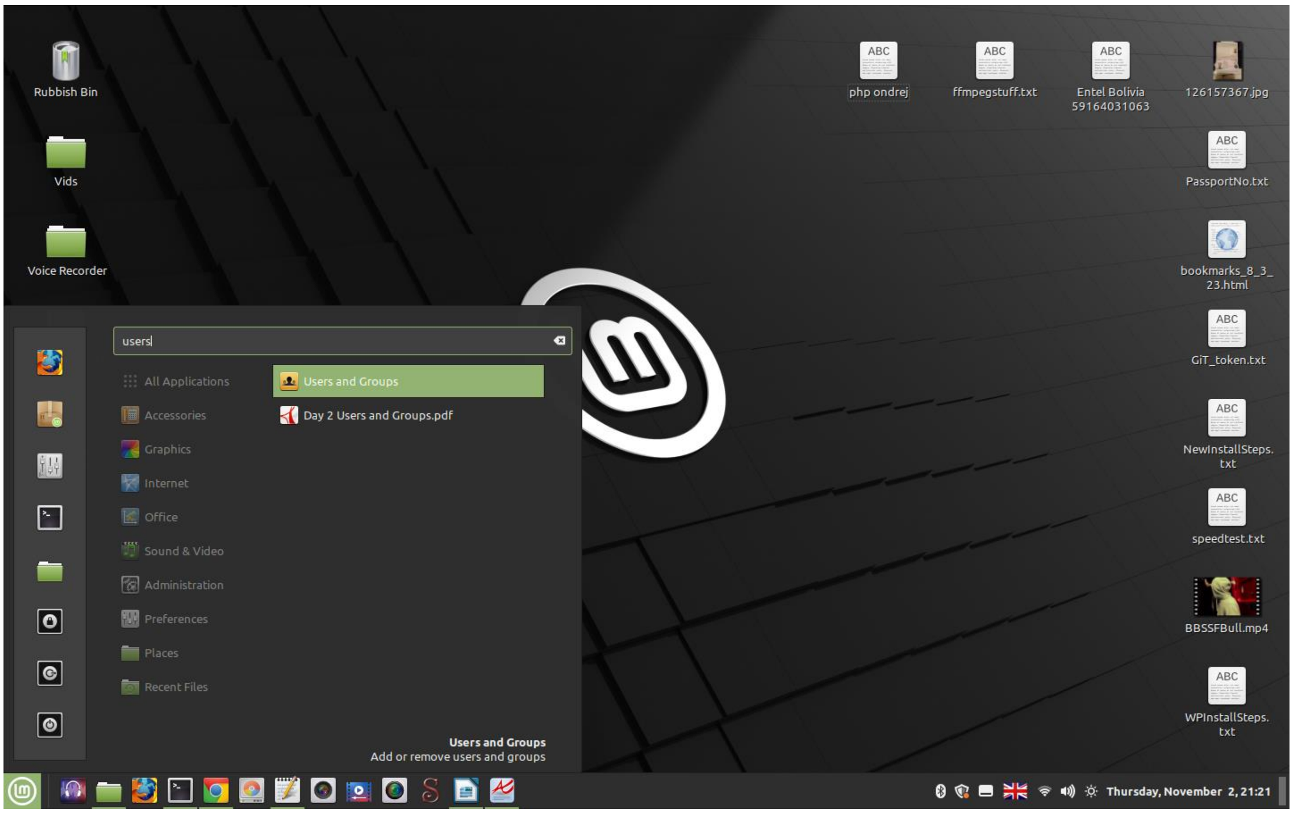
The installer of an Linux OS is given "sudo" or Super User Do permissions, so user stevee can act as root by logging in as the root user using the sudo user password.
The root user password should be changed immediately the OS is installed for security reasons.

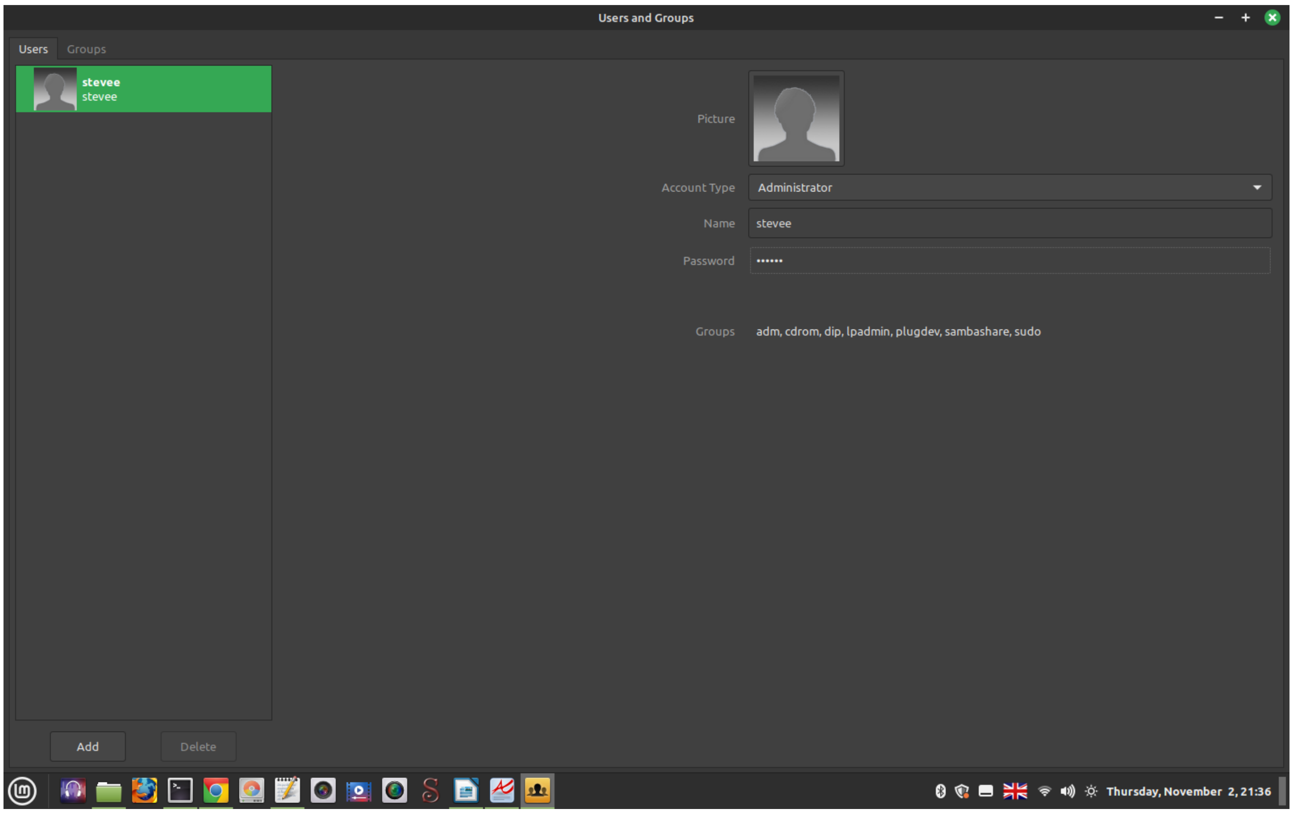
This is achieved using the passwd command with the sudo prefix which temporarily elevates your sudo allowed user to access root level privileges to accomplish the task:

```
stevee@laptop:~$ sudo passwd root
[sudo] password for stevee:
New password:
Retype new password:
passwd: password updated successfully
```

Day 2: Admin, Users and Groups

The OS has a GUI for adding Users and Groups by searching the Menu button:





Picture
Account Type: Administrator
Name: stevee
Password: ••••••
Groups: adm, cdrom, dip, lpadmin, plugdev, sambashare, sudo

The changeable user options are:

Photo; Account type (admin or standard user); User Name; Password; Group Memberships

As the OS installer, you see I am in the sudo group, which allows me elevation privileges to root user level so I can manage all aspects of the OS – including all other users!

```
man sudo
sudo, sudoedit — execute a command as another user
DESCRIPTION
sudo allows a permitted user to execute a command as the superuser or
another user, as specified by the security policy.
sudo supports a plugin architecture for security policies and input/out-
put logging.
```

So, sudo “logs” the sudo user's activity, so Admin staff know who did what, when.

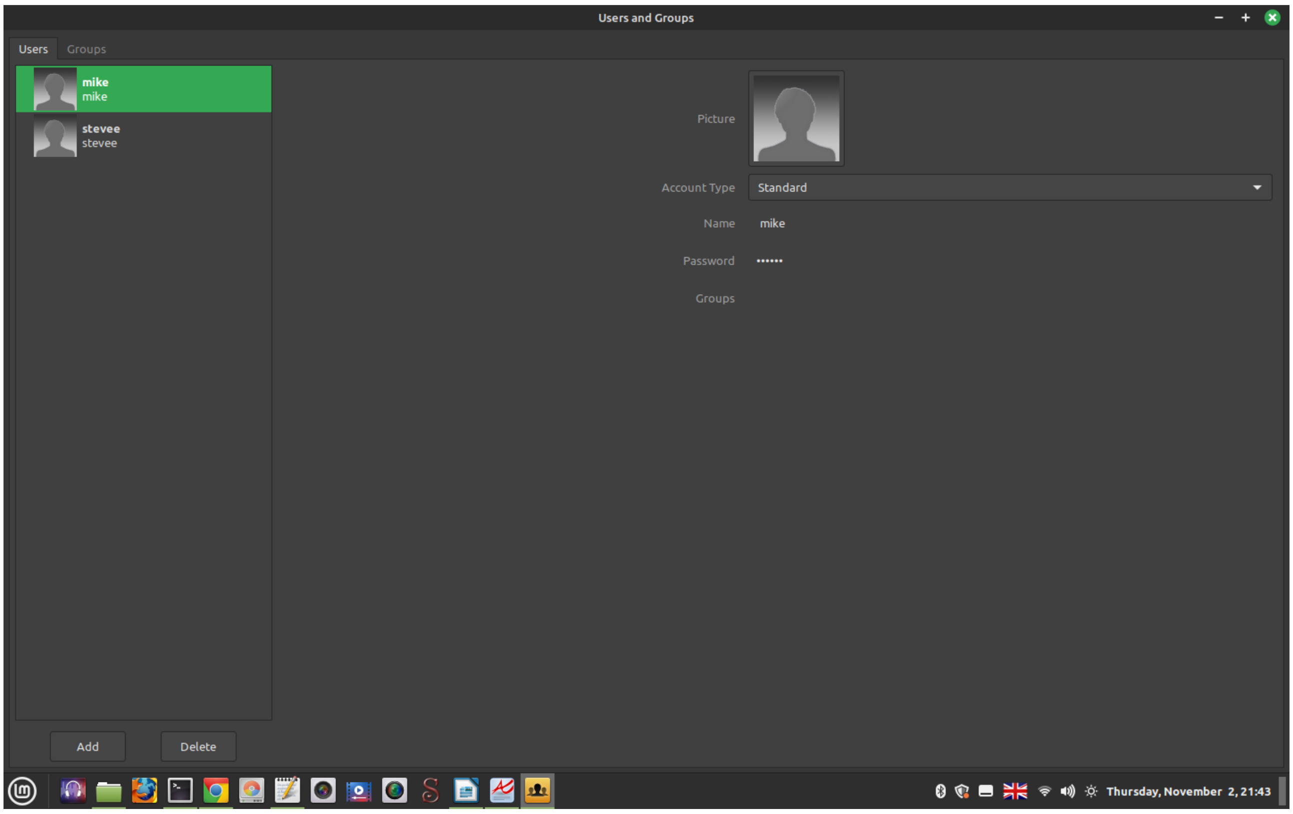
Using these Superuser powers, I can add another user to the system who can then login to their own account when ready. As this is obvious within the User/Groups GUI via the Add button bottom left, I’ll show you the command line method:

```
stevee@laptop:~$ sudo adduser mike
Adding user `mike' ...
Adding new group `mike' (1001) ...
Adding new user `mike' (1001) with group `mike' ...
```

```
Creating home directory `/home/mike' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for mike
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n]
```

Note a major difference between the GUI and the Terminal is that the GUI password length minimum is 8 characters, but the Terminal allows a single character password!
Older Linux systems allowed a blank password via the Terminal method - a security risk. More on that later.

To view the new user and their defaults in the GUI it needs closing and re-opening:



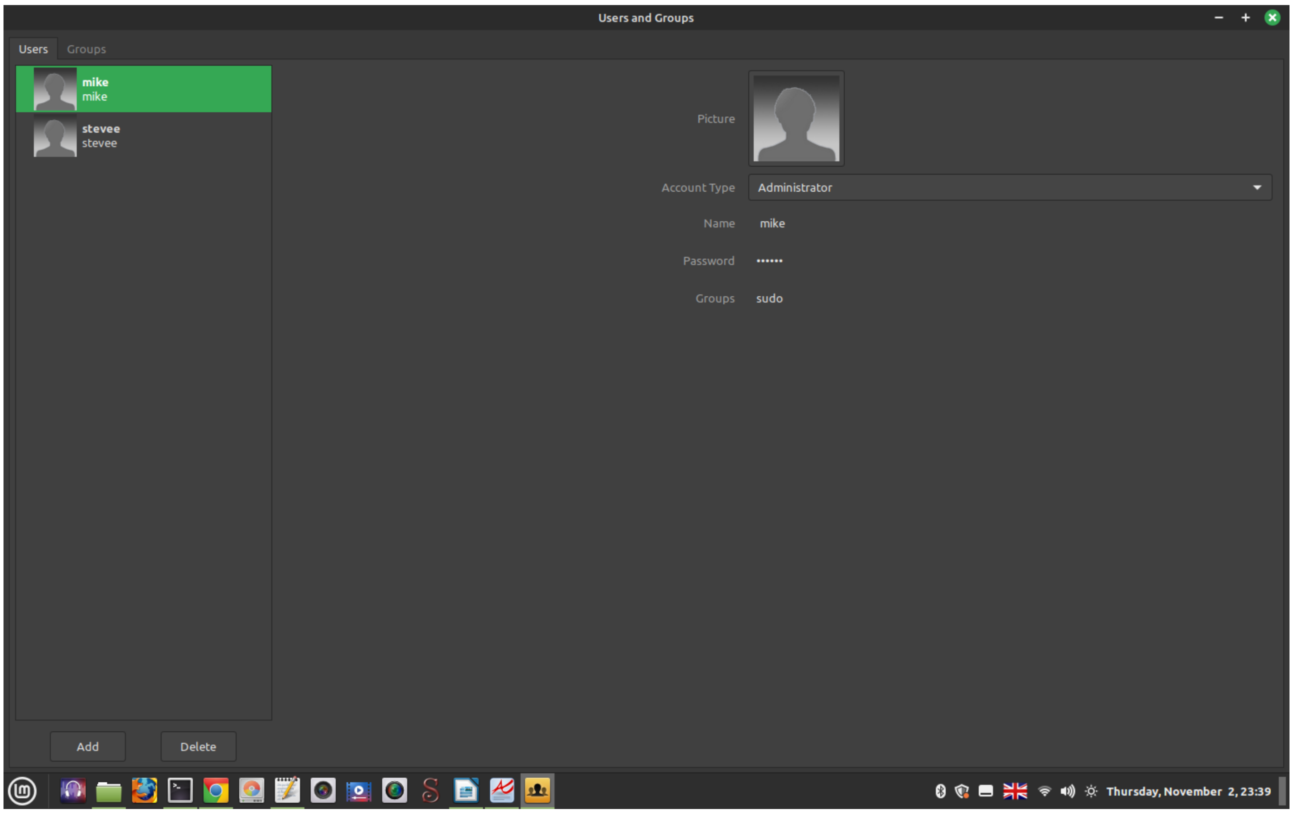
Picture
Account Type: Standard
Name: mike
Password: ••••••
Groups:

The new user “mike” is shown a standard privilege user with no group memberships.

So, what if I wanted to elevate new user mike to a sudo admin level user?

```
stevee@laptop:~$ sudo addgroup mike sudo
Adding user `mike' to group `sudo' ...
Adding user mike to group sudo
Done.
```

Now if we look at mike in the GUI, he's now an Administrator in the sudo group:



Picture
Account Type: Administrator
Name: mike
Password: ••••••
Groups: sudo

Linux is a text file-based OS, so even hardware like hard drives are converted to text file aliases by the system, which also means that the user and group information is also stored as text files, but obviously hashed for security reasons by appropriate design.

Some important user and group files to know of relating to admin security are the /etc/group/, /etc/passwd and /etc/shadow files. What is inside and why?

Using the command "cat" to read the contents of these files (for con**cat**enate or joining files and text together, often), we can see the file content.

First, to know all the different groups that comprise a Linux system, look in:

```
stevee@laptop:~$ cat /etc/group
root:x:0:
daemon:x:1:
bin:x:2:
```

```
sys:x:3:
adm:x:4:syslog,stevee
tty:x:5:
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
uucp:x:10:
man:x:12:
proxy:x:13:
kmem:x:15:
dialout:x:20:
fax:x:21:
voice:x:22:
cdrom:x:24:stevee
floppy:x:25:
tape:x:26:
sudo:x:27:stevee,mike
audio:x:29:pulse
dip:x:30:stevee
www-data:x:33:
backup:x:34:
operator:x:37:
list:x:38:
irc:x:39:
src:x:40:
gnats:x:41:
shadow:x:42:
utmp:x:43:
video:x:44:
sasl:x:45:
plugdev:x:46:stevee
staff:x:50:
games:x:60:
users:x:100:
nogroup:x:65534:
systemd-journal:x:101:
systemd-network:x:102:
systemd-resolve:x:103:
crontab:x:104:
messagebus:x:105:
systemd-timesync:x:106:
input:x:107:
sgx:x:108:
kvm:x:109:
render:x:110:
syslog:x:111:
tss:x:112:
```

```
rtkit:x:113:
systemd-coredump:x:114:
lpadmin:x:115:stevee
bluetooth:x:116:
ssl-cert:x:117:
netdev:x:118:
uuidd:x:119:
lightdm:x:120:
nopasswdlogin:x:121:
tcpdump:x:122:
_ssh:x:123:
plocate:x:124:
avahi-autoipd:x:125:
nm-openvpn:x:126:
geoclue:x:127:
pulse:x:128:
pulse-access:x:129:
scanner:x:130:saned
_flatpak:x:131:
avahi:x:132:
saned:x:133:
colord:x:134:
fwupd-refresh:x:135:
stevee:x:1000:
sambashare:x:136:stevee
mysql:x:137:
mike:x:1001:
```

Remember I just added new user mike to the system? He is shown as a group here also and by default the user mike is assigned into his own group - mike, but that does NOT show in the GUI, neither does user stevee in group stevee.

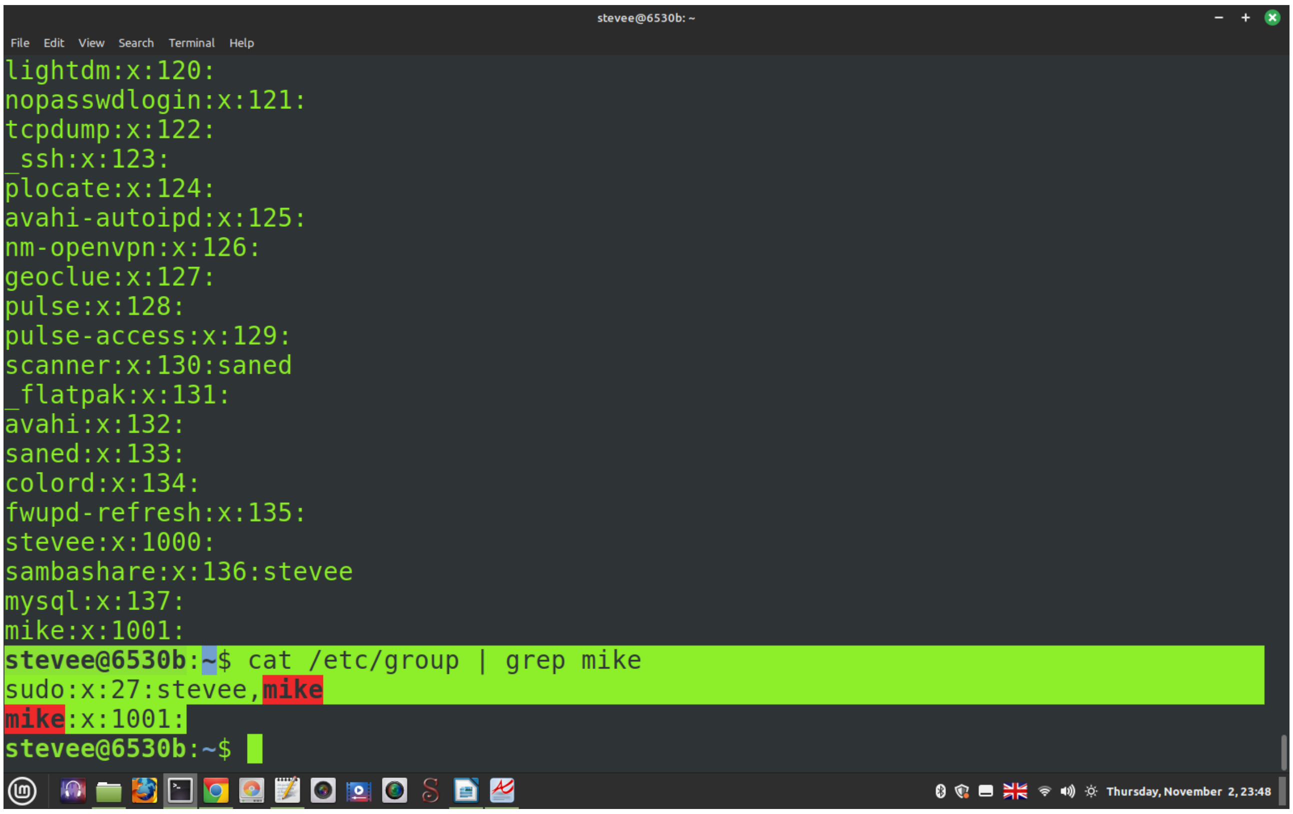
The OS install user group (stevee) starts at number 1000, so group mike as the second user added is in group number 1001.

Also notice that the sudo group contains users stevee and mike.

This may be a quick way for a sysadmin to check who is in what groups, so what overall system privileges they have, for example, in a very busy system with many users.
A sysadmin could search this file for a particular username to limit the many lines of output, using the concept of "piping", whereby the output of one command becomes the input to another! Powerful and efficient:

```
cat /etc/group | grep mike
```

grep is a character search program that will find all instances of a string in a file or directory.
The | character is called a "pipe", hence the term "piping" output to input.


stevee@6530b: ~
File Edit View Search Terminal Help
Thursday, November 2, 23:48

```
lightdm:x:120:
nopasswdlogin:x:121:
tcpdump:x:122:
_ssh:x:123:
plocate:x:124:
avahi-autoipd:x:125:
nm-openvpn:x:126:
geoclue:x:127:
pulse:x:128:
pulse-access:x:129:
scanner:x:130:saned
_flatpak:x:131:
avahi:x:132:
saned:x:133:
colord:x:134:
fwupd-refresh:x:135:
stevee:x:1000:
sambashare:x:136:stevee
mysql:x:137:
mike:x:1001:
stevee@6530b:~$ cat /etc/group | grep mike
sudo:x:27:stevee,mike
mike:x:1001:
stevee@6530b:~$
```

This shows that user mike belongs to groups mike and sudo.
Note that grep colourises the search string on a successful find.

The x in the second column of the /etc/group file means that a password has been set but not shown in this file for plain text security reasons as it is hashed and logged in the next important file, /etc/shadow.
As it is a security risk, only root level privileges can access it by user stevee (and now user mike) becoming root temporarily by the sudo command:

```
sudo cat /etc/shadow
```

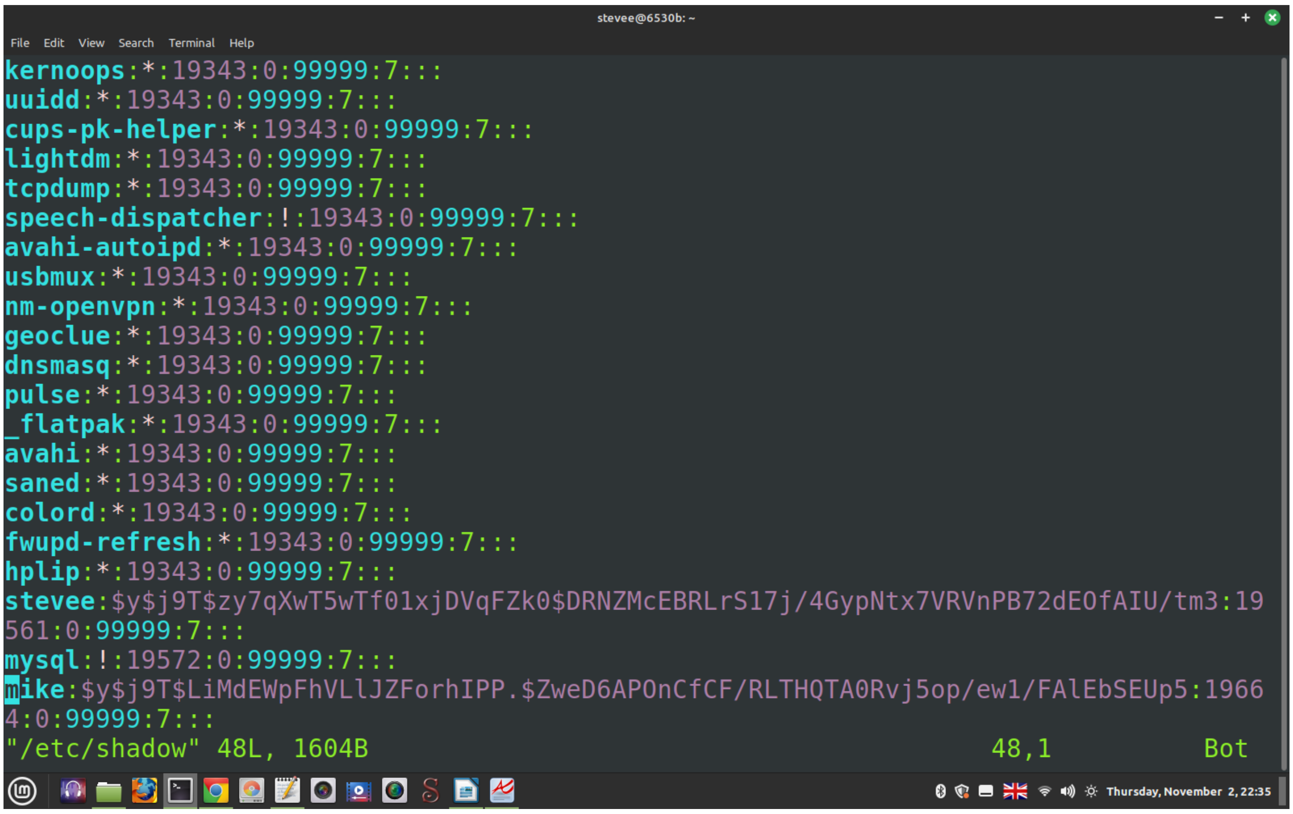
```
[sudo] password for stevee:
root:$y$j9T$sVqSVl60oDMrmZ5bI.aV00$0I9S.ketFYMDc5U9i.eBP67bIETCloj9yxhFH55h9
/1:19664:0:99999:7:::
daemon:*:19343:0:99999:7:::
bin:*:19343:0:99999:7:::
sys:*:19343:0:99999:7:::
sync:*:19343:0:99999:7:::
games:*:19343:0:99999:7:::
man:*:19343:0:99999:7:::
lp:*:19343:0:99999:7:::
mail:*:19343:0:99999:7:::
news:*:19343:0:99999:7:::
uucp:*:19343:0:99999:7:::
proxy:*:19343:0:99999:7:::
```

```
www-data:*:19343:0:99999:7:::
backup:*:19343:0:99999:7:::
list:*:19343:0:99999:7:::
irc:*:19343:0:99999:7:::
gnats:*:19343:0:99999:7:::
nobody:*:19343:0:99999:7:::
systemd-network:*:19343:0:99999:7:::
systemd-resolve:*:19343:0:99999:7:::
messagebus:*:19343:0:99999:7:::
systemd-timesync:*:19343:0:99999:7:::
syslog:*:19343:0:99999:7:::
_apt:*:19343:0:99999:7:::
tss:*:19343:0:99999:7:::
rtkit:*:19343:0:99999:7:::
systemd-coredump:*:19343:0:99999:7:::
kernoops:*:19343:0:99999:7:::
uuidd:*:19343:0:99999:7:::
cups-pk-helper:*:19343:0:99999:7:::
lightdm:*:19343:0:99999:7:::
tcpdump:*:19343:0:99999:7:::
speech-dispatcher:!:19343:0:99999:7:::
avahi-autoipd:*:19343:0:99999:7:::
usbmux:*:19343:0:99999:7:::
nm-openvpn:*:19343:0:99999:7:::
geoclue:*:19343:0:99999:7:::
dnsmasq:*:19343:0:99999:7:::
pulse:*:19343:0:99999:7:::
_flatpak:*:19343:0:99999:7:::
avahi:*:19343:0:99999:7:::
saned:*:19343:0:99999:7:::
colord:*:19343:0:99999:7:::
fwupd-refresh:*:19343:0:99999:7:::
hplip:*:19343:0:99999:7:::
stevee:$y$j9T$zy7qXwT5wTf01xjDVqFZk0$DRNZMcEBRLrS17j/4GypNtx7VRVnPB72dE
OfAIU/tm3:19561:0:99999:7:::
mysql:!:19572:0:99999:7:::
mike:$y$j9T$LiMdEWpFhVLlJZForhIPP.$ZweD6APOnCfCF/RLTHQTA0Rvj5op/ew1/FAl
EbSEUp5:19664:0:99999:7:::
```

This shows that most service groups have an asterisk * in the second column, which means there is no password set for the group (it wouldn't make sense having a service password required by the system to run a service of the system!), but it shows that the groups that have user accounts = root, stevee and mike, have a long string of characters in the second column that means that a hashed password HAS been set for the user in that group.

If I open it using vim, its a lot clearer:

```
sudo vim /etc/shadow
```


stevee@6530b: ~
File Edit View Search Terminal Help
Thursday, November 2, 22:35

```
kernoops:*:19343:0:99999:7:::
uuidd:*:19343:0:99999:7:::
cups-pk-helper:*:19343:0:99999:7:::
lightdm:*:19343:0:99999:7:::
tcpdump:*:19343:0:99999:7:::
speech-dispatcher:!:19343:0:99999:7:::
avahi-autoipd:*:19343:0:99999:7:::
usbmux:*:19343:0:99999:7:::
nm-openvpn:*:19343:0:99999:7:::
geoclue:*:19343:0:99999:7:::
dnsmasq:*:19343:0:99999:7:::
pulse:*:19343:0:99999:7:::
_flatpak:*:19343:0:99999:7:::
avahi:*:19343:0:99999:7:::
saned:*:19343:0:99999:7:::
colord:*:19343:0:99999:7:::
fwupd-refresh:*:19343:0:99999:7:::
hplip:*:19343:0:99999:7:::
stevee:$y$j9T$zy7qXwT5wTf01xjDVqFZk0$DRNZMcEBRLrS17j/4GypNtx7VRVnPB72dEOfAIU/tm3:19
561:0:99999:7:::
mysql:!:19572:0:99999:7:::
mike:$y$j9T$LiMdEWpFhVLlJZForhIPP.$ZweD6APOnCfCF/RLTHQTA0Rvj5op/ew1/FAlEbSEUp5:1966
4:0:99999:7:::
"/etc/shadow" 48L, 1604B                                        48,1          Bot
```

Here we see that the passwords that are set for users stevee and mike are hashed in the second column after the colon column separators, starting with **$y$**.

There are many types of encryption and “one way hash” methods used for securing passwords (and other text) in computer systems and the one used for Ubuntu and Mint uses “yescrypt” shown by the **$y$** in /etc/shadow file second column prefix:

yescrypt

yescrypt is a scalable passphrase hashing scheme designed by Solar Designer, which is based on Colin Percival's scrypt. Recommended for new hashes.

Prefix

"$y$"

Hashed passphrase format

\$y\$[./A-Za-z0-9]+\$[./A-Za-z0-9]{,86}\$[./A-Za-z0-9]{43}

Maximum passphrase length

unlimited

Hash size

256 bits

Salt size

up to 512 bits

So why would an Administrator want to know this? Its another possible way to quickly check what users may not have a password set on their account by checking for the **$y$** string in the shadow file second column – a useful check is whether the root user has had a password added after the initial system install!
If not, a (!) will be present in the second column, so needs tending to!

```
sudo cat /etc/shadow | grep '$y$*'
root:$y$j9T$sVqSVl60oDMrmZ5bI.aV00$0I9S.ketFYMDc5U9i.eBP67bIETCloj9yxhFH55h9/1:1
9664:0:99999:7:::
stevee:$y$j9T$zy7qXwT5wTf01xjDVqFZk0$DRNZMcEBRLrS17j/4GypNtx7VRVnPB72dEOfAI
U/tm3:19561:0:99999:7:::
mike:$y$j9T$LiMdEWpFhVLlJZForhIPP.$ZweD6APOnCfCF/RLTHQTA0Rvj5op/ew1/FAlEbSE
Up5:19664:0:99999:7:::
```

Ok, a difficult example to explain…

This shows only the 3 users that have passwords set and hashed by the yescrypt hash process.

If there were more users on the system but not showing here from this search, then they cannot have had passwords set, so the admin would have to check why not – not a great real-world example, but you get the point?

We saw from the earlier User and Groups example that it is mandatory to set at least a 1-character password in this Linux version, but past versions it was not, so blank passwords were possible, so a security risk.

Linux has certain “special” reserved characters that the OS recognises for command line parsing purposes, which are:

`	Command
#	Comment
$	Variable
&	Background
*	String
(	Start
)	End
\	Quote
|	Pipe
[	Start
]	End

{ Start
} End
; Shell
' Strong
<"> Weak
< Input
> Output
/ Pathname
? Single-character
! Pipeline

To actually search for these in a text document they have to be “identified” as “non” special to prevent the system treating them as command line “system code” characters.

In this case, the $ is a special character so has to be pre and post fixed by another special character, the (' Strong). Also, Linux differentiates between the commands on the command line using white space between commands and search string characters.

As there is no white space between the **y** string and the following characters, grep cannot identify just the **y** component alone without further wizardry.

The way round this issue is to follow the **y** with the special “wildcard” character *, which stands for substitute for “none or any number of characters”, so grep can find ***just*** the **y** part of the whole associated string:

yj9T$LiMdEWpFhVLlJZForhIPP.$ZweD6APOnCfCF/RLTHQTA0Rvj5op/ew1/FAlEbSEUp5:1
9664:0:99999:7:::

A simpler example of the useful wildcard * character is to find any MP3 files I have in my Music folder in my Home directory.

```
find Music/  *mp3
Music/
Music/CarnivalRide
Music/CarnivalRide/03SoSmall.mp3
Music/CarnivalRide/13WheeloftheWorld.mp3
Music/CarnivalRide/04JustaDream.mp3
Music/CarnivalRide/01FlatontheFloor.mp3
Music/CarnivalRide/12Twisted.mp3
Music/CarnivalRide/02All-AmericanGirl.mp3
Music/CarnivalRide/09YouWon'tFindThis.mp3
Music/CarnivalRide/10IToldYouSo.mp3
Music/CarnivalRide/06CrazyDreams.mp3
Music/CarnivalRide/11TheMoreBoysIMeet.mp3
find: ‘*mp3’: No such file or directory
```

Pattern searching and piping commands in Linux is an art and takes a lot of practice to ensure particular patterns definitely find what you’re looking for, especially if you are combining delete commands! For example, that find command can be amended to delete all those mp3 files once found:

```
find Music/ -name *mp3 -delete
```

Now if I re-run the prior find:

```
find Music/  *mp3
```

No output is shown as the Music/CarnivalRide/ folder is empty!

Those mp3 files don’t go into the Rubbish bin either so are not retrievable without special tools.

Irreversible damage can be done by incorrect multiple file deletion commands!

Always make sure you have backups before doing ANY deletion processes and try them first on a non-essential computer system or folder to be sure the command does what you expect!!

You have been warned!

There are many more pattern examples in my other eBook, “All My IT Knowledge Tech Posts”.

As the history file was mentioned earlier, you can run it to see past commands – both correct and incorrect, as I was trying to remember the delete option with find - should you want to re-run any by copying/pasting them again or using an exclamation mark in front of the command number:

```
history
797  find Music/ -name *mp3
  798  find Music/ -name *mp3 -exec delete {}
  799  find Music/ -name *mp3 -exec delete {}\
  800  find Music/ -name *mp3 -exec delete {}\;
  801  find Music/ -name *mp3 -delete {}\;
  802  find Music/ -name *mp3 -delete
  803  find Music/ -name *mp3
  804  history
```

```
!803
```

Back to file reading and text manipulation – head and tail – it should be obvious that the -3 reads only the first and last 3 lines of the file in each case:

```
stevee@laptop:~$ cat /etc/passwd | head -3
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

```
stevee@laptop:~$ cat /etc/passwd | tail -3
hplip:x:126:7:HPLIP system user,,,:/run/hplip:/bin/false
stevee:x:1000:1000:stevee,,,:/home/stevee:/bin/bash
```

```
mike:x:1001:1001:,,,:/home/mike:/bin/bash
```

A compact and capable text manipulation programming language called AWK comes with Linux, which is really good at stripping columns from text files, amongst many other things.

It can be used like cat to read a whole file, like the /etc/passwd example, also piped into tail -3 to get the same result, where $0 prints all columns, i.e. the whole file.

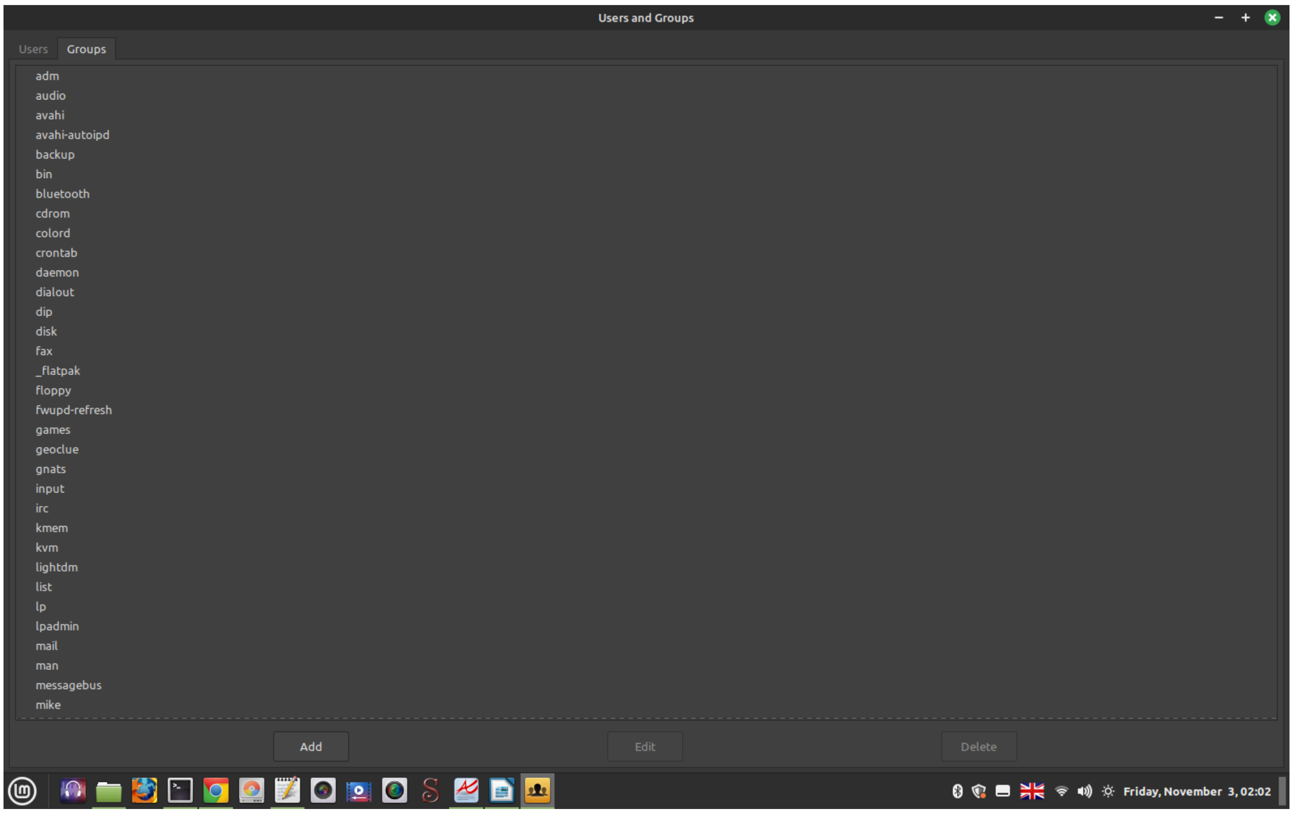
```
awk '{print $0}' /etc/passwd | tail -3
stevee:x:1000:1000:stevee,,,:/home/stevee:/bin/bash
mysql:x:127:137:MySQL Server,,,:/nonexistent:/bin/false
mike:x:1001:1001:,,,:/home/mike:/bin/bash
```

A column delimiter can be set to tell awk which column to identify.
The colon (:) is the delimiter, so that only certain data can be stripped as required.
If I wanted the group number info from column 3 of /etc/passwd, I tell awk which column and delimiter to use, so:

```
awk -F: '{print $3}' /etc/passwd | tail -3
1000
127
1001
```

You can see that the
-F:
states the column delimiter, and the $3 states the column number to print.

If you look at the User/Group GUI group tab you see a list of all system groups in alphabetical order:



```
adm
audio
avahi
avahi-autoipd
backup
bin
bluetooth
cdrom
colord
crontab
daemon
dialout
dip
disk
fax
_flatpak
floppy
fwupd-refresh
games
geoclue
gnats
input
irc
kmem
kvm
lightdm
list
lp
lpadmin
mail
man
messagebus
mike
```

You can get exactly the same list using Awk by column stripping the /etc/group file but then ordering it alphabetically by piping into the sort command:

```
awk -F: '{print $1}' /etc/group | sort
adm
audio
avahi
avahi-autoipd
backup
bin
bluetooth
cdrom
colord
crontab
daemon
dialout
dip
disk
fax
_flatpak
floppy
fwupd-refresh
games
geoclue
```

```
gnats
input
irc
kmem
kvm
lightdm
list
lp
lpadmin
mail
man
messagebus
mike
mysql
netdev
news
nm-openvpn
nogroup
nopasswdlogin
operator
plocate
plugdev
proxy
pulse
pulse-access
render
root
rtkit
sambashare
saned
sasl
scanner
sgx
shadow
src
_ssh
ssl-cert
staff
stevee
sudo
sys
syslog
systemd-coredump
systemd-journal
systemd-network
systemd-resolve
systemd-timesync
tape
```

```
tcpdump
tss
tty
users
utmp
uucp
uuidd
video
voice
www-data
```

Powerful stuff! And so quick! You can strip data from all sorts of text files like CSV - comma separated values files - or tab delimited database files etc. This is difficult to do in a word processor as it requires more copy and cut actions to isolate columns from lines.

Day 3: File and Directory Permissions

What is a directory or file?

Unix/Linux sees "everything" in its system as a file of some type.

Both folders and files are reserved "areas" in a "file system" – in memory or a hard drive, for example - that has an amount of available "space" allocated for its use, that may or may not contain data (an empty directory or file).

A file has to have a directory to define it. Why?

The create/delete/rename and permissions data is stored in the parent directory, because that is where the name to data space mapping is stored.

This means the owner of a directory containing another users file may be able to delete it, even though they don't own the file or can't write to the file, depending on how the permissions are set on either the parent directory or the file.

So, file protection has to be considered by the write bits on the container and who or what User/Group/Others (UGO) can access the directory - BEWARE 777 permissions on a directory or file!
Anyone may be able to Read, Write (so delete) or eXecute (run) the contents!!

First, what does all this UGO, RWX and decimal 777 mean?

To understand these concepts, its necessary to understand what defines anything in terms of 1s or 0s in a computer system, which is based on the binary mathematical system.

Binary values are two relative voltages set as the electrical states of billions of transistors inside the CPU, memory, hard drive or other electrical hardware that comprises a computer system, usually 0V and 5V, but may be other values in lower power, more efficient systems e.g. 0V and 3.6V, positive or negative etc.

These 1s and 0s have to be counted and ordered according to “rules” or “protocols” to have any meaning, by programmes designed to group them in specific ways (a protocol) so they make sense when read by components and software in any system according to design need, instead of being just an arbitrary stream of 1s and 0s stored somewhere or streamed through a connection.

Smaller groups of bits are counted then combined together (according to a specific rule set or protocol) to form larger “structures” of different names, according to different protocols (bytes, packets, datagrams, frames, containers etc.), in a process called “encapsulation”.
You can read a Post I wrote called “From Bits to Gigabytes in 200 Years”, explaining this in more detail in my eBook “All My IT Knowledge Posts”.

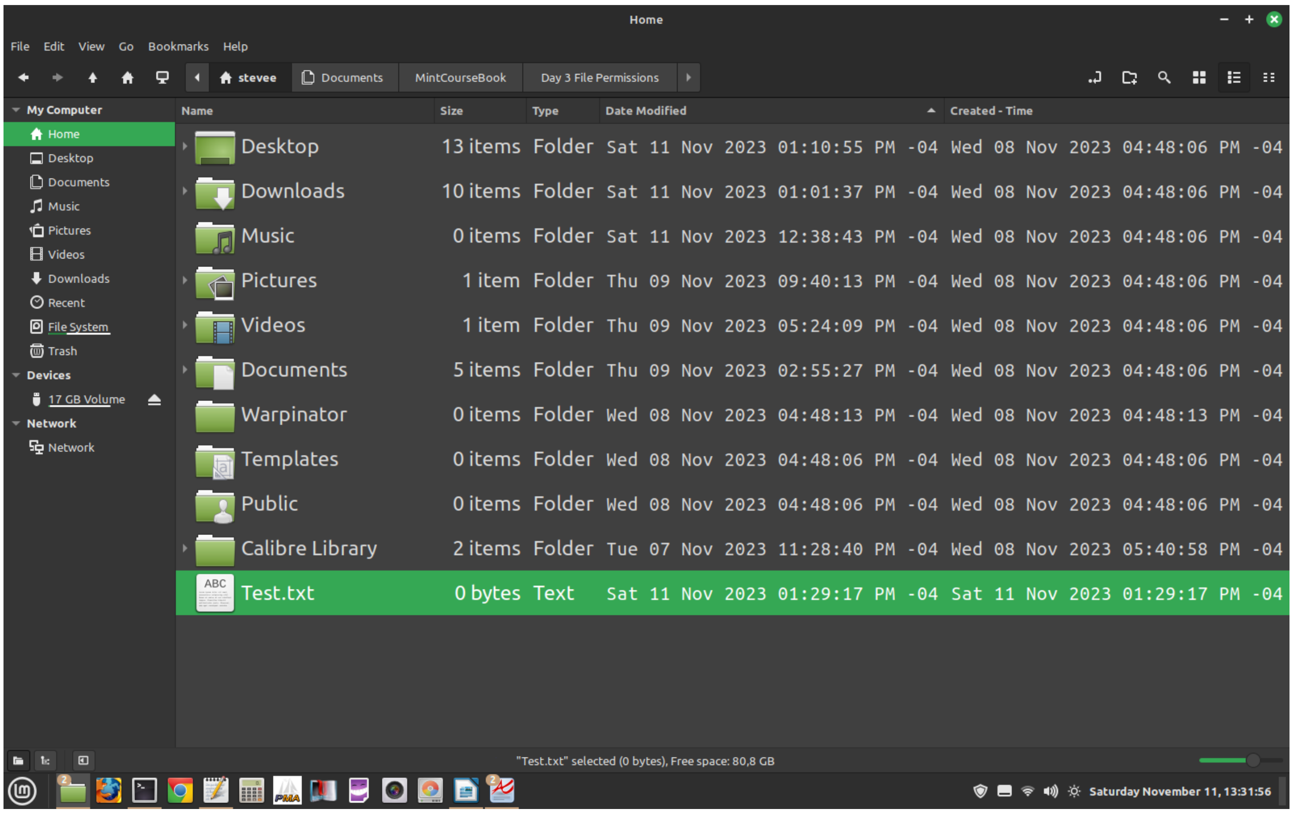
A practical Linux example helps to explain how protocol defined, ordered bits and bytes can represent different human language letters, so are a foundational concept in human/machine communication, enabling more complex yet easier to read programming languages to have been developed since the early days of computing, when programs were written in binary (base 2), octal (base 8) or hexadecimal (base 16) mathematics using punch cards! Tedious and time consuming!

Open a Terminal and create an empty file in your Home directory:

```
stevee@laptop:~$ cd
```

```
stevee@laptop:~$ touch Test.txt
```

In the GUI, you will see the file listed with its size (0) in bytes:



Name	Size	Type	Date Modified	Created - Time
Desktop	13 items	Folder	Sat 11 Nov 2023 01:10:55 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Downloads	10 items	Folder	Sat 11 Nov 2023 01:01:37 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Music	0 items	Folder	Sat 11 Nov 2023 12:38:43 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Pictures	1 item	Folder	Thu 09 Nov 2023 09:40:13 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Videos	1 item	Folder	Thu 09 Nov 2023 05:24:09 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Documents	5 items	Folder	Thu 09 Nov 2023 02:55:27 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Warpinator	0 items	Folder	Wed 08 Nov 2023 04:48:13 PM -04	Wed 08 Nov 2023 04:48:13 PM -04
Templates	0 items	Folder	Wed 08 Nov 2023 04:48:06 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Public	0 items	Folder	Wed 08 Nov 2023 04:48:06 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Calibre Library	2 items	Folder	Tue 07 Nov 2023 11:28:40 PM -04	Wed 08 Nov 2023 05:40:58 PM -04
Test.txt	0 bytes	Text	Sat 11 Nov 2023 01:29:17 PM -04	Sat 11 Nov 2023 01:29:17 PM -04

"Test.txt" selected (0 bytes), Free space: 80,8 GB

I can also list this in the Terminal to show info about the file:

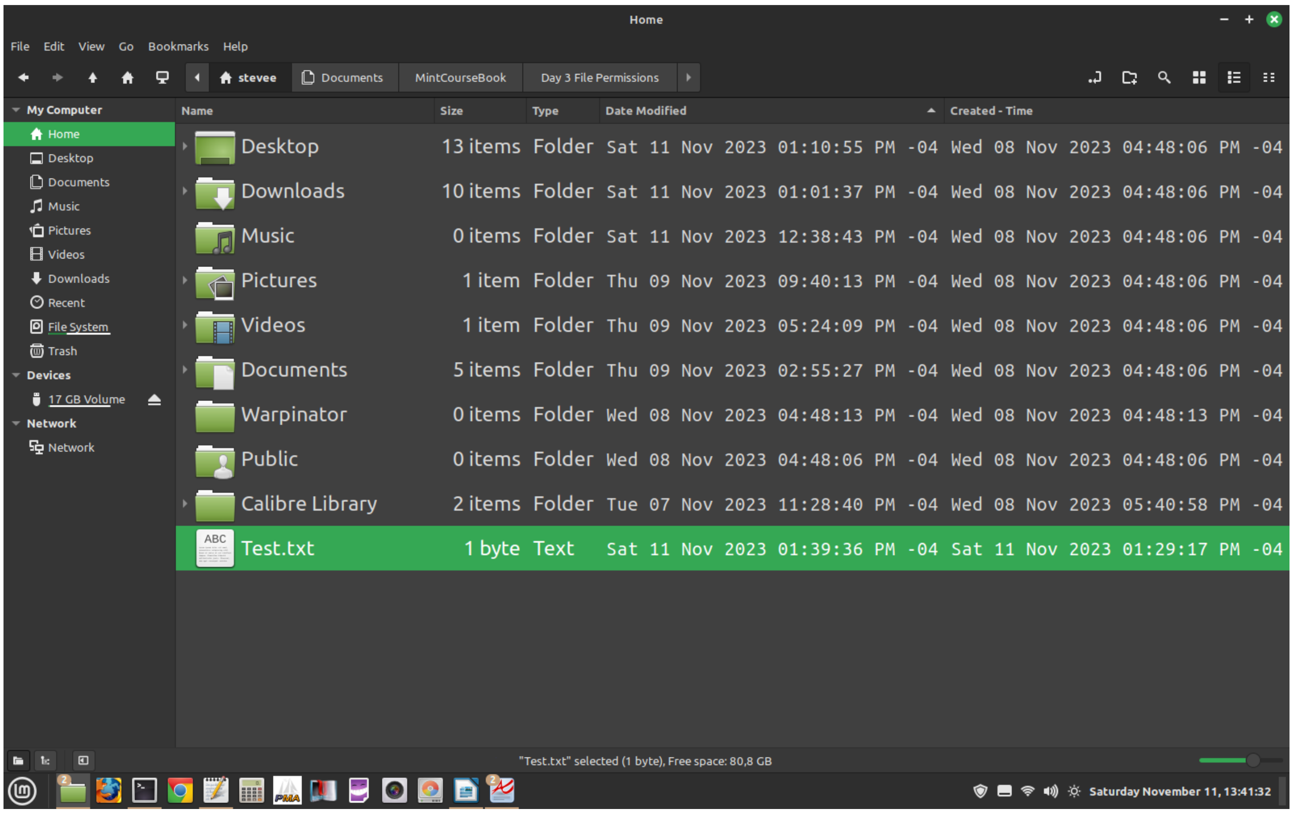
```
stevee@laptop:~$ ls -l Test.txt
-rw-rw-r-- 1 stevee stevee 0 Nov 11 13:29 Test.txt
```

The size on disk is also shown as 0 bytes in the 5th column, as covered on Day 1.

If I now add a letter to the file, without adding invisible "Control" characters, (which may occur using text editor), using the command line:

```
stevee@laptop:~$ echo -n A >> Test.txt
stevee@laptop:~$ ls -l Test.txt
-rw-rw-r-- 1 stevee stevee 1 Nov 11 13:39 Test.txt
```

I see the size on disk has increased to 1 byte, in the Terminal and the GUI:

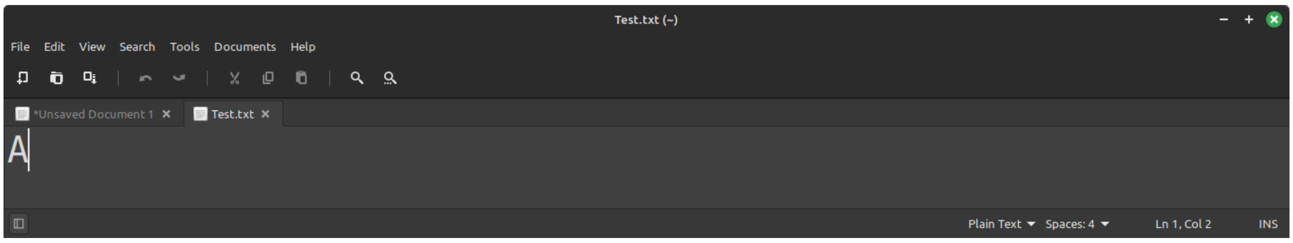


Name	Size	Type	Date Modified	Created - Time
Desktop	13 items	Folder	Sat 11 Nov 2023 01:10:55 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Downloads	10 items	Folder	Sat 11 Nov 2023 01:01:37 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Music	0 items	Folder	Sat 11 Nov 2023 12:38:43 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Pictures	1 item	Folder	Thu 09 Nov 2023 09:40:13 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Videos	1 item	Folder	Thu 09 Nov 2023 05:24:09 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Documents	5 items	Folder	Thu 09 Nov 2023 02:55:27 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Warpinator	0 items	Folder	Wed 08 Nov 2023 04:48:13 PM -04	Wed 08 Nov 2023 04:48:13 PM -04
Public	0 items	Folder	Wed 08 Nov 2023 04:48:06 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Calibre Library	2 items	Folder	Tue 07 Nov 2023 11:28:40 PM -04	Wed 08 Nov 2023 05:40:58 PM -04
Test.txt	1 byte	Text	Sat 11 Nov 2023 01:39:36 PM -04	Sat 11 Nov 2023 01:29:17 PM -04

"Test.txt" selected (1 byte), Free space: 80,8 GB

Saturday November 11, 13:41:32

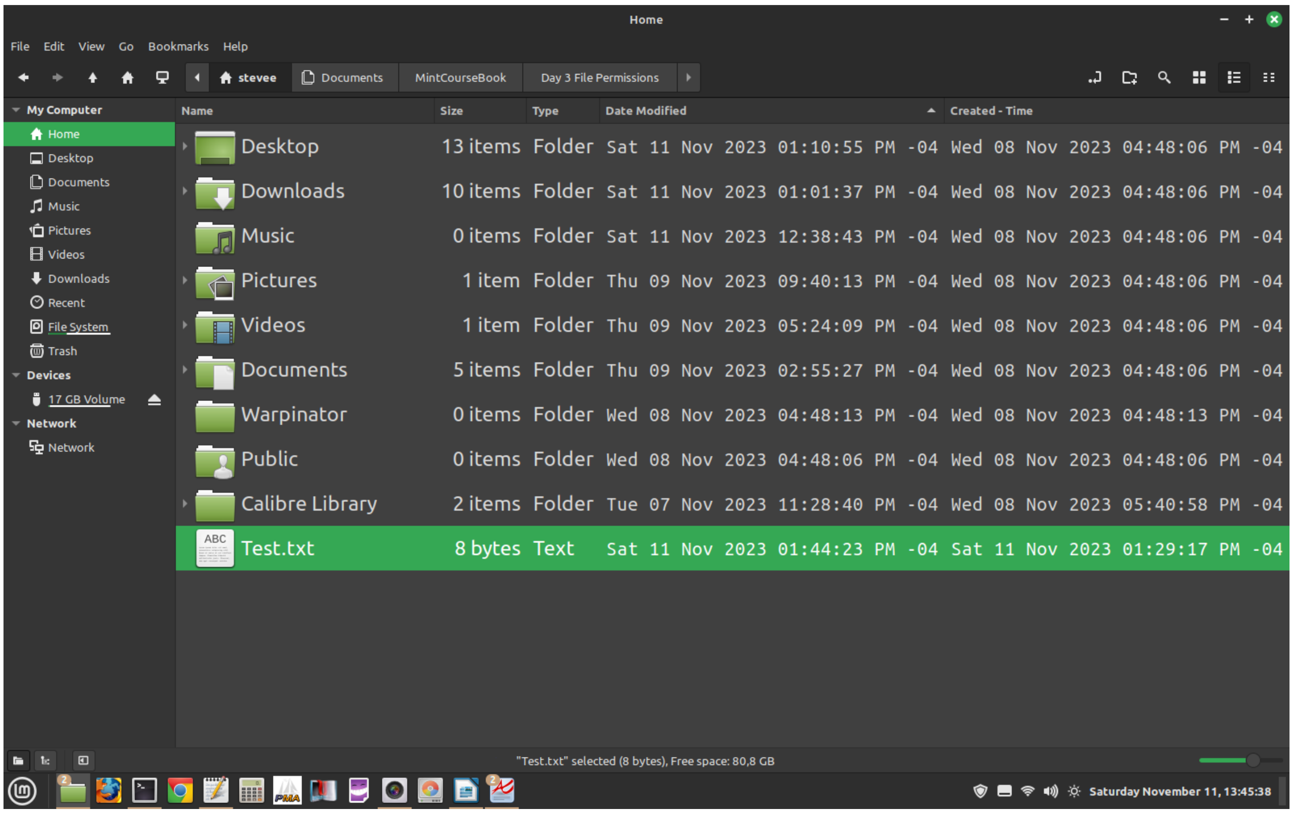
If I open the file with the Text editor I see the file contains the letter A:



```
A
```

To prove that a letter does actually occupy 1 byte of disk space, I add another 7 letters:

```
stevee@laptop:~$ echo -n BCDEFGH >> Test.txt
stevee@laptop:~$ ls -l Test.txt
-rw-rw-r-- 1 stevee stevee 8 Nov 11 13:44 Test.txt
```

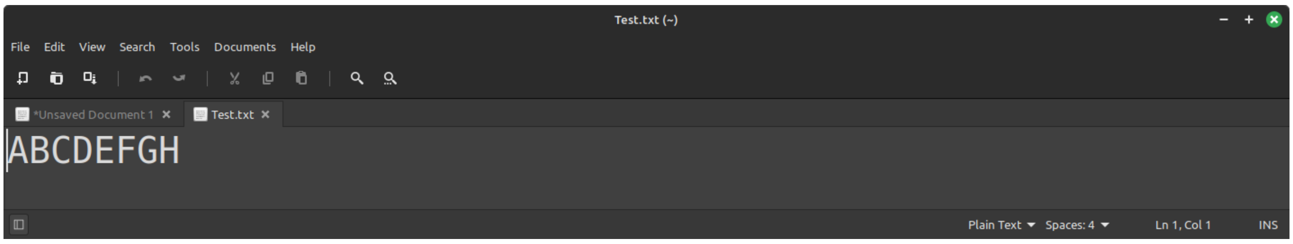


Name	Size	Type	Date Modified	Created - Time
Desktop	13 items	Folder	Sat 11 Nov 2023 01:10:55 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Downloads	10 items	Folder	Sat 11 Nov 2023 01:01:37 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Music	0 items	Folder	Sat 11 Nov 2023 12:38:43 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Pictures	1 item	Folder	Thu 09 Nov 2023 09:40:13 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Videos	1 item	Folder	Thu 09 Nov 2023 05:24:09 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Documents	5 items	Folder	Thu 09 Nov 2023 02:55:27 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Warpinator	0 items	Folder	Wed 08 Nov 2023 04:48:13 PM -04	Wed 08 Nov 2023 04:48:13 PM -04
Public	0 items	Folder	Wed 08 Nov 2023 04:48:06 PM -04	Wed 08 Nov 2023 04:48:06 PM -04
Calibre Library	2 items	Folder	Tue 07 Nov 2023 11:28:40 PM -04	Wed 08 Nov 2023 05:40:58 PM -04
Test.txt	8 bytes	Text	Sat 11 Nov 2023 01:44:23 PM -04	Sat 11 Nov 2023 01:29:17 PM -04

"Test.txt" selected (8 bytes), Free space: 80,8 GB

The GUI has updated to reflect the size change also – 1 byte per character!

The text editor, on reload shows the new contents:



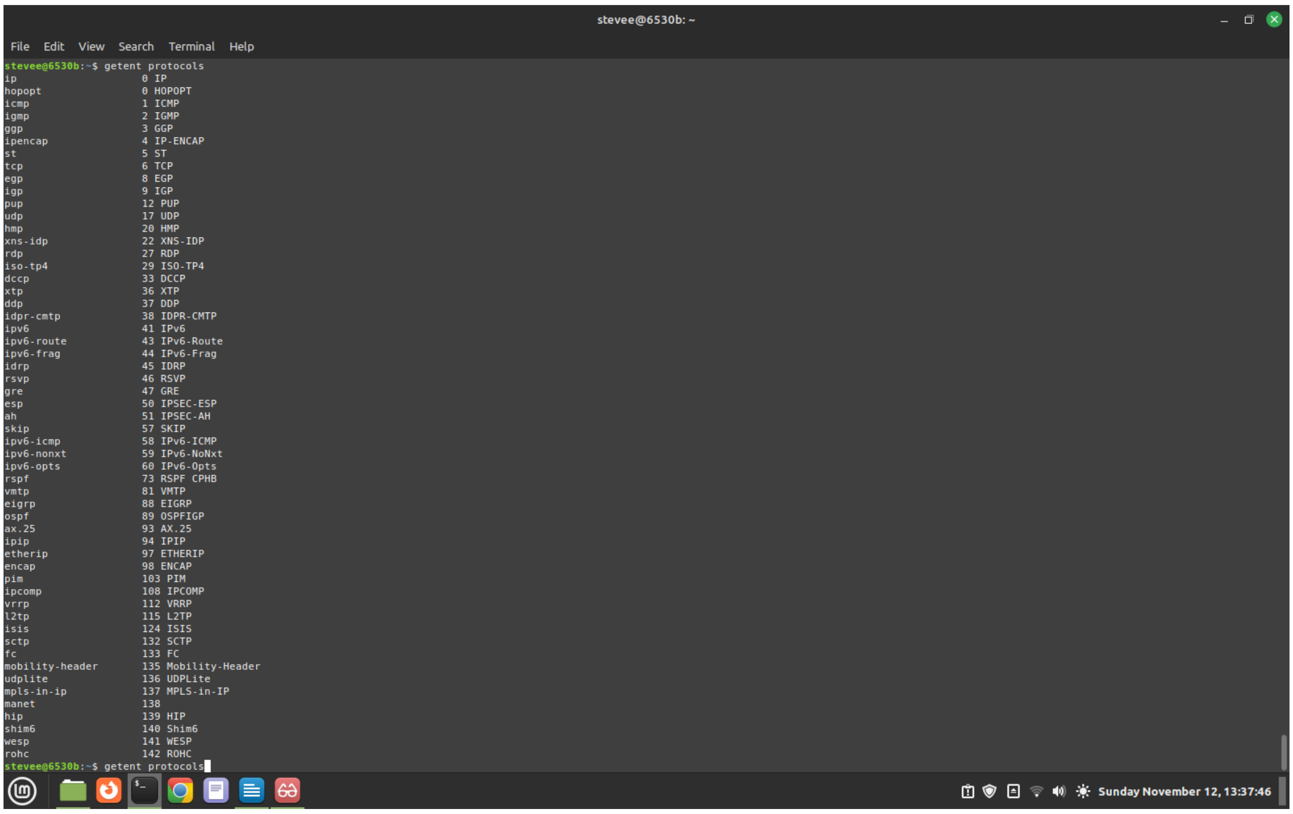
```
ABCDEFGH
```

This shows that each character comprises 8 bits (1 byte) of data, which can represent a particular letter, depending on how the software is designed to read those bits in that alphabetical form, according to a specific protocol, so it is readable by humans.

To aid further understanding of computer text Standards, google William Shott's excellent free Linux guide, The Linux Command Line, TLCL.PDF.

To get an idea of just some protocols that are listed in a Linux system, to enable all sorts of different networking operations:

```
stevee@laptop:~$ getent protocols
```


stevee@6530b: ~
File Edit View Search Terminal Help
Sunday November 12, 13:37:46

```
stevee@6530b:~$ getent protocols
ip                    0 IP
hopopt                0 HOPOPT
icmp                  1 ICMP
igmp                  2 IGMP
ggp                   3 GGP
ipencap               4 IP-ENCAP
st                    5 ST
tcp                   6 TCP
egp                   8 EGP
igp                   9 IGP
pup                   12 PUP
udp                   17 UDP
hmp                   20 HMP
xns-idp               22 XNS-IDP
rdp                   27 RDP
iso-tp4               29 ISO-TP4
dccp                  33 DCCP
xtp                   36 XTP
ddp                   37 DDP
idpr-cmtp             38 IDPR-CMTP
ipv6                  41 IPv6
ipv6-route            43 IPv6-Route
ipv6-frag             44 IPv6-Frag
idrp                  45 IDRP
rsvp                  46 RSVP
gre                   47 GRE
esp                   50 IPSEC-ESP
ah                    51 IPSEC-AH
skip                  57 SKIP
ipv6-icmp             58 IPv6-ICMP
ipv6-nonxt            59 IPv6-NoNxt
ipv6-opts             60 IPv6-Opts
rspf                  73 RSPF CPHB
vmtp                  81 VMTP
eigrp                 88 EIGRP
ospf                  89 OSPFIGP
ax.25                 93 AX.25
ipip                  94 IPIP
etherip               97 ETHERIP
encap                 98 ENCAP
pim                   103 PIM
ipcomp                108 IPCOMP
vrrp                  112 VRRP
l2tp                  115 L2TP
isis                  124 ISIS
sctp                  132 SCTP
fc                    133 FC
mobility-header       135 Mobility-Header
udplite               136 UDPLite
mpls-in-ip            137 MPLS-in-IP
manet                 138
hip                   139 HIP
shim6                 140 Shim6
wesp                  141 WESP
rohc                  142 ROHC
stevee@6530b:~$ getent protocols
```

Similarly, ALL data that makes up an OS filesystem is designed and stored in such ways to be manipulated by particular software programs to be of use to humans, by that data having particular attributes set against it, so it is useful, especially for security and access.

A database is a particularly good example of "ordered" sets of data that represent an analogy of something that exists in the real world, the traits of a human for example – name, age, address etc. Read my Database research paper for more information in my other eBook, "All My IT Knowledge Posts".

To prevent unwarranted damage or manipulation of file data, security mechanisms were developed to protect the state of the 1s and 0s that make up a particular data structure, which in Linux (and other) filesystems are called "permissions" that are set against folders and files using 3 blocks of 3 bytes that allow access (or not) to a particular file or folder according to which combinations of Users, Groups or Others (UGO) access values are set on a particular folder or file.

This system is shown in the first column of file attributes, by the list command:

```
stevee@laptop:~$ ls -l Test.txt
-rw-rw-r-- 1 stevee stevee 8 Nov 11 13:44 Test.txt
```

The first character (-) denotes a file, but can be other letters such as a (d) for directory, or a (b) for a block device like a hard drive (remember that everything in a linux system is aliased as a file, even hardware):

```
stevee@laptop:~$ ls -l /dev/sda
brw-rw---- 1 root disk 8, 0 Nov 11 11:34 /dev/sda
```

The next 9 characters are the 3 blocks of 3 RWX,RWX,RWX blocks that relate to the Users, Groups or Others (UGO) concept.

Each block is a decimal value of 000-777, where:

Read = 4
Write = 2
Exe = 1

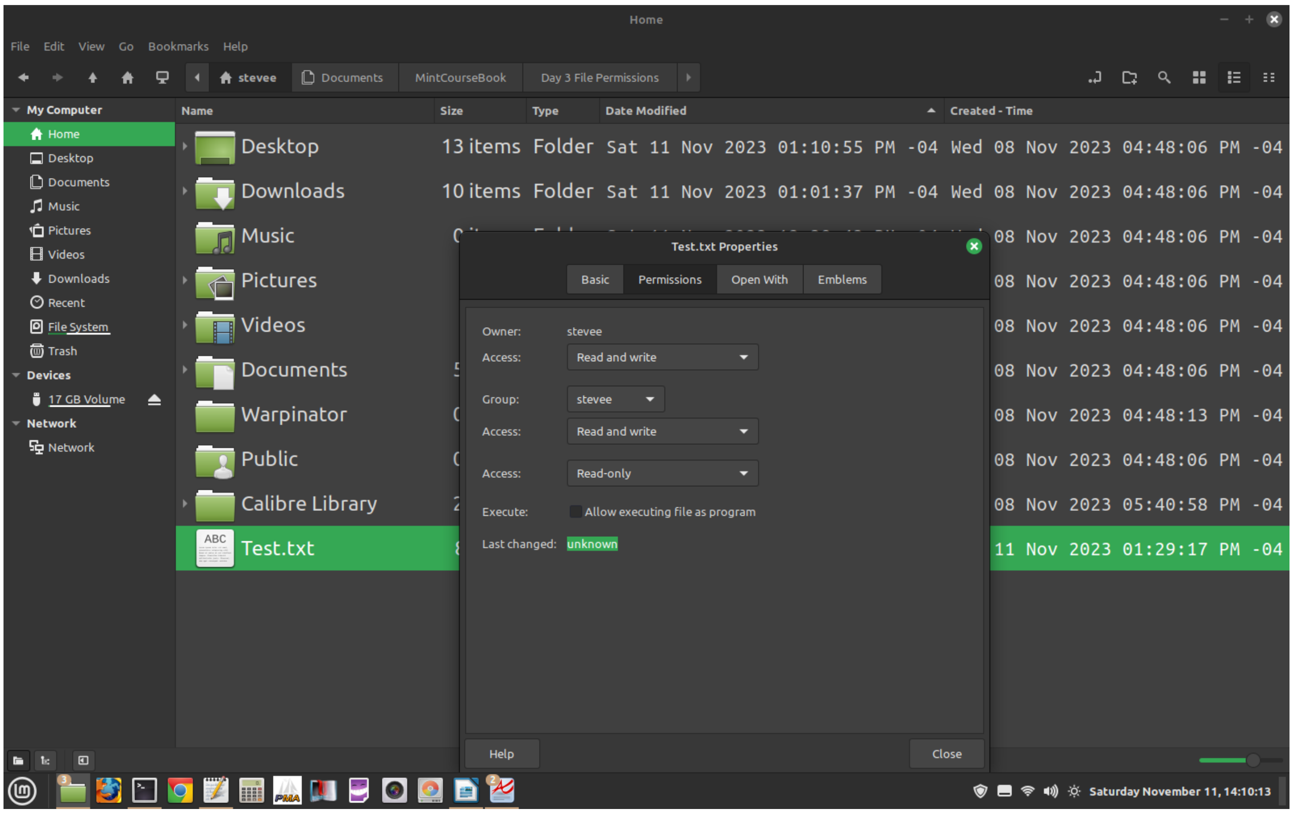
All add up to 7 max (i.e. octal, 0-7), for each UGO block. e.g.

r+w = 6 (4+2) = (110 binary)
r+x = 5 (4+1) = (101 binary)
r+w+x = 7 (4+2+1) = (111 binary)
--- = 0 (0+0+0) = (000 binary)

So, what's the practicality of this system?

It means that a Sysadmin has at least 3 ways to manipulate the permissions, so the access to, a particular folder or file – using the GUI, or the two similar Terminal methods:

To change file permissions in the GUI, you can right click a files Properties and use the Permissions tab:



```
Name            Size      Type    Date Modified                        Created - Time
Desktop         13 items  Folder  Sat 11 Nov 2023 01:10:55 PM -04  Wed 08 Nov 2023 04:48:06 PM -04
Downloads       10 items  Folder  Sat 11 Nov 2023 01:01:37 PM -04  Wed 08 Nov 2023 04:48:06 PM -04
Music                                                                   08 Nov 2023 04:48:06 PM -04
Pictures                                                                08 Nov 2023 04:48:06 PM -04
Videos                                                                  08 Nov 2023 04:48:06 PM -04
Documents                                                               08 Nov 2023 04:48:06 PM -04
Warpinator                                                              08 Nov 2023 04:48:13 PM -04
Public                                                                  08 Nov 2023 04:48:06 PM -04
Calibre Library                                                         08 Nov 2023 05:40:58 PM -04
Test.txt                                                                11 Nov 2023 01:29:17 PM -04
```

Test.txt Properties

Basic | Permissions | Open With | Emblems

Owner: steevee
Access: Read and write
Group: steevee
Access: Read and write
Access: Read-only
Execute: Allow executing file as program
Last changed: unknown

Help | Close

For those trying to use Windows WSL linux, the Windows OS permissions override the linux permissions, so the section below won't work without modifying /etc/wsl.conf, by adding:

```
[automount]
enabled  = true
root    = /mnt/
options  = "metadata,umask=22,fmask=11"
```

A reboot may be required, but try closing and opening WSL first.

To use the Terminal chmod (change mode) command, for example, to make a file eXecutable (so it could run as a program):

```
stevee@laptop:~$ chmod +x Test.txt
stevee@laptop:~$ ls -l Test.txt
-rwxrwxr-x 1 stevee stevee 8 Nov 11 13:44 Test.txt
```

or:

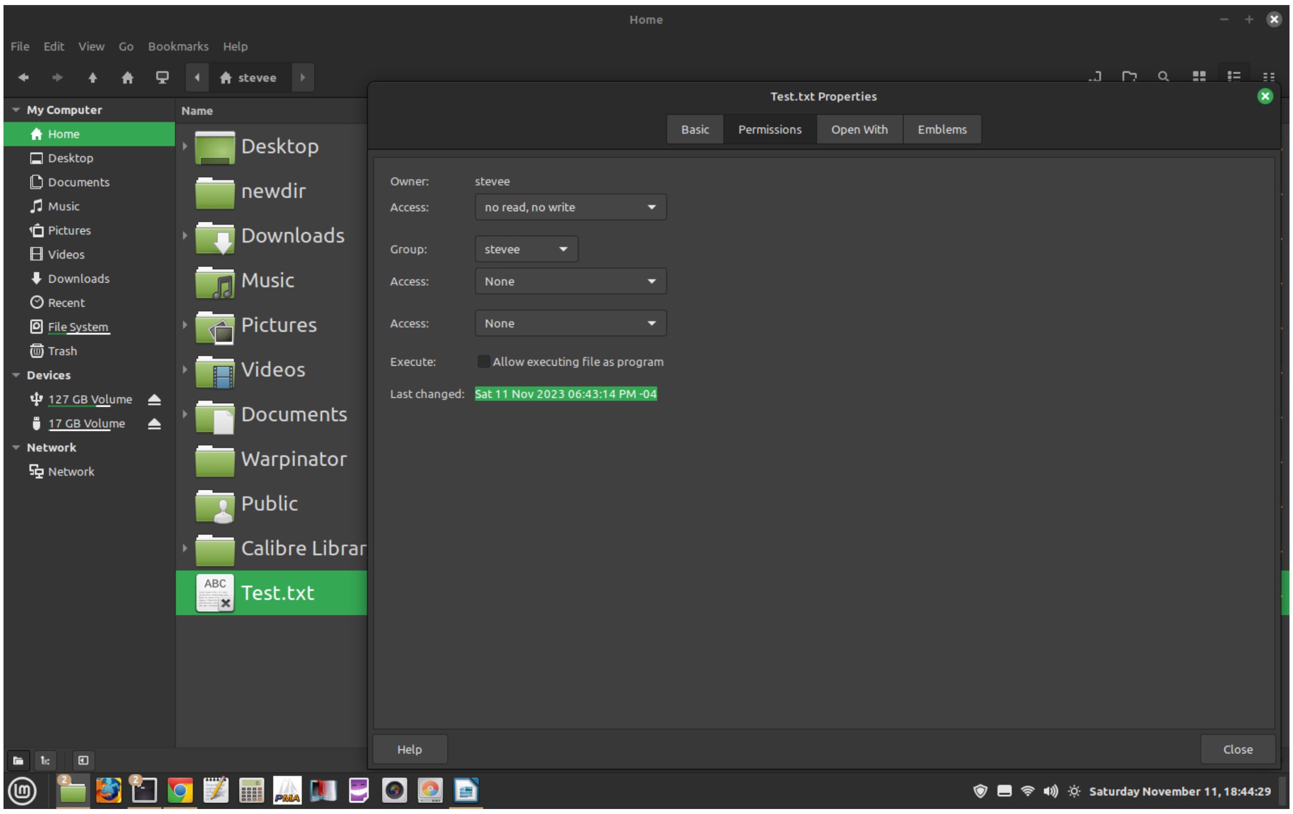
```
stevee@laptop:~$ chmod 777 Test.txt
stevee@laptop:~$ ls -l Test.txt
-rwxrwxrwx 1 stevee stevee 8 Nov 11 13:44 Test.txt
```

In each case, the file has been made executable for all members of all Users/Groups/Others

This is a maximum access, minimum security setting for a file! Hence beware 777 permissions, as anyone with system access could delete, modify or execute the file!

Conversely, a file with 000 permissions is totally inaccessible by anyone in any way (except root of course to modify it again), so practically useless, except for a short-term security protection for some reason?

```
stevee@laptop:~$ chmod 000 Test.txt
stevee@laptop:~$ ls -l Test.txt
---------- 1 stevee stevee 8 Nov 11 13:44 Test.txt
```



Owner: stevee
Access: no read, no write
Group: stevee
Access: None
Access: None
Execute: Allow executing file as program
Last changed: Sat 11 Nov 2023 06:43:14 PM -04

Note the black X for inaccessible for the Test.txt file. Changing it back to defaults for standard access:

```
stevee@laptop:~$ chmod 664 Test.txt
```

How do you read a files permissions in terms of the decimals?

```
stevee@laptop:~$ stat -c '%a' Test.txt
664
```

The default "create" permissions for folders and files in Linux represent a balance between access and security for the owners (creators) of a folder or file, as well as file system traversal by Group users and Others – 775 for folders, and 664 for files:

```
stevee@laptop:~$ mkdir newdir
stevee@laptop:~$ stat -c '%a' newdir/
775

stevee@laptop:~$ touch newfile.txt
stevee@laptop:~$ stat -c '%a' newfile.txt
664
```

Folder traversal is made possible by the default creation value of **775:**

5 (4+1 = Read + Exe), as it´s the 5 (x) value that allows traversal through a folder nest by Others. The 5 is trumped by the 7s for Users/Groups, so traversal (x=1) as well as Read and Write permissions is included in the value of 7 for Users/Groups. (Read = 4 + Write = 2 + eXe =1, total =7), so User/Group/Others traversal is a default condition of any newly created user directory.

However! As you will see in the Shared Group example below, both the root (/) directory and all users Home directories are restricted to root user access and the respective Home directory users access by default at the OS install and for new user account creation. This means a Standard user cannot roam freely anywhere they like, for obvious security and other user account privacy reasons.

It pays to experiment with the chmod command on a test file to see the effects different values (000-777) for different Users/Groups/Others makes, so what access different users and groups have, remembering the ownership aspects on a file and the files parent directory.

Shared Group Directory example:

A practical but hypothetical example may be user mike being a Dept. Head who creates a shared project folder for other users to collaborate on files with.

As the Project Manager and directory creator/owner he may have total responsibility for the shared folder contents access and security, and possibly the users file content also, depending on requirements.

It may be better to create a project group first to add other users to, including himself, for cleaner system admin. To do this as sudo user stevee, I need to become user mike:

```
stevee@laptop:~$ sudo su mike
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

mike@laptop:/home/stevee$ sudo addgroup mikeproject1
[sudo] password for mike:
Adding group `mikeproject1' (GID 1002) ...
Done.
```

Now let’s look at the initial member status of group mikeproject1, using grep to find the string mike in any form:

```
mike@laptop:/home/stevee$ cat /etc/group | grep mike
sudo:x:27:stevee,mike
mike:x:1001:
mikeproject1:x:1002:
```

The group mikeproject1 has no users at present, unlike group sudo.

Mike wants to add user stevee and himself to his new group:

```
mike@laptop:/home/stevee$ sudo gpasswd -M stevee,mike mikeproject1
mike@laptop:/home/stevee$ cat /etc/group | grep mike
sudo:x:27:stevee,mike
mike:x:1001:
mikeproject1:x:1002:stevee,mike
```

The comma separated list, stevee,mike, adds those users to group mikeproject1, but checking to cd into his own Home directory first, not stevee's!

Now mike can create a directory to house shared access files:

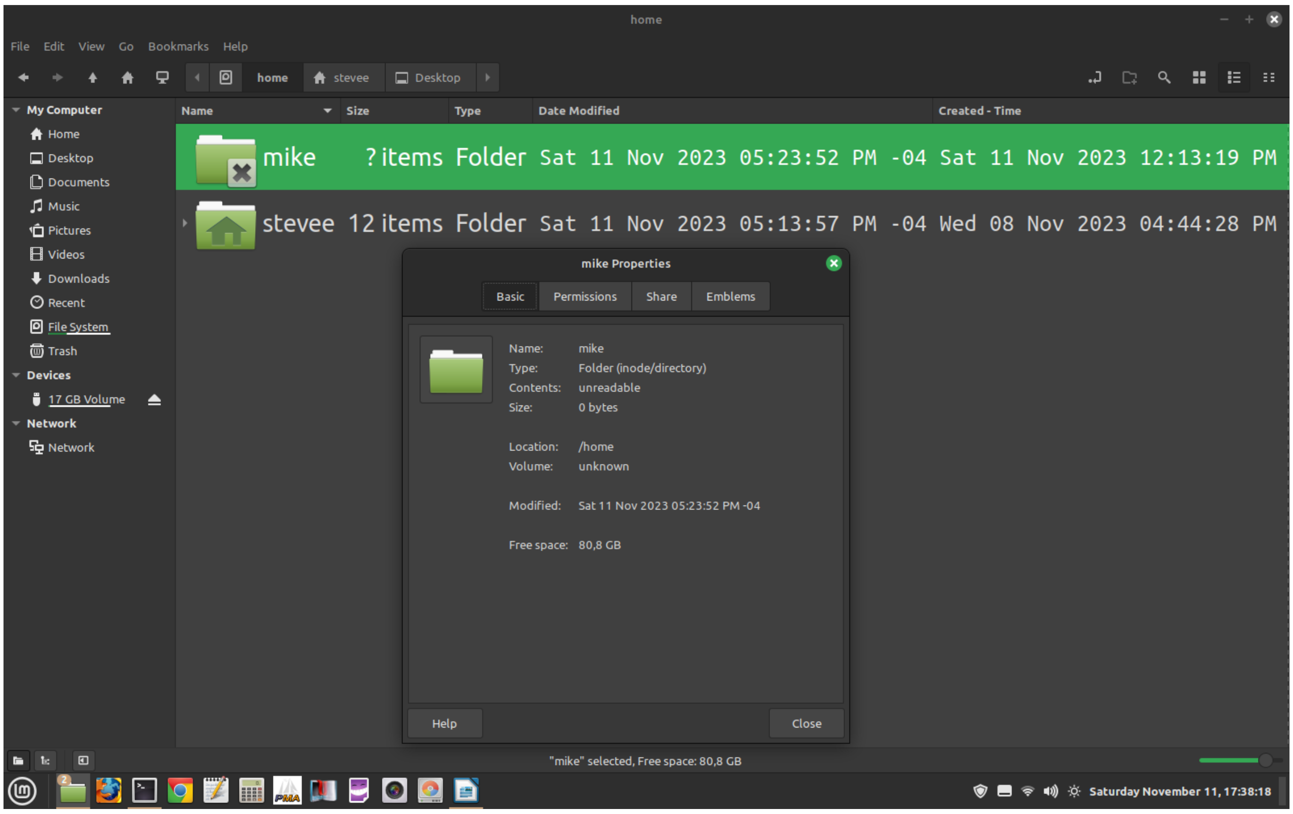
```
mike@laptop:/home/stevee$ cd
mike@laptop:~$ mkdir mikeprojects

mike@laptop:~$ ls -ld /home/mike/mikeprojects/
drwxrwxr-x 2 mike mike 4096 Nov 11 17:23 /home/mike/mikeprojects/
```

You can see the default directory permissions are drwxrwxr-x = 775 but the folder belongs to user and group mike not to the mikeproject1 group.

This can be changed with the change owner command, sudo chown:

```
 mike@laptop:~$ sudo chown mike:mikeproject1 mikeprojects/
mike@laptop:~$ ls -ld /home/mike/mikeprojects/
drwxrwxr-x 2 mike mikeproject1 4096 Nov 11 17:23 /home/mike/mikeprojects/
```

The group owner is now shown in the 4th column.
The problem is that access to mike's Home directory, where the project folder is housed, is inaccessible to all but user mike!



```
Name      Size      Type    Date Modified                       Created - Time
mike      ? items   Folder  Sat 11 Nov 2023 05:23:52 PM -04  Sat 11 Nov 2023 12:13:19 PM
stevee    12 items  Folder  Sat 11 Nov 2023 05:13:57 PM -04  Wed 08 Nov 2023 04:44:28 PM
```

mike Properties

```
Name:        mike
Type:        Folder (inode/directory)
Contents:    unreadable
Size:        0 bytes

Location:    /home
Volume:      unknown

Modified:    Sat 11 Nov 2023 05:23:52 PM -04

Free space:  80,8 GB
```

"mike" selected, Free space: 80,8 GB

Why are all users Home directories inaccessible except by the user/owner by default?

Its a combination of directory owner and the permissions set, for obvious user security reasons. You don’t want everyone else having access to your account from elsewhere on the system or over a network, if the PC is connected to one!:

```
stevee@laptop:~$ ls -l /home/
total 8
drwxr-x--- 14 mike   mike   4096 Nov 11 17:23 mike
drwxr-x--- 20 stevee stevee 4096 Nov 11 17:13 stevee
```

You can see that Users/Groups/Others is set to rwxr-x---, which in principle allows directory traversal for user mike and group mike, by the Home directory creation default value 750, but ONLY to the directory owner. So how to allow traversal access to the group mikeproject1?

```
mike@laptop:~$ stat -c '%a' .
750
```

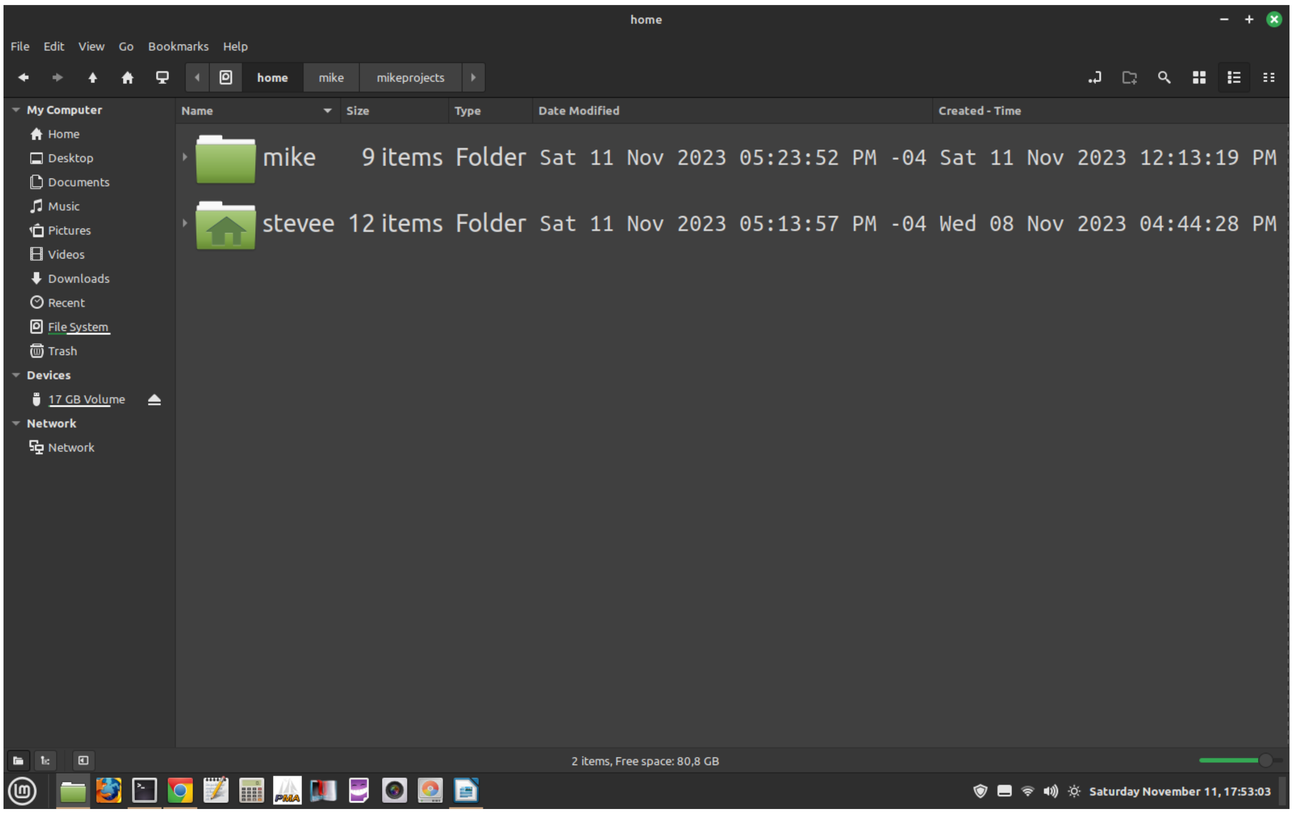
Mike can use the chown command on his Home directory for part ownership by the mikeproject1 Group:

```
mike@laptop:~$ chown mike:mikeproject1 /home/mike
```

```
mike@laptop:~$ ls -l /home/
total 8
```

```
drwxr-x--- 14 mike   mikeproject1 4096 Nov 11 17:23 mike
drwxr-x--- 20 stevee stevee       4096 Nov 11 17:13 stevee
```

Now mike’s Home folder is accessible to stevee, the X has gone:

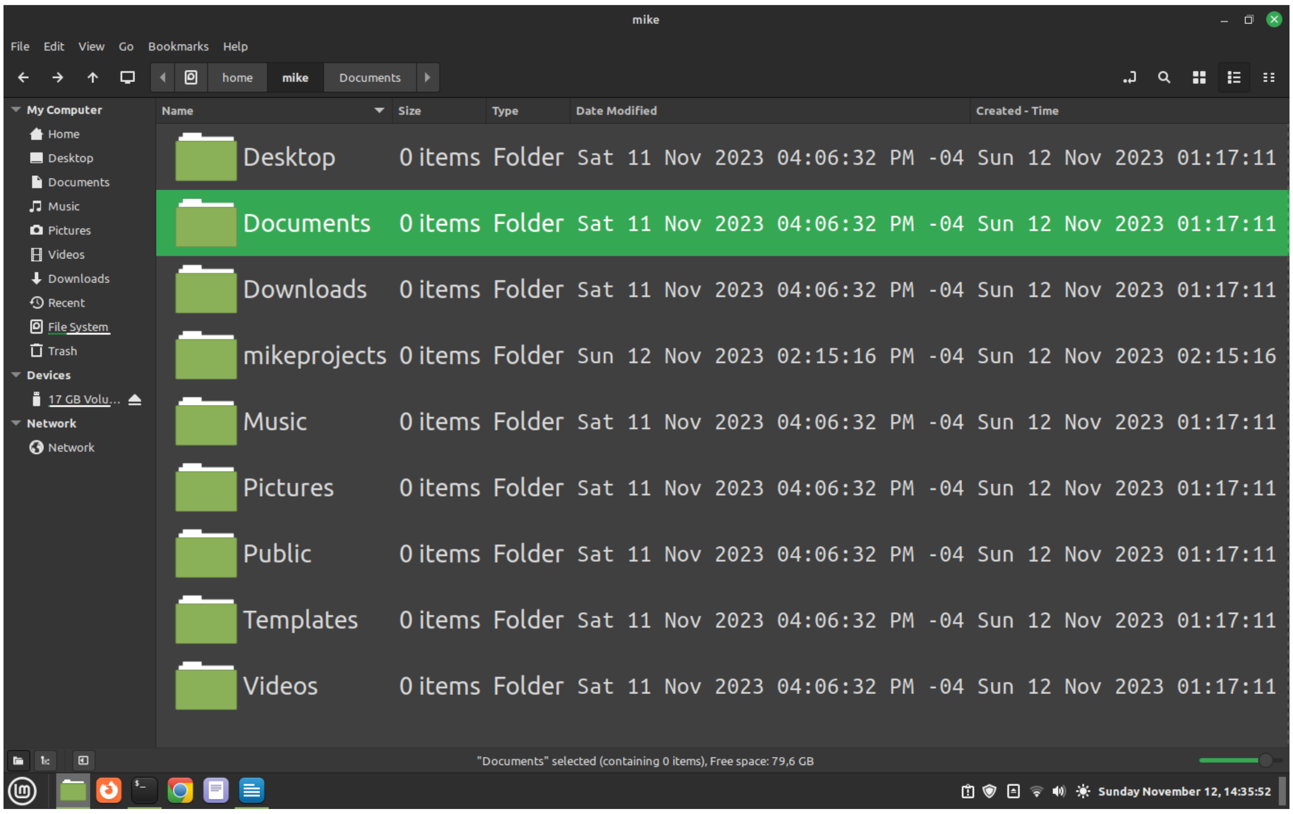


```
Name     Size      Type    Date Modified                         Created - Time
mike     9 items   Folder  Sat 11 Nov 2023 05:23:52 PM -04  Sat 11 Nov 2023 12:13:19 PM
stevee   12 items  Folder  Sat 11 Nov 2023 05:13:57 PM -04  Wed 08 Nov 2023 04:44:28 PM
```

2 items, Free space: 80,8 GB

Saturday November 11, 17:53:03

However, this is NOT ideal, as now ALL mike’s sub folders are also accessible to all other members of the mikeproject1 Group due to the inheritance of the parent Home directory permissions. This is not desirable as it´s insecure for mike’s folders, as the content is viewable to all users in the mikeproject1 group!

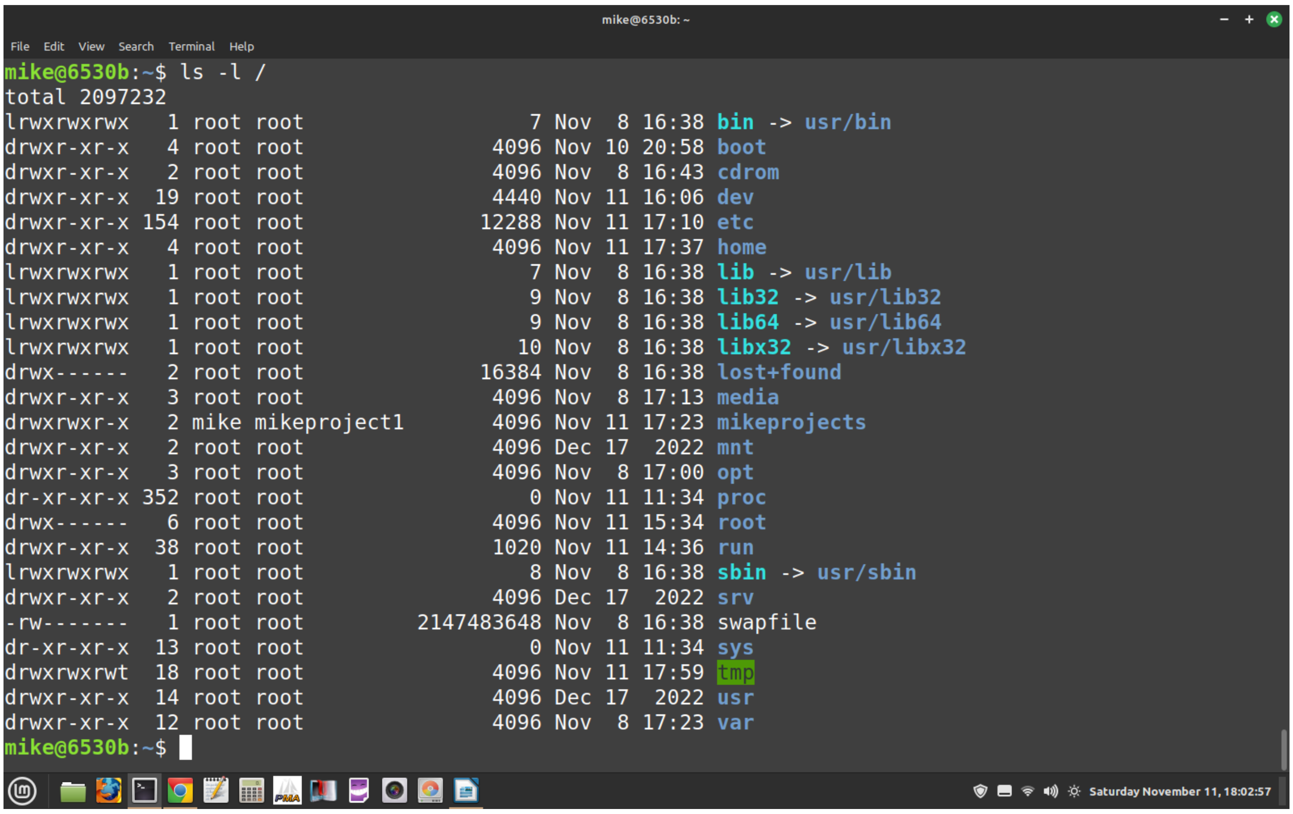


Name	Size	Type	Date Modified	Created - Time
Desktop	0 items	Folder	Sat 11 Nov 2023 04:06:32 PM -04	Sun 12 Nov 2023 01:17:11
Documents	0 items	Folder	Sat 11 Nov 2023 04:06:32 PM -04	Sun 12 Nov 2023 01:17:11
Downloads	0 items	Folder	Sat 11 Nov 2023 04:06:32 PM -04	Sun 12 Nov 2023 01:17:11
mikeprojects	0 items	Folder	Sun 12 Nov 2023 02:15:16 PM -04	Sun 12 Nov 2023 02:15:16
Music	0 items	Folder	Sat 11 Nov 2023 04:06:32 PM -04	Sun 12 Nov 2023 01:17:11
Pictures	0 items	Folder	Sat 11 Nov 2023 04:06:32 PM -04	Sun 12 Nov 2023 01:17:11
Public	0 items	Folder	Sat 11 Nov 2023 04:06:32 PM -04	Sun 12 Nov 2023 01:17:11
Templates	0 items	Folder	Sat 11 Nov 2023 04:06:32 PM -04	Sun 12 Nov 2023 01:17:11
Videos	0 items	Folder	Sat 11 Nov 2023 04:06:32 PM -04	Sun 12 Nov 2023 01:17:11

"Documents" selected (containing 0 items), Free space: 79,6 GB

A better way would be to move the mikeprojects folder to the root directory to avoid messing up mike's Home directory privacy, then changing mike's Home directory permissions back to their original 750 values. Mike can move his folder to the root directory because he has sudo privileges after being made a member of the sudo group, remember?:

```
mike@laptop:~$ sudo mv -v /home/mike/mikeprojects/ /
[sudo] password for mike:
renamed '/home/mike/mikeprojects/' -> '/mikeprojects'

mike@laptop:~$ chmod 750 /home/mike/
mike@laptop:~$ ls -l /home/
total 8
drwxr-x--- 13 mike   mikeproject1 4096 Nov 11 18:00 mike
drwxr-x--- 20 stevee stevee       4096 Nov 11 17:13 stevee


mike@laptop:~$ ls /
bin   dev   lib    libx32      mikeprojects  proc  sbin      sys  var
boot  etc   lib32  lost+found  mnt           root  srv       tmp
cdrom home  lib64  media       opt           run   swapfile  usr
```


mike@6530b: ~
File Edit View Search Terminal Help
Saturday November 11, 18:02:57

```
mike@6530b:~$ ls -l /
total 2097232
lrwxrwxrwx   1 root root                7 Nov  8 16:38 bin -> usr/bin
drwxr-xr-x   4 root root             4096 Nov 10 20:58 boot
drwxr-xr-x   2 root root             4096 Nov  8 16:43 cdrom
drwxr-xr-x  19 root root             4440 Nov 11 16:06 dev
drwxr-xr-x 154 root root            12288 Nov 11 17:10 etc
drwxr-xr-x   4 root root             4096 Nov 11 17:37 home
lrwxrwxrwx   1 root root                7 Nov  8 16:38 lib -> usr/lib
lrwxrwxrwx   1 root root                9 Nov  8 16:38 lib32 -> usr/lib32
lrwxrwxrwx   1 root root                9 Nov  8 16:38 lib64 -> usr/lib64
lrwxrwxrwx   1 root root               10 Nov  8 16:38 libx32 -> usr/libx32
drwx------   2 root root            16384 Nov  8 16:38 lost+found
drwxr-xr-x   3 root root             4096 Nov  8 17:13 media
drwxrwxr-x   2 mike mikeproject1     4096 Nov 11 17:23 mikeprojects
drwxr-xr-x   2 root root             4096 Dec 17  2022 mnt
drwxr-xr-x   3 root root             4096 Nov  8 17:00 opt
dr-xr-xr-x 352 root root                0 Nov 11 11:34 proc
drwx------   6 root root             4096 Nov 11 15:34 root
drwxr-xr-x  38 root root             1020 Nov 11 14:36 run
lrwxrwxrwx   1 root root                8 Nov  8 16:38 sbin -> usr/sbin
drwxr-xr-x   2 root root             4096 Dec 17  2022 srv
-rw-------   1 root root       2147483648 Nov  8 16:38 swapfile
dr-xr-xr-x  13 root root                0 Nov 11 11:34 sys
drwxrwxrwt  18 root root             4096 Nov 11 17:59 tmp
drwxr-xr-x  14 root root             4096 Dec 17  2022 usr
drwxr-xr-x  12 root root             4096 Nov  8 17:23 var
mike@6530b:~$
```

Again, not totally ideal, as if everyone did this, the root directory would become cluttered with shared folders. Better to create one Shared folder on the / root directory with totally open, 777 permissions, for all users to create and then lockdown their personal, shared folders within that Shared folder.

I hope the example illustrates basic ideas about the many real-world requirements and combinations possible with the Linux permissions system, as well as some pitfalls to watch out for with the combination of permissions and ownerships?

So, does mike's root directory shared Project folder fulfil its purpose of the project Group members needs to access, create, amend, share and for mike to delete their content?

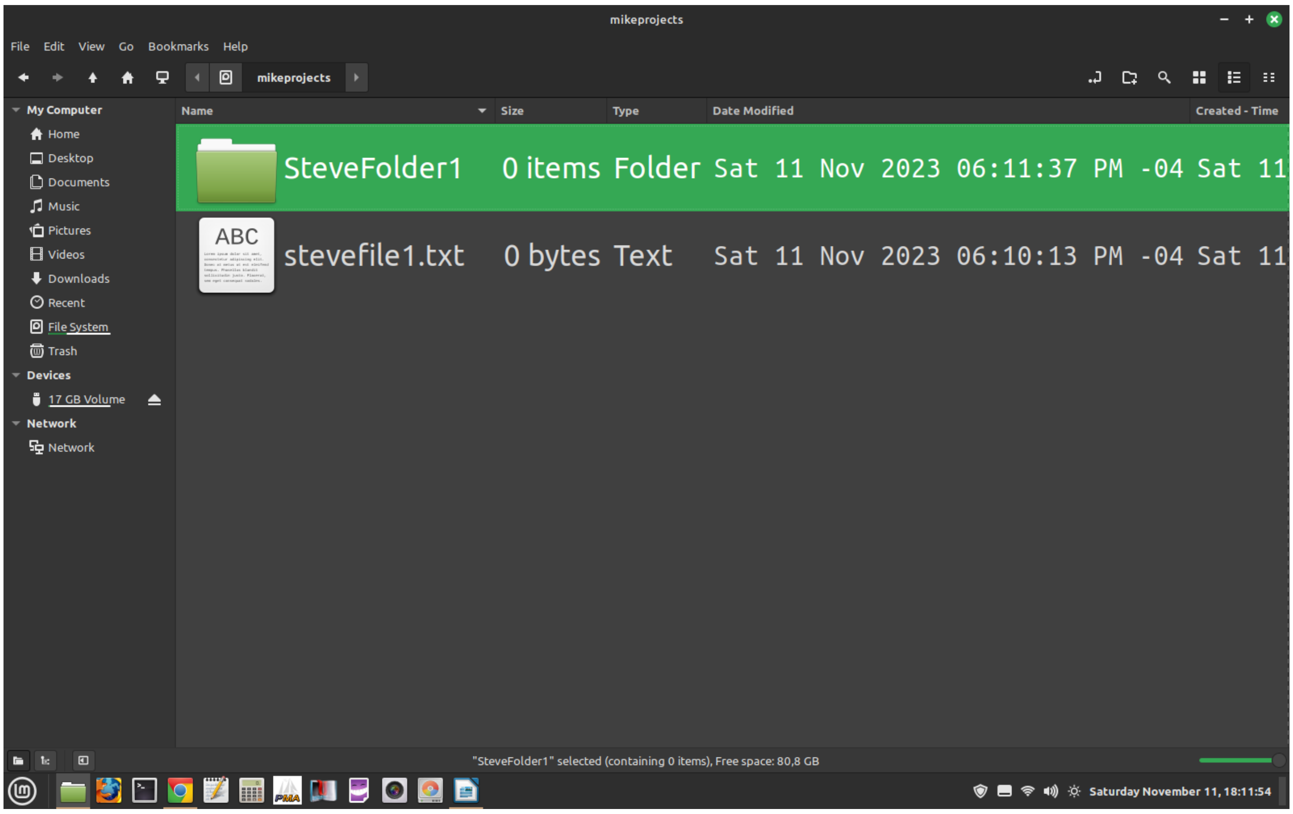
Yes, user stevee can create a folder and a file in the /mikeprojects folder after we exit out of mike's account to become stevee again:

mike@laptop:~$ exit

stevee@laptop: cd /mikeprojects

stevee@hplaptop:/mikeprojects$ mkdir SteveFolder1

stevee@hplaptop:/mikeprojects$ touch stevefile1.txt /mikeprojects/



```
SteveFolder1   0 items  Folder  Sat 11 Nov 2023 06:11:37 PM -04 Sat 11
stevefile1.txt 0 bytes  Text    Sat 11 Nov 2023 06:10:13 PM -04 Sat 11
```

"SteveFolder1" selected (containing 0 items), Free space: 80,8 GB

Saturday November 11, 18:11:54

Can user mike delete stevee's folder and files even though he does not own them? Yes! Due to the ownership permissions that he has on the /mikeprojects parent folder, even though he does not own stevee's file!

```
stevee@hplaptop:/mikeprojects$ su mike
Password:
mike@hplaptop:/mikeprojects$

mike@hplaptop:/mikeprojects$ rm -v /mikeprojects/stevefile1.txt
rm: remove write-protected regular empty file '/mikeprojects/stevefile1.txt'? y
removed '/mikeprojects/stevefile1.txt'

mike@laptop:~$ ls -l /mikeprojects/
total 4
drwxrwxr-x 2 stevee stevee 4096 Nov 11 18:11 SteveFolder1

mike@hplaptop:/mikeprojects$ rmdir SteveFolder1/
mike@hplaptop:/mikeprojects$ ls -l
total 0
```

If mike creates a file, can user stevee delete or amend it?

```
mike@laptop:~$ touch /mikeprojects/mikefile.txt
mike@laptop:~$ ls -l /mikeprojects/
```

```
total 4
-rw-rw-r-- 1 mike   mike      0 Nov 11 18:16 mikefile.txt
drwxrwxr-x 2 stevee stevee 4096 Nov 11 18:11 SteveFolder1
```

No! User stevee cannot delete mike's file because it is in mike's parent directory and stevee does not own the file!

```
stevee@laptop:~$ rm -v /mikeprojects/mikefile.txt
rm: remove write-protected regular empty file '/mikeprojects/mikefile.txt'? y
rm: cannot remove '/mikeprojects/mikefile.txt': Permission denied
```

This fulfils the original requirement of mike being the project manager with overall control of the contents of the shared Projects directory, but still allows project Group users to create and amend their own files in that folder that are readable by other group members, but they cannot delete or amend those files of other Group users. Only users mike and root can do that, unless sudo is invoked by other users with that sudo admin permission.

It's a combination of individual folder/file permissions and parent directory ownership that decides overall access to file system content.

There are other "special" file permissions beyond the scope of this text, for your research, such as the "sticky bit" and hidden/undeletable files properties that can be set with the chattr and read by lsattr commands. Read their man pages e.g:

```
man chattr
…
i     A  file  with the 'i' attribute cannot be modified: it cannot be
        deleted or renamed, no link can be created to this file, most of
        the file's metadata can not be modified, and the file can not be
        opened in write mode.  Only the superuser or a process  possess-
        ing the CAP_Linux_IMMUTABLE capability can set or clear this at-
        tribute.
```

Ever had a situation where you could not recursively remove some directories due to one or more files having a permission that even root could not take control of? Did you ever find out why?

I have had this scenario on a backup drive, with Windows files having an attribute set from being copied from a compressed Win file system, which would not then allow root to remove it when the external drive was attached to a Linux box.

Infuriating - as it meant I could not delete the whole, non-empty directory with this file under it.

I ended up re-attaching the drive to a Win PC, then taking ownership that way, before I could delete the file with the special attribute, and then returning the drive to a Linux system for full control, short of just formatting the drive completely in Linux and losing everything else of course.

In a similar vein for special file attributes, Linux can prevent file deletion to all users, including root, using the chattr command.

Day 4: File Permissions, Network Access via Samba

After sleeping on the complexity yet elegance of permissions, think of the ways and consequences of sharing files with other users using:

1 chmod 777
Not usually, only for traversal access. 777 = write for Others (Anyone can delete files or folders)

2 chmod 755
Common, sets security access for the folder/file user only

3 chmod 775
Common, sets security access for user and group access only, gives Others traversal access to sub folders

4 chmod 770
Common and secure, sets security access for user and group access only, with NO Others access or traversal at all

5 chmod 000
Rare. It renders a folder/file inaccessible. May be a temporary anti hacking measure for a compromised system, unless the attacker has root privileges anyway?

6 chmod 700
Useful. This can make a directory inaccessible to all but the user/owner, so private.

What else has to be set up re Users and Groups?
Users added to appropriately created or existing Groups

What other non-octal chmod method is available?
Variations on the form:
```
chmod ugo -rwx /filename
```

Are there local and network access differences to consider – e.g. local and network passwords - Samba ?
Yes, network access permissions on Linux are setup in a text file you can read in:

```
stevee@laptop:~$ vi /etc/samba/smb.conf
```

It is very complicated overall with a lot of parameters, but the most basic network access usually required is to give a user network access to their own Home directory on other networked PCs they have an account on by uncommenting the (;) semicolons in the section:

```
#======================= Share Definitions =======================

# Un-comment the following (and tweak the other settings below to suit)
# to enable the default home directory shares. This will share each
# user's home directory as \\server\username
```

```
;[homes]
;   comment = Home Directories
;   browseable = no

# By default, the home directories are exported read-only. Change the
# next parameter to 'no' if you want to be able to write to them.
;   read only = yes

# File creation mask is set to 0700 for security reasons. If you want to
# create files with group=rw permissions, set next parameter to 0775.
;   create mask = 0700

# Directory creation mask is set to 0700 for security reasons. If you want to
# create dirs. with group=rw permissions, set next parameter to 0775.
;   directory mask = 0700

# By default, \\server\username shares can be connected to by anyone
# with access to the samba server.
# Un-comment the following parameter to make sure that only "username"
# can connect to \\server\username
# This might need tweaking when using external authentication schemes
;   valid users = %S
```

Samba may need to be installed to allow access to other PCs on a network

```
stevee@laptop:~$ sudo apt install samba
```

This is beyond this documents scope to show as I am not on a network. More research for you if you want to experiment with networked PCs...

It may be necessary for a user to have a samba network password setup as well as their normal account login password to access another PC over the network, using:

```
stevee@laptop:~$ sudo smbpasswd -a stevee
New SMB password:
Retype new SMB password:
Added user stevee.
```

I mention it to explain a principle, that ultimately, the individual file permissions on a PC dictate the access level, so usually, any network access permissions are very loose in comparison to ensure the user can at least connect to the remote system and access a login box at least.

Final Thoughts

Correct security settings on any system are crucial, so file permission familiarity requires practise with testing to be sure you know what really happens. Mistakes are easy here!

Day 5: Package Installation, Programming Environments, Backups

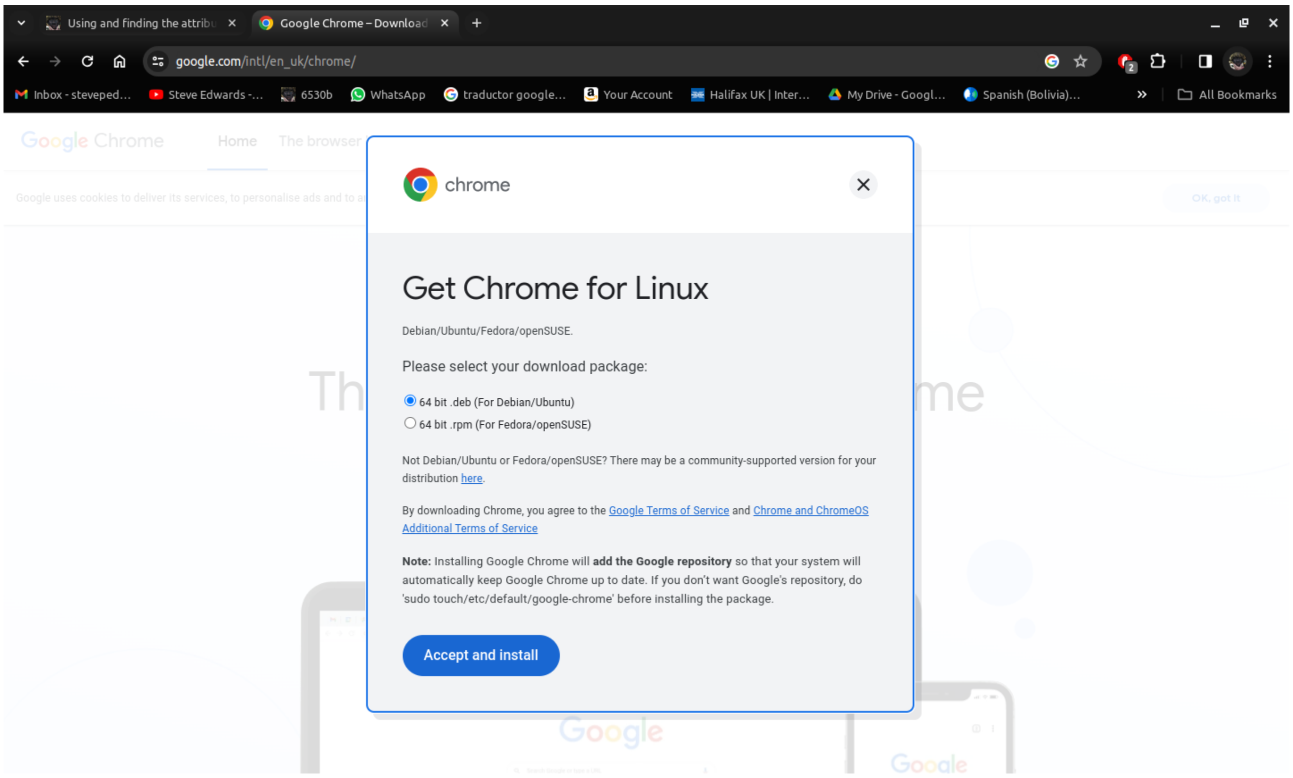
The Linux system and software packages are stored in “online repositories” and are regularly changed and updated.

To get an idea how many different packages are available for a given flavour of Linux – Mint, Ubuntu, Debian etc:

```
 stevee@laptop:~$ apt-cache pkgnames | wc -l
76992
```

Nearly 77k different software packages and their dependencies at this time of writing!

Some software that is not available in the managed repositories is available online from other authors and packaged with a .deb extension, the most common probably being the google chrome browser. For a new Linux install, it has to be downloaded from google:



Get Chrome for Linux

Debian/Ubuntu/Fedora/openSUSE.

Please select your download package:

[✓ 64 bit .deb (For Debian/Ubuntu)]
64 bit .rpm (For Fedora/openSUSE)

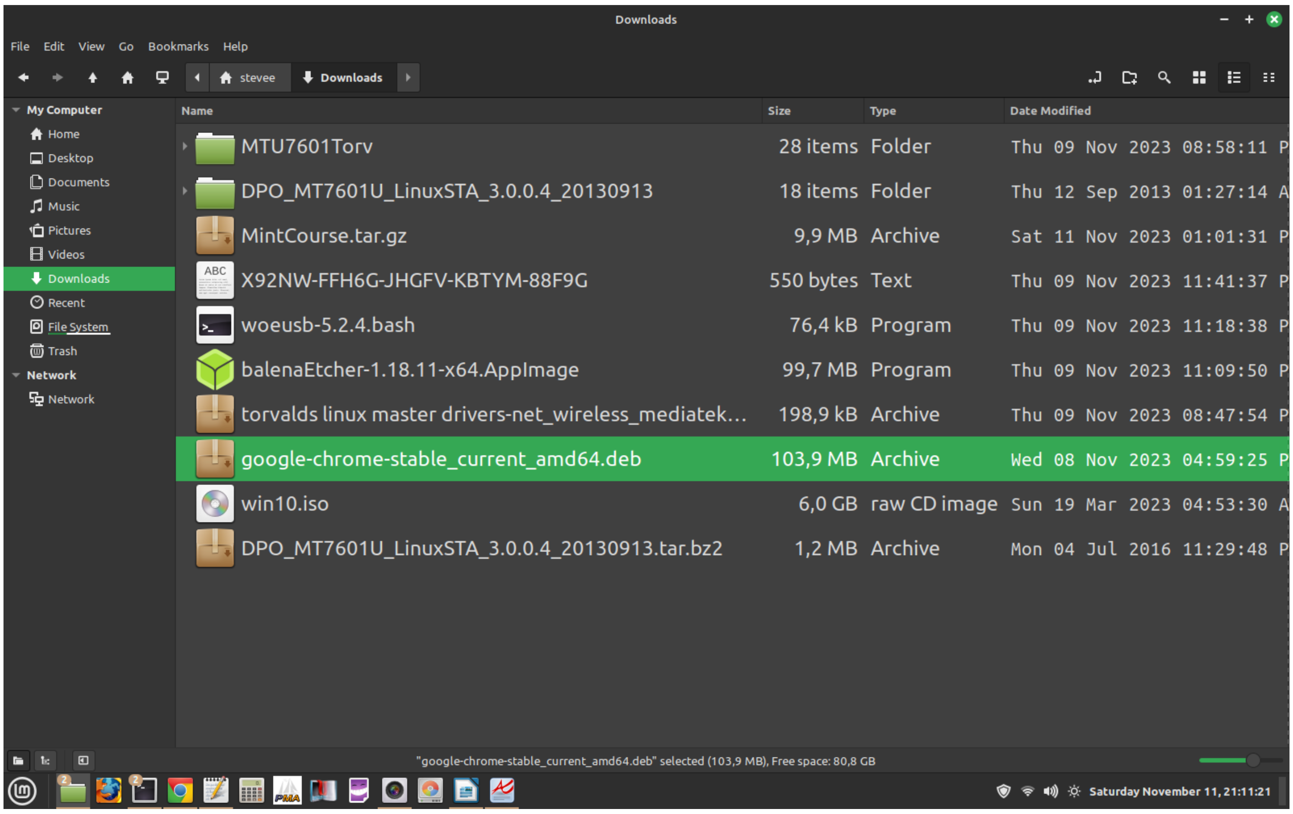
Not Debian/Ubuntu or Fedora/openSUSE? There may be a community-supported version for your distribution here.

By downloading Chrome, you agree to the Google Terms of Service and Chrome and ChromeOS Additional Terms of Service

Note: Installing Google Chrome will **add the Google repository** so that your system will automatically keep Google Chrome up to date. If you don't want Google's repository, do 'sudo touch/etc/default/google-chrome' before installing the package.

Accept and install

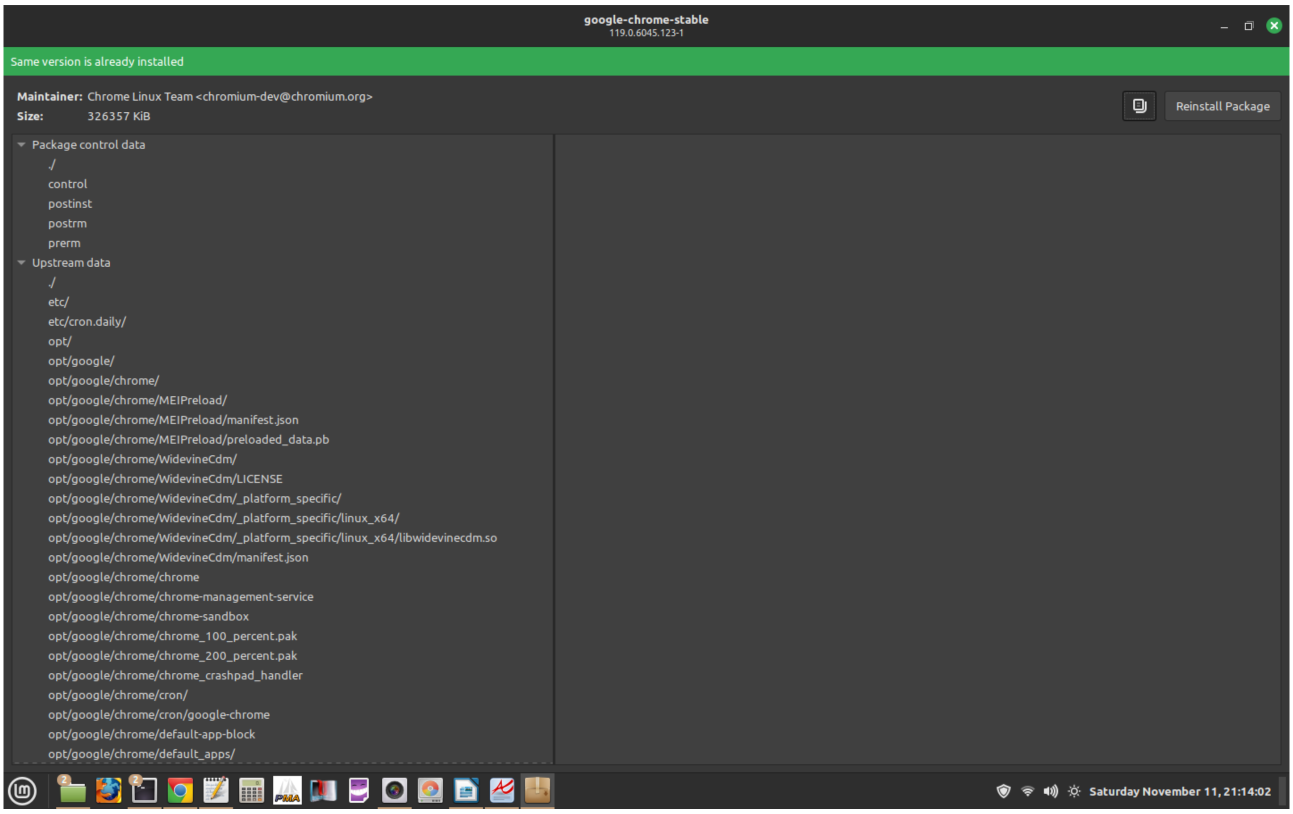
Once downloaded, it can be double clicked:



```
Name                                              Size        Type           Date Modified
MTU7601Torv                                       28 items    Folder         Thu 09 Nov 2023 08:58:11 P
DPO_MT7601U_LinuxSTA_3.0.0.4_20130913             18 items    Folder         Thu 12 Sep 2013 01:27:14 A
MintCourse.tar.gz                                 9,9 MB      Archive        Sat 11 Nov 2023 01:01:31 P
X92NW-FFH6G-JHGFV-KBTYM-88F9G                     550 bytes   Text           Thu 09 Nov 2023 11:41:37 P
woeusb-5.2.4.bash                                 76,4 kB     Program        Thu 09 Nov 2023 11:18:38 P
balenaEtcher-1.18.11-x64.AppImage                 99,7 MB     Program        Thu 09 Nov 2023 11:09:50 P
torvalds linux master drivers-net_wireless_mediatek…  198,9 kB  Archive      Thu 09 Nov 2023 08:47:54 P
google-chrome-stable_current_amd64.deb            103,9 MB    Archive        Wed 08 Nov 2023 04:59:25 P
win10.iso                                         6,0 GB      raw CD image   Sun 19 Mar 2023 04:53:30 A
DPO_MT7601U_LinuxSTA_3.0.0.4_20130913.tar.bz2     1,2 MB      Archive        Mon 04 Jul 2016 11:29:48 P
```

"google-chrome-stable_current_amd64.deb" selected (103,9 MB), Free space: 80,8 GB

The package management software installs it and any dependencies required:


```
google-chrome-stable
119.0.6045.123-1
Same version is already installed
Maintainer: Chrome Linux Team <chromium-dev@chromium.org>
Size:       326357 KiB
Package control data
    ./
    control
    postinst
    postrm
    prerm
Upstream data
    ./
    etc/
    etc/cron.daily/
    opt/
    opt/google/
    opt/google/chrome/
    opt/google/chrome/MEIPreload/
    opt/google/chrome/MEIPreload/manifest.json
    opt/google/chrome/MEIPreload/preloaded_data.pb
    opt/google/chrome/WidevineCdm/
    opt/google/chrome/WidevineCdm/LICENSE
    opt/google/chrome/WidevineCdm/_platform_specific/
    opt/google/chrome/WidevineCdm/_platform_specific/linux_x64/
    opt/google/chrome/WidevineCdm/_platform_specific/linux_x64/libwidevinecdm.so
    opt/google/chrome/WidevineCdm/manifest.json
    opt/google/chrome/chrome
    opt/google/chrome/chrome-management-service
    opt/google/chrome/chrome-sandbox
    opt/google/chrome/chrome_100_percent.pak
    opt/google/chrome/chrome_200_percent.pak
    opt/google/chrome/chrome_crashpad_handler
    opt/google/chrome/cron/
    opt/google/chrome/cron/google-chrome
    opt/google/chrome/default-app-block
    opt/google/chrome/default_apps/
Saturday November 11, 21:14:02
```

After this, the updates are handled along with all others by the Update Manager. The update commands are:

```
stevee@laptop:~$ sudo apt update
[sudo] password for stevee:
Get:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:2 https://ppa.launchpadcontent.net/obsproject/obs-studio/ubuntu jammy InRelease [18,1 kB]
Hit:3 https://ppa.launchpadcontent.net/ondrej/php/ubuntu jammy InRelease
Get:4 https://ppa.launchpadcontent.net/obsproject/obs-studio/ubuntu jammy/main amd64 Packages
[1.124 B]
Hit:5 http://mirrors.eze.sysarmy.com/ubuntu jammy InRelease
Get:6 http://mirrors.eze.sysarmy.com/ubuntu jammy-updates InRelease [119 kB]
Hit:7 https://dl.google.com/Linux/chrome/deb stable InRelease
Get:8 http://mirrors.eze.sysarmy.com/ubuntu jammy-backports InRelease [109 kB]
Ign:9 https://mirror.rackspace.com/Linuxmint/packages victoria InRelease
Get:10 http://mirrors.eze.sysarmy.com/ubuntu jammy-updates/main amd64 DEP-11 Metadata [101
kB]
Get:11 http://mirrors.eze.sysarmy.com/ubuntu jammy-updates/universe amd64 DEP-11 Metadata
[305 kB]
Get:12 http://mirrors.eze.sysarmy.com/ubuntu jammy-updates/multiverse amd64 DEP-11 Metadata
[940 B]
Get:13 http://mirrors.eze.sysarmy.com/ubuntu jammy-backports/main amd64 DEP-11 Metadata
[4.944 B]
```

```
Get:14 http://mirrors.eze.sysarmy.com/ubuntu jammy-backports/universe amd64 DEP-11 Metadata
[18,8 kB]
Hit:15 https://mirror.rackspace.com/Linuxmint/packages victoria Release
Fetched 787 kB in 17s (46,0 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
2 packages can be upgraded. Run 'apt list --upgradable' to see them.

stevee@laptop:~$ sudo apt upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages will be upgraded:
  firefox firefox-locale-en
2 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Need to get 0 B/71,2 MB of archives.
After this operation, 189 kB disk space will be freed.
Do you want to continue? [Y/n]
```

Linux as a Programming Environment
Linux can be used as an effective programming environment, though the industry standard GUI at the moment is MS Visual Studio Code, which is free from, and for MS Windows.

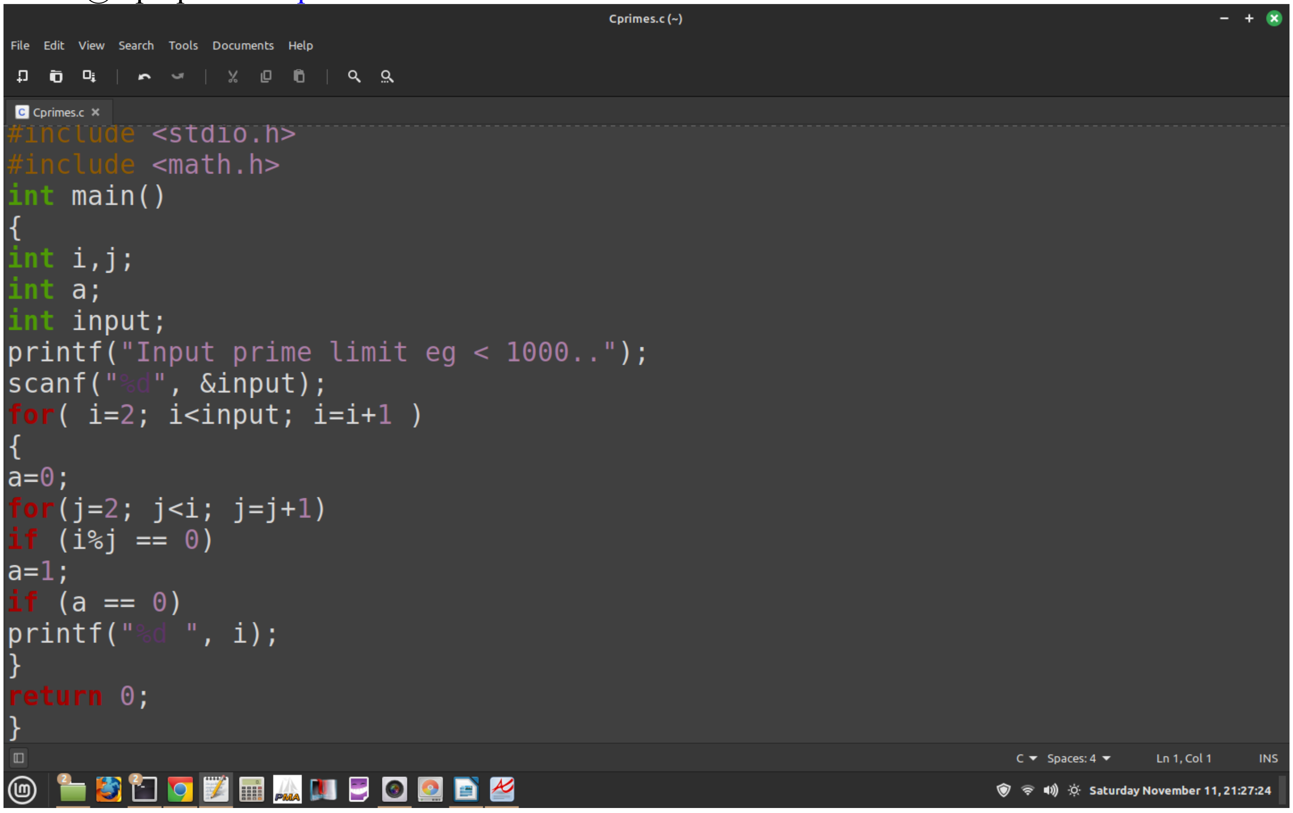
I will show 3 common programming language examples for the same Prime Numbers printing programme and what is required to compile and run them.

C programme language script:

```
#include <stdio.h>
#include <math.h>
int main()
{
int i,j;
int a;
int input;
printf("Input prime limit eg < 1000..");
scanf("%d", &input);
for( i=2; i<input; i=i+1 )
{
a=0;
for(j=2; j<i; j=j+1)
if (i%j == 0)
a=1;
if (a == 0)
printf("%d ", i);
}
return 0;
}
```

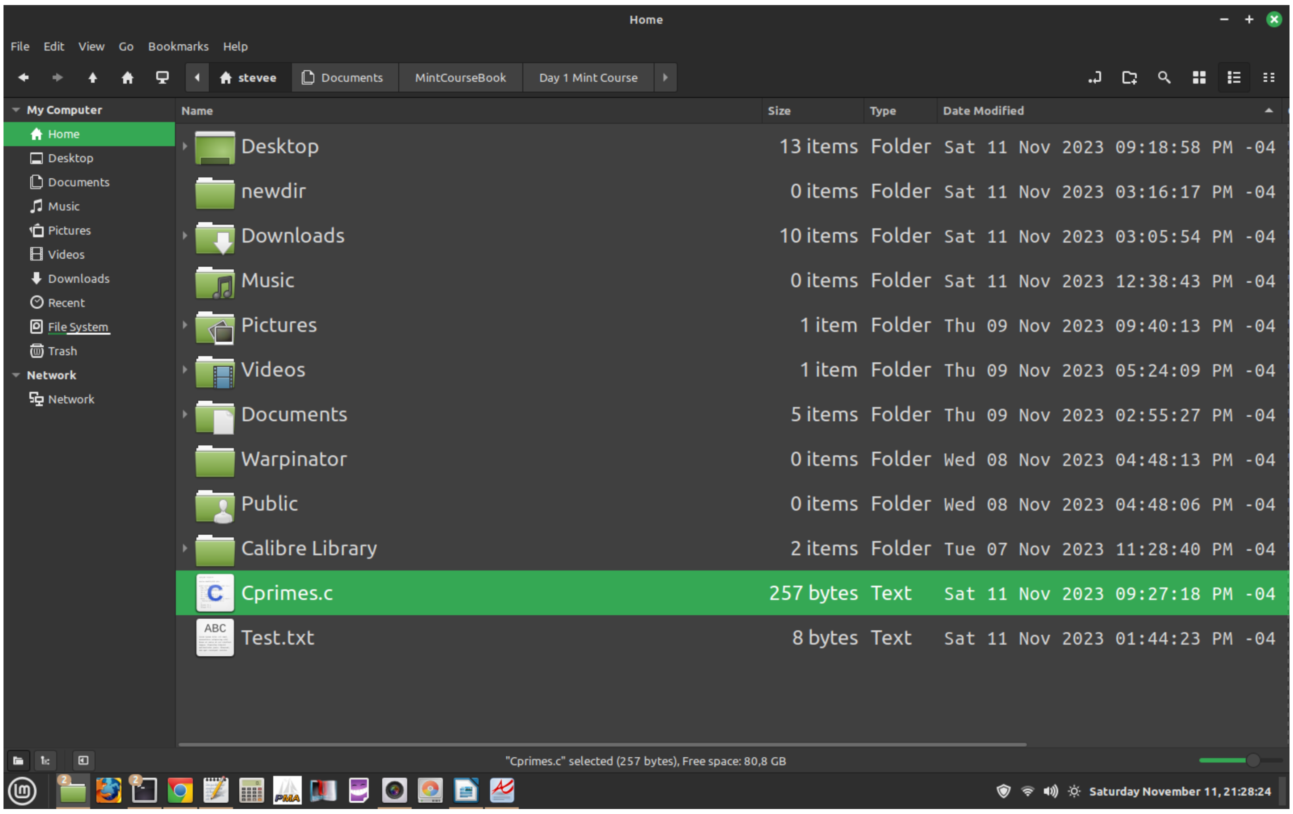
On your computer, copy this script into a text file using the Text editor, save it as Cprimes.c in your Home directory and view it with vim:

```
stevee@laptop:~$ vi Cprimes.c
```


Cprimes.c (~)
File Edit View Search Tools Documents Help
Cprimes.c
C Spaces: 4 Ln 1, Col 1 INS
Saturday November 11, 21:27:24

```
#include <stdio.h>
#include <math.h>
int main()
{
int i,j;
int a;
int input;
printf("Input prime limit eg < 1000..");
scanf("%d", &input);
for( i=2; i<input; i=i+1 )
{
a=0;
for(j=2; j<i; j=j+1)
if (i%j == 0)
a=1;
if (a == 0)
printf("%d ", i);
}
return 0;
}
```

As you see, Vim colourises the code according to Theme and C language automatically.


Home
File Edit View Go Bookmarks Help
stevee Documents MintCourseBook Day 1 Mint Course
My Computer
Home
Desktop
Documents
Music
Pictures
Videos
Downloads
Recent
File System
Trash
Network
Network
Saturday November 11, 21:28:24

Name	Size	Type	Date Modified
Desktop	13 items	Folder	Sat 11 Nov 2023 09:18:58 PM -04
newdir	0 items	Folder	Sat 11 Nov 2023 03:16:17 PM -04
Downloads	10 items	Folder	Sat 11 Nov 2023 03:05:54 PM -04
Music	0 items	Folder	Sat 11 Nov 2023 12:38:43 PM -04
Pictures	1 item	Folder	Thu 09 Nov 2023 09:40:13 PM -04
Videos	1 item	Folder	Thu 09 Nov 2023 05:24:09 PM -04
Documents	5 items	Folder	Thu 09 Nov 2023 02:55:27 PM -04
Warpinator	0 items	Folder	Wed 08 Nov 2023 04:48:13 PM -04
Public	0 items	Folder	Wed 08 Nov 2023 04:48:06 PM -04
Calibre Library	2 items	Folder	Tue 07 Nov 2023 11:28:40 PM -04
Cprimes.c	257 bytes	Text	Sat 11 Nov 2023 09:27:18 PM -04
Test.txt	8 bytes	Text	Sat 11 Nov 2023 01:44:23 PM -04

"Cprimes.c" selected (257 bytes), Free space: 80,8 GB

First, view the source code file permissions in /Home with ls :

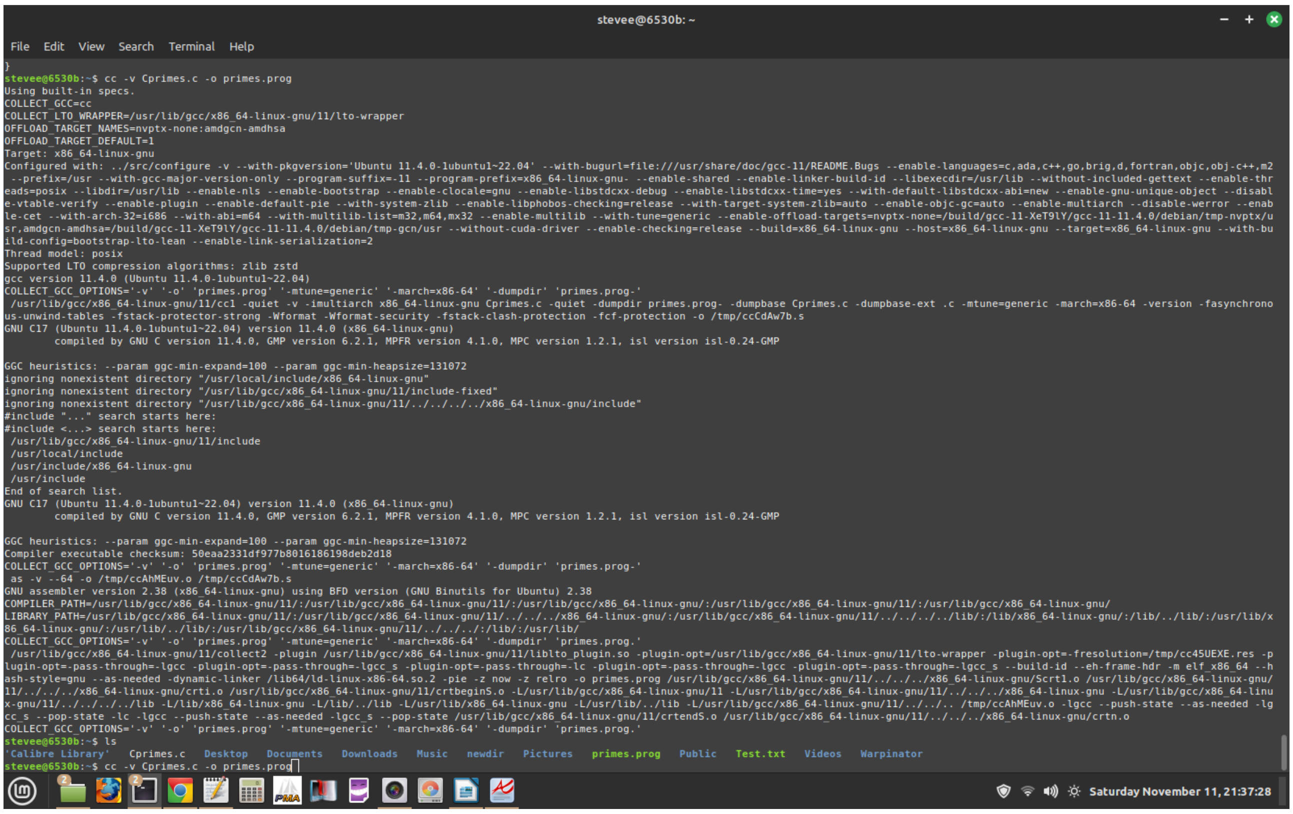
```
stevee@laptop:~$ ls -l
total 48
drwxrwxr-x 3 stevee stevee 4096 Nov  7 23:28 'Calibre Library'
-rw-rw-r-- 1 stevee stevee  257 Nov 11 21:27  Cprimes.c
```

View the code on the command line:

```
stevee@laptop:~$ cat Cprimes.c
#include <stdio.h>
#include <math.h>
int main()
{
int i,j;
int a;
int input;
printf("Input prime limit eg < 1000..");
scanf("%d", &input);
for( i=2; i<input; i=i+1 )
{
a=0;
for(j=2; j<i; j=j+1)
if (i%j == 0)
a=1;
if (a == 0)
printf("%d ", i);
}
return 0;
}
```

Now compile it with GNU C Compiler:

```
cc -v Cprimes.c -o primes.prog
```



```
}
stevee@6530b:~$ cc -v Cprimes.c -o primes.prog
Using built-in specs.
COLLECT_GCC=cc
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/11/lto-wrapper
OFFLOAD_TARGET_NAMES=nvptx-none:amdgcn-amdhsa
OFFLOAD_TARGET_DEFAULT=1
Target: x86_64-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Ubuntu 11.4.0-1ubuntu1~22.04' --with-bugurl=file:///usr/share/doc/gcc-11/README.Bugs --enable-languages=c,ada,c++,go,brig,d,fortran,objc,obj-c++,m2
 --prefix=/usr --with-gcc-major-version-only --program-suffix=-11 --program-prefix=x86_64-linux-gnu- --enable-shared --enable-linker-build-id --libexecdir=/usr/lib --without-included-gettext --enable-thr
eads=posix --libdir=/usr/lib --enable-nls --enable-bootstrap --enable-clocale=gnu --enable-libstdcxx-debug --enable-libstdcxx-time=yes --with-default-libstdcxx-abi=new --enable-gnu-unique-object --disabl
e-vtable-verify --enable-plugin --enable-default-pie --with-system-zlib --enable-libphobos-checking=release --with-target-system-zlib=auto --enable-objc-gc=auto --enable-multiarch --disable-werror --enab
le-cet --with-arch-32=i686 --with-abi=m64 --with-multilib-list=m32,m64,mx32 --enable-multilib --with-tune=generic --enable-offload-targets=nvptx-none=/build/gcc-11-XeT9lY/gcc-11-11.4.0/debian/tmp-nvptx/u
sr,amdgcn-amdhsa=/build/gcc-11-XeT9lY/gcc-11-11.4.0/debian/tmp-gcn/usr --without-cuda-driver --enable-checking=release --build=x86_64-linux-gnu --host=x86_64-linux-gnu --target=x86_64-linux-gnu --with-bu
ild-config=bootstrap-lto-lean --enable-link-serialization=2
Thread model: posix
Supported LTO compression algorithms: zlib zstd
gcc version 11.4.0 (Ubuntu 11.4.0-1ubuntu1~22.04)
COLLECT_GCC_OPTIONS='-v' '-o' 'primes.prog' '-mtune=generic' '-march=x86-64' '-dumpdir' 'primes.prog-'
 /usr/lib/gcc/x86_64-linux-gnu/11/cc1 -quiet -v -imultiarch x86_64-linux-gnu Cprimes.c -quiet -dumpdir primes.prog- -dumpbase Cprimes.c -dumpbase-ext .c -mtune=generic -march=x86-64 -version -fasynchrono
us-unwind-tables -fstack-protector-strong -Wformat -Wformat-security -fstack-clash-protection -fcf-protection -o /tmp/ccCdAw7b.s
GNU C17 (Ubuntu 11.4.0-1ubuntu1~22.04) version 11.4.0 (x86_64-linux-gnu)
        compiled by GNU C version 11.4.0, GMP version 6.2.1, MPFR version 4.1.0, MPC version 1.2.1, isl version isl-0.24-GMP

GGC heuristics: --param ggc-min-expand=100 --param ggc-min-heapsize=131072
ignoring nonexistent directory "/usr/local/include/x86_64-linux-gnu"
ignoring nonexistent directory "/usr/lib/gcc/x86_64-linux-gnu/11/include-fixed"
ignoring nonexistent directory "/usr/lib/gcc/x86_64-linux-gnu/11/../../../../x86_64-linux-gnu/include"
#include "..." search starts here:
#include <...> search starts here:
 /usr/lib/gcc/x86_64-linux-gnu/11/include
 /usr/local/include
 /usr/include/x86_64-linux-gnu
 /usr/include
End of search list.
GNU C17 (Ubuntu 11.4.0-1ubuntu1~22.04) version 11.4.0 (x86_64-linux-gnu)
        compiled by GNU C version 11.4.0, GMP version 6.2.1, MPFR version 4.1.0, MPC version 1.2.1, isl version isl-0.24-GMP

GGC heuristics: --param ggc-min-expand=100 --param ggc-min-heapsize=131072
Compiler executable checksum: 50eaa2331df977b8016186198deb2d18
COLLECT_GCC_OPTIONS='-v' '-o' 'primes.prog' '-mtune=generic' '-march=x86-64' '-dumpdir' 'primes.prog-'
 as -v --64 -o /tmp/ccAhMEuv.o /tmp/ccCdAw7b.s
GNU assembler version 2.38 (x86_64-linux-gnu) using BFD version (GNU Binutils for Ubuntu) 2.38
COMPILER_PATH=/usr/lib/gcc/x86_64-linux-gnu/11/:/usr/lib/gcc/x86_64-linux-gnu/11/:/usr/lib/gcc/x86_64-linux-gnu/:/usr/lib/gcc/x86_64-linux-gnu/11/:/usr/lib/gcc/x86_64-linux-gnu/
LIBRARY_PATH=/usr/lib/gcc/x86_64-linux-gnu/11/:/usr/lib/gcc/x86_64-linux-gnu/11/../../../x86_64-linux-gnu/:/usr/lib/gcc/x86_64-linux-gnu/11/../../../../lib/:/lib/x86_64-linux-gnu/:/lib/../lib/:/usr/lib/x
86_64-linux-gnu/:/usr/lib/../lib/:/usr/lib/gcc/x86_64-linux-gnu/11/../../../:/lib/:/usr/lib/
COLLECT_GCC_OPTIONS='-v' '-o' 'primes.prog' '-mtune=generic' '-march=x86-64' '-dumpdir' 'primes.prog.'
 /usr/lib/gcc/x86_64-linux-gnu/11/collect2 -plugin /usr/lib/gcc/x86_64-linux-gnu/11/liblto_plugin.so -plugin-opt=/usr/lib/gcc/x86_64-linux-gnu/11/lto-wrapper -plugin-opt=-fresolution=/tmp/cc45UEXE.res -p
lugin-opt=-pass-through=-lgcc -plugin-opt=-pass-through=-lgcc_s -plugin-opt=-pass-through=-lc -plugin-opt=-pass-through=-lgcc -plugin-opt=-pass-through=-lgcc_s --build-id --eh-frame-hdr -m elf_x86_64 --h
ash-style=gnu --as-needed -dynamic-linker /lib64/ld-linux-x86-64.so.2 -pie -z now -z relro -o primes.prog /usr/lib/gcc/x86_64-linux-gnu/11/../../../x86_64-linux-gnu/Scrt1.o /usr/lib/gcc/x86_64-linux-gnu/
11/../../../x86_64-linux-gnu/crti.o /usr/lib/gcc/x86_64-linux-gnu/11/crtbeginS.o -L/usr/lib/gcc/x86_64-linux-gnu/11 -L/usr/lib/gcc/x86_64-linux-gnu/11/../../../x86_64-linux-gnu -L/usr/lib/gcc/x86_64-linu
x-gnu/11/../../../../lib -L/lib/x86_64-linux-gnu -L/lib/../lib -L/usr/lib/x86_64-linux-gnu -L/usr/lib/../lib -L/usr/lib/gcc/x86_64-linux-gnu/11/../../.. /tmp/ccAhMEuv.o -lgcc --push-state --as-needed -lg
cc_s --pop-state -lc -lgcc --push-state --as-needed -lgcc_s --pop-state /usr/lib/gcc/x86_64-linux-gnu/11/crtendS.o /usr/lib/gcc/x86_64-linux-gnu/11/../../../x86_64-linux-gnu/crtn.o
COLLECT_GCC_OPTIONS='-v' '-o' 'primes.prog' '-mtune=generic' '-march=x86-64' '-dumpdir' 'primes.prog.'
stevee@6530b:~$ ls
'Calibre Library'   Cprimes.c   Desktop   Documents   Downloads   Music   newdir   Pictures   primes.prog   Public   Test.txt   Videos   Warpinator
stevee@6530b:~$ cc -v Cprimes.c -o primes.prog
```

Now run the compiled executable file, primes.prog:

```
stevee@laptop:~$ ./primes.prog
Input prime limit eg < 1000..999
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109 113 127
131 137 139 149 151 157 163 167 173 179 181 191 193 197 199 211 223 227 229 233 239 241 251
257 263 269 271 277 281 283 293 307 311 313 317 331 337 347 349 353 359 367 373 379 383 389
397 401 409 419 421 431 433 439 443 449 457 461 463 467 479 487 491 499 503 509 521 523 541
547 557 563 569 571 577 587 593 599 601 607 613 617 619 631 641 643 647 653 659 661 673 677
683 691 701 709 719 727 733 739 743 751 757 761 769 773 787 797 809 811 821 823 827 829 839
853 857 859 863 877 881 883 887 907 911 919 929 937 941 947 953 967 971 977 983 991 997
```

The last prime number under 1000 is 997. You can experiment with a range up to 100000 which took 21 seconds on my old dual core laptop.

A similar program written in Python:

Copy the code (red) into a text file called PythonPrimes.py using a Text editor, as Python has to be Tab indented correctly to run:

```
upper = 100000
for i in range(2, upper + 1):
    for j in range(2, i):
        if (i % j) == 0:
            break
    else:
        print(i, end=' ')
print(" Done")
```

```
It is NOT executable yet! No x permissions:
```

```
stevee@laptop:~$ ls -l PythonPrimes.py
-rw-rw-r-- 1 stevee stevee 198 Nov 11 21:47 PythonPrimes.py
```

To run the file, it needs to be made executable. The Python libraries installed in Linux should run it:

```
stevee@laptop:~$ chmod +x PythonPrimes
stevee@laptop:~$ ls -l PythonPrimes
-rwxrwxr-x 1 stevee stevee 198 Nov 11 21:47 PythonPrimes.py
```

See the difference in x file attributes covered in Day 4?

Now run it:

```
stevee@laptop:~$ python3 PythonPrimes.py
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97  Done
```

Change the range as you want by editing the code.
To print to 100,000 this program took 75 secs:
This slow speed (compared to the C example) is a consequence of many factors beyond the scope of this text. Python is generally a very fast language.

```
... 99721 99733 99761 99767 99787 99793 99809 99817 99823 99829 99833 99839 99859 99871
99877 99881 99901 99907 99923 99929 99961 99971 99989 99991  Done
```

Java requires the Developer Kit installed:

```
stevee@laptop:~$ sudo apt-get install openjdk-19-jdk
```

View the source code:

```
cat primes.java
```

```
------------------------
public class primes {
   public static void main(String args[]) {
      int i, j, a;
      int limit = 100000;
      for (i = 2; i <= limit; i = i + 1) {
         a = 0;
         for (j = 2; j < i; j = j + 1)
            if (i % j == 0)
               a = 1;
         if (a == 0)
            System.out.print(" " + i);
      }
      System.out.print(" Done\n");
   }
}

------------------------
```

Compile the code – the file primes.class is created:

```
javac primes.java
```

The java command id's the class file and runs it:

```
java primes
```

```
.. 99809 99817 99823 99829 99833 99839 99859 99871 99877 99881 99901 99907 99923 99929
99961 99971 99989 99991 Done
```

This took 24 seconds. Slower than C but faster than Python, but bear in mind, the scripts are not technically identical between languages, so operation.

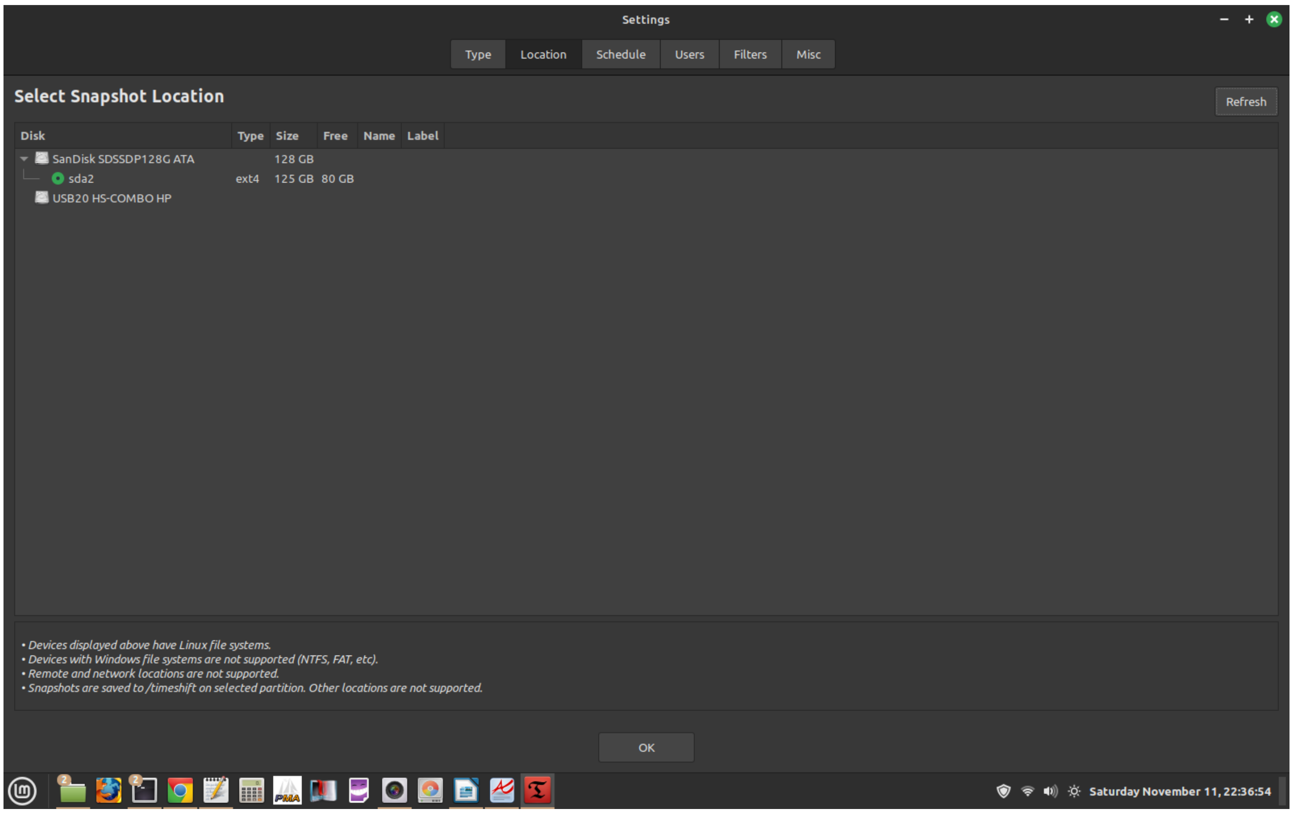
These programs are very inefficient because they re-run the division by 2 Prime check as each new prime is found, so waste more and more cycles as the amount of primes found increases.
A better algorithm for finding prime numbers you can research is the "Eratosthenes Sieve".

The examples are to show the flexibility and convenience of the Linux system and its associated software.

Disaster Prevention: Backups

I can tell you, from IT industry experience within a small printing firm. that I've seen a Director almost in tears of desperation when a client list was lost due to insufficient interest in taking the company's data value seriously until it was too late and the only client list copy (so he thought) was lost forever from his PC after a crash, because he was always "too busy right now" to learn how, or let me back up his PC system adequately and set up scheduled daily backups after hours etc.

Nowadays, most OS’s have “snapshot” style Applications built in like Timeshift to take copies of the system files to guard against crash corruption and/or accidental or malicious deletion (viral infection etc.).



Select Snapshot Location

Disk	Type	Size	Free	Name	Label
SanDisk SDSSDP128G ATA		128 GB			
sda2	ext4	125 GB	80 GB		
USB20 HS-COMBO HP					

- *Devices displayed above have Linux file systems.*
- *Devices with Windows file systems are not supported (NTFS, FAT, etc).*
- *Remote and network locations are not supported.*
- *Snapshots are saved to /timeshift on selected partition. Other locations are not supported.*

All very well, but unless that snapshot is saved on a different drive external to the PC, all will still be lost if the PC hard drive fails or data is deleted or irreversibly corrupted!

I prefer to use Terminal methods to keep backups of my data as it is a consistent process, rather than learning various GUI backup packages.

I have multiple SSD drives of varying sizes, as the cost per gigabyte is very low now and using a USB dongle connector similar to this:

I can connect SSDs directly to my laptop USB and run rsync – a very fast, differential file copy utility.

Differential means that once an initial full copy of all the files you want to backup is done, then backup runs after that only update the files that have changed, not everything, so saves a lot of time.

You can do a simple backup of your Pictures folder contents to your Music folder to understand the format – remember the * means none or any character, so covers all possible file names in a folder:

```
stevee@laptop:~$ rsync Pictures/* Music/ -vahn
sending incremental file list
WebCamBlurred.png

sent 75 bytes  received 19 bytes  188,00 bytes/sec
total size is 991,68K  speedup is 10.549,84 (DRY RUN)
```

If you wanted the whole Pictures folder *and* its contents transferred, not *just* the contents, the subtle difference is:

```
rsync Pictures Music/ -vahn
```

The switches: -vahn means:
-v verbose (gives more user info during the process)
-a archive (runs in recursive mode to copy all files and sub folder contents)
-h (gives file sizes in human readable format e.g. MB)
-n (DRY RUN) – This MUST be run every time to check that the command is doing what you think it is! If you ran certain -delete copy options then you may delete files you did not intend, so get in the habit of using the -n DRY RUN option before you run the command for real!!

Read :

```
man rsync
NAME
```

```
    rsync - a fast, versatile, remote (and local) file-copying tool
```

When you are happy rsync is copying what you want correctly, remove the -n from -vahn and run the command for real.

For the content only:

```
stevee@laptop:~$ rsync Pictures/* Music/ -vah
sending incremental file list
WebCamBlurred.png
sent 992,04K bytes received 35 bytes  1,98M bytes/sec
total size is 991,68K speedup is 1,00

stevee@laptop:~$ ls Music/Pictures/
WebCamBlurred.png
```

For the Pictures folder with content:

```
stevee@laptop:~$ rsync Pictures Music/ -vah
sending incremental file list
Pictures/
Pictures/WebCamBlurred.png
sent 992,08K bytes  received 39 bytes  1,98M bytes/sec
total size is 991,68K  speedup is 1,00
```

This is subtle but important difference to appreciate so you don’t end up copying or merging folders incorrectly, say, with a folder copied inside another of the same name, wasting space and causing messy filesystems etc.

I use the rsync command a lot.

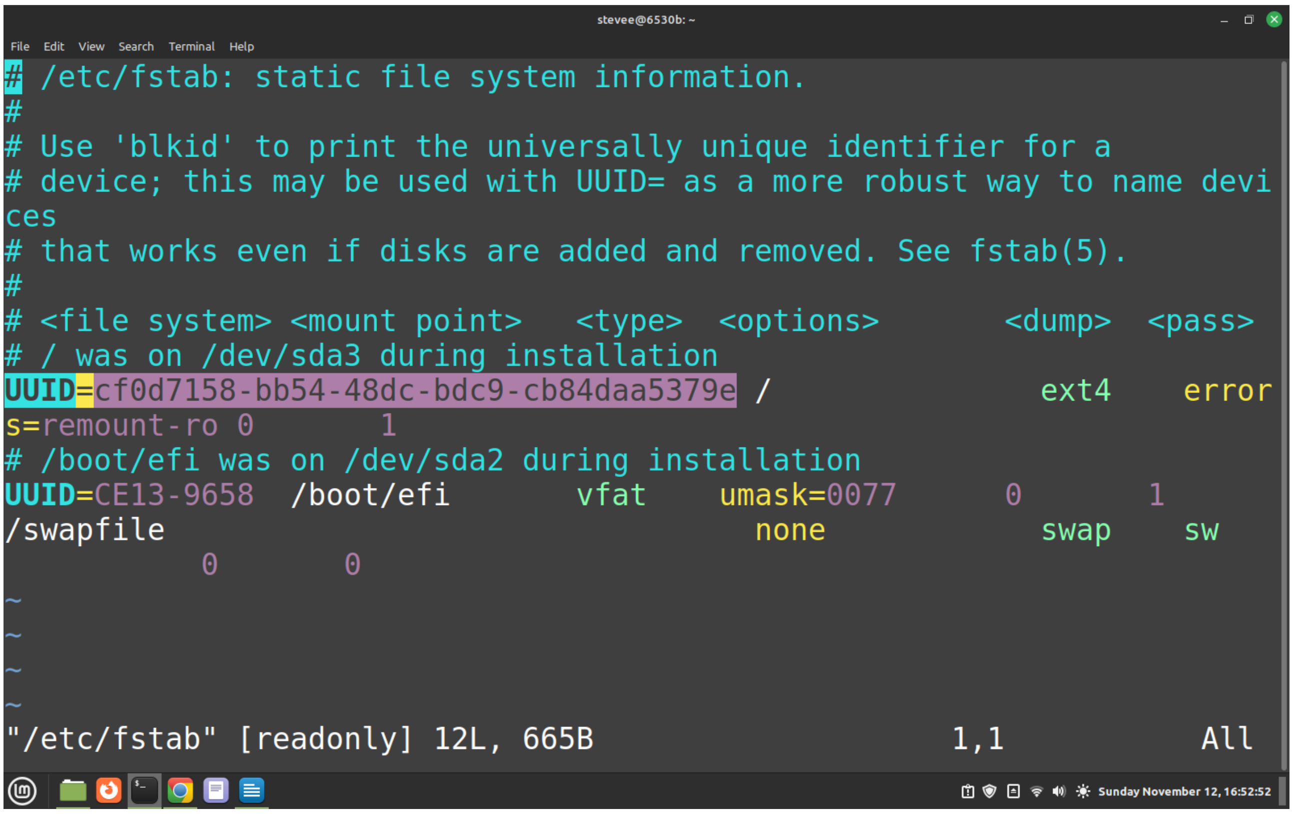
It can be used with compression and SSH encryption across slower network links too.

A major realisation I made and developed was being able to clone a working Linux install to a fresh install of the same flavour or repairing a running but partially corrupt installation.
It involved working out what dynamic, temporary processes needed to be skipped to prevent the copy process interfering with a running installation then changing the UID numbers back to their original values in the files:

```
 stevee@laptop:~$ vi /boot/grub/grub.cfg
```

and

```
stevee@laptop:~$ vi /etc/fstab
```


stevee@6530b: ~
File Edit View Search Terminal Help
Sunday November 12, 16:52:52

```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devi
ces
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/sda3 during installation
UUID=cf0d7158-bb54-48dc-bdc9-cb84daa5379e /               ext4    error
s=remount-ro 0       1
# /boot/efi was on /dev/sda2 during installation
UUID=CE13-9658  /boot/efi       vfat    umask=0077      0       1
/swapfile                                 none            swap    sw
           0       0
~
~
~
~
"/etc/fstab" [readonly] 12L, 665B                          1,1           All
```

before rebooting the PC again.

If this sounds to complex, don’t worry about understanding it at a beginner level.

Its further described in my other eBook “All My IT Knowledge Posts” – full of many more topics I have explored over the last 20 years of being a Linux hobbyist, IT and comms tech.

I cannot stress enough that keeping as many backups of all your important folders and files across as many different media types as possible will save you much angst – use Gdrive, Gmail attachments, pen drives, SD cards, SSD drives, your cell phone and whatever else you can afford to keep as many versions of your data as you can! You won’t regret it.

I hope this book fulfils its intent of being a suitable introduction to some interesting and useful topics and acts as a catalyst for further study for those of the right inclination.

Linux has provided me with a massive technical education because of its open-source model, as the system is open to investigation being non-proprietary, with a massive online support from very experienced Linux admin gurus providing free information on all types of issues.

It has enabled me to understand the workings of computer systems, data, network and electrical processes at a much deeper level than would have been possible with just the limited access that Windows offered historically, thankfully much improved nowadays, due to the necessary inter-operation of Windows and Linux systems worldwide as part of the WWW network.