

```
>>> Linux 101
>>> The terminal is your friend.
```

```
Name: João Pedro Dias
      jpmdias@fe.up.pt
      Operating Systems
Date: February 12, 2019
```



>>> The Origin

Back in 1991, the 21 years old Linus Torvalds posted the following to comp.os.minix, a newsgroup on Usenet:

I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones. This has been brewing since April, and is starting to get ready. I'd like any feedback on things people like/dislike in minix, as my OS resembles it somewhat (same physical layout of the file-system (due to practical reasons) among other things).

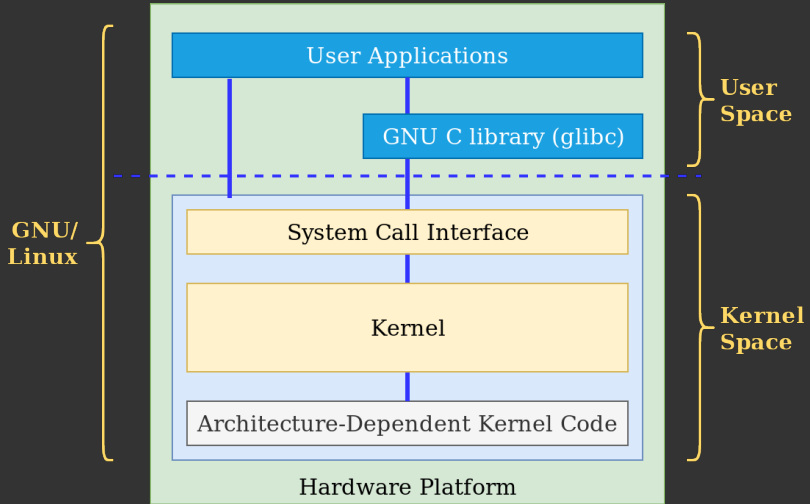
>>> Disclaimer

Linux is the kernel, one of the essential major components of the system. The system as a whole is basically the GNU system, with Linux added. When you're talking about this combination, please call it "GNU/Linux" .

Linux and the GNU System
by Richard Stallman

>>> Kernel versus Operating System

The kernel is part of the operating system.



>>> Linux kernel?

- * Is a free and open-source, monolithic, Unix-like operating system kernel.
- * Was conceived and created in 1991 by Linus Torvalds.
- * It is highly-influenced by MINIX (Andrew S. Tanenbaum)
 - * Linux is a monolithic kernel rather than a microkernel (MINIX).
- * Is a core part of most non-Windows Operating Systems: Ubuntu, Mint, Debian, Android,...

>>> Unix, POSIX and Linux

- * Unix is a family of operating systems, originally started at Bell Labs (by Thompson, Ritchie, Kernighan et al.). Later, Unix inspired POSIX.
- * POSIX (Portable Operating System Interface) is an IEEE standard for UNIX systems.
 - * POSIX 7 defines: C API, CLI utilities and API, shell language, environment variables, program exit status, regular expressions, directory structure (FHS) and filenames.
- * Linux is (*nearly*, and *quite*) POSIX compliant, and quite inspired by Unix. Mac OS, Solaris and BSD are other examples of Unix-like OSs.
 - * Unix is proprietary software and Linux is FOSS (Free and Open-Source Software).

>>> Linux Distributions

- * Linux distribution (a.k.a. *distro*) is an operating system made from a software collection, which is based upon the Linux kernel and, often, a package management system.
- * A typical *Linux distribution* comprises a Linux kernel, GNU tools and libraries, additional software, documentation, a window system (the most common being the X Window System¹), a window manager, and a desktop environment.

¹X Window System is a windowing system for bitmap displays, common on UNIX-like computer operating systems. X provides the basic framework for a GUI environment.

>>> Linux solves all the problems?

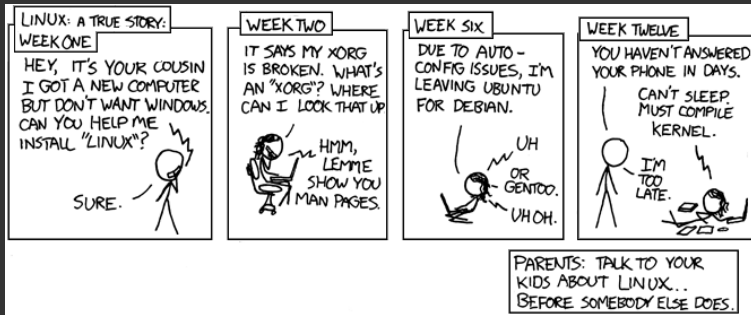
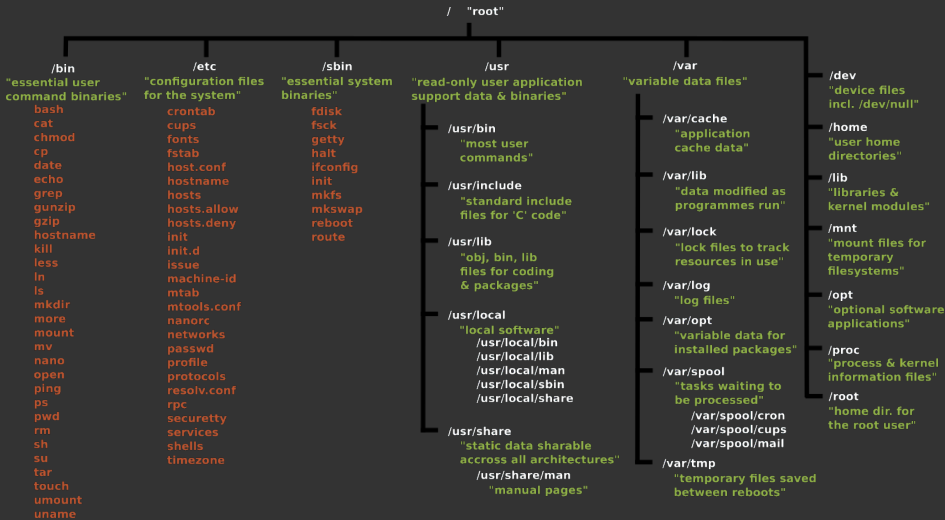


Figure: This really is a true story, and she doesn't know I put it in my comic because her wifi hasn't worked for weeks.

¹<https://xkcd.com/456/>

>>> Filesystem Hierarchy Standard



>>> FHS Simplified

- / root directory (like C: in Windows)
- /bin system binaries/applications (e.g. cd, echo, rm)
 - * /usr/bin - user binaries (e.g. firefox)
- /sbin administration system tools (e.g. shutdown)
- /boot GRUB, etc
- /usr user files and applications
- /dev devices (/dev/sd[a-z][1-9]*)
 - * /dev/null, /dev/zero and other pseudo-devices
- /etc configurations (system-wide)
- /home contains a folder for each user, /home/<user>
with their files and configurations (user-specific)
- /root home of root user (/ != /root)
- /tmp temporary files (e.g. Web browsing cache)
- /opt optional software packages
- /proc process files
- /var files that change recurrently (e.g. backups, logs)

>>> File Permissions

* \$ ls -la

* List all the content of a given directory (folder)

```
-rw----- 1 root  root   576 Apr 17  weather.txt
drwxr-xr-x 6 root  root  1024 Oct  9  web_page
-rw-rw-r-- 1 root  root  6480 Feb 11  web_site.tar
-rw----- 1 root  root  5743 Dec 16  xmas_file.txt
-----
```

							File Name
					+-----		Modification Time
				+-----			Size (in bytes)
			+-----				Group
		+-----					Owner
	+-----						No. of dirs/links
+-----							File Permissions

>>> File Permissions

- * How to read drwxr-xr-x or -rw-r--r-- ?
- * User, group and others: each one has an octal according to their permissions over a specific file.
- * r (read), w (write), x (execute)
- * `| - | --- | --- | --- |`
 - * `| directory? | user | group | others |`

binary	octal	permission level
000	0	---
001	1	--x
010	2	-w-
011	3	-wx
100	4	r--
101	5	r-x
110	6	rw-
111	7	rwX

>>> Changing Permissions and Ownership

- * `$ chmod <permission code> <file or directory>`
 - * `$ chmod 777 <file> : -rwxrwxrwx` (read, write and execute to all users)
 - * `$ chmod +x <file> : execute permission given to all users`
 - * `$ chmod -r <file> : read permission revoked to all users`

- * `$ chown <user> <file or directory>`
 - * `$ chown root file.txt` (owner = root)
 - * `$ chown jp:students file.txt` (owner = jp, group = students)
 - * `$ chown -R jp:students ./exams` (similar to the last one, but changes the ownership of a directory)

>>> Shell and Terminal

- * The shell is a program that takes commands from the keyboard and gives them to the operating system to perform. In the old days command line interfaces (CLIs), were the only user interface available on a Unix-like system such as Linux.
- * On most Linux systems a program called `bash`² acts as the shell program.
 - * Alternatives to `bash`: `ksh`, `tcsh`, `zsh` and `fish`.
- * Nowadays, we have graphical user interfaces (GUIs) in addition to CLIs.
- * Terminal: *terminal emulator*. A program that opens a window and lets you interact with the shell.
 - * Known terminal emulators: `gnome-terminal`, `konsole`, `xterm`, `rxvt`, `kvt`, `nxtterm`, `eterm` and `terminator`.

²`bash` stands for Bourne Again SHell, an enhanced version of the original Unix shell program, `sh`, written by Steve Bourne

>>> CLI commands: The survival guide (1/4)

* Command

- * A *command* is an instruction given by a user telling a computer to do something, such as run a single program or a group of linked programs.

* Linux Command Chaining

AND (&&) This command that follows this operator will execute only if the preceding command executes successfully.

OR (||) The command that follows will execute only if the preceding command fails.

S'Colon (;) The succeeding commands will execute regardless of the exit status of the command that precedes it.

Pipe (|) The output of the previous command acts as the input to the next command in the chain.

Ampersand (&) This sends the current command to the background.

>, <, >> The operator can be used to redirect the output of a command or a group of commands to a stream or file.

>>> CLI commands: The survival guide (2/4)

- * Executing commands as root: `$ sudo <command>`
- * Login as root: `$ sudo su`
- * If the last character of your shell prompt is `#` rather than `$`, you are operating as the *superuser* (root).
- * `~/.bash_profile` is the name of file used to store the bash environment settings (e.g. personal configurations).
- * Shell scripting (file `.sh`): A quick-and-dirty method of prototyping complex applications.

>>> CLI commands: The survival guide (3/4)

- `cd` change directory
- `ls` list directory content
- `pwd` present working directory
- `touch` create new file or update file modification date
- `cat` prints file content to stdout
- `more` prints file content with pagination
- `head` prints the begin of a file content
- `tail` prints the end of a file content
- `tar` compress/decompress files
- `mkdir` make a new directory
- `ln` create a shortcut/link
- `rm` remove file
- `rmdir` remove directory
- `mv` move files or directories

>>> CLI commands: The survival guide (4/4)

find search for a file or directory
which find commands and their location
 * **which** ssh => /usr/bin/ssh
grep find a string inside a file or inside any file in a folder
awk pattern scanning
sed stream editor, similar to awk
ps process snapshot, ls of processes
top task manager
echo prints a string to the stdout
ip network configuration
passwd set or change user password
man manual, presents the meaning, functionality and syntax of any command

man pages = awesome

(sometimes. Quality may vary 😊)

JULIA EVANS
@b0rk



I found out I can get documentation for programs (like grep) with **man grep**!

but that's not all!! lots of other things have man pages too!



man pages are split up into 8 sections

① ② ③ ④ ⑤ ⑥ ⑦ ⑧

/usr/share/man/man5

has section 5 on my machine.

GREAT

① programs

\$ man grep
\$ man ls

③ C functions

\$ man 3 printf
\$ man fopen

⑤ file formats

\$ man sudoers
for /etc/sudoers
→ \$ man proc

⑦ miscellaneous

\$ man 7 pipe
\$ man 7 symlink
(these are cool!)

② system calls

\$ man sendfile

④ devices

\$ man null
for /dev/null docs

⑥ games

(not very useful)
man sl is good if you have sl though

⑧ sysadmin programs

\$ man apt
\$ man chroot

Figure: From julia's drawings, <https://drawings.jvns.ca/>

>>> Compiling and Executing C code

- * cc or gcc or g++ commands
- * gcc -Wall -o <name_of_executable> <name_of_source_code>
 - * Compile Example: gcc -o hello helloWorld.c
 - * Run Example: ./hello

* Makefile

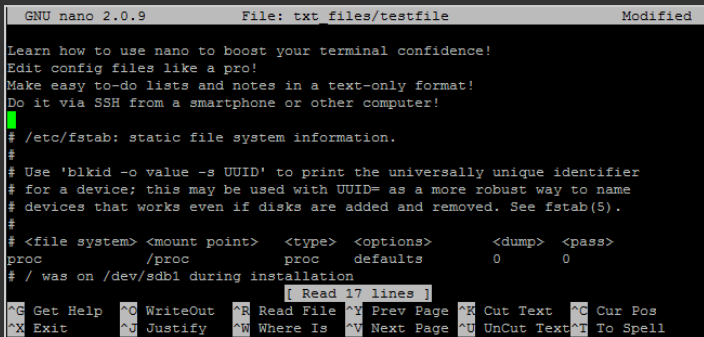
- * make (compiles the code by using the makefile)
- * make clean (removes the compile result)

```
# build an executable named hello from helloWorld.c
all: helloWorld.c
    gcc -g -Wall -o hello helloWorld.c

clean:
    $(RM) hello
```

>>> Code Editors

- * CLI editors: vi, vim, nano, emacs
- * GUI editors: Visual Studio Code, Atom, Sublime Text



```
GNU nano 2.0.9          File: txt_files/testfile          Modified
Learn how to use nano to boost your terminal confidence!
Edit config files like a pro!
Make easy to-do lists and notes in a text-only format!
Do it via SSH from a smartphone or other computer!
#
# /etc/fstab: static file system information.
#
# Use 'blkid -o value -s UUID' to print the universally unique identifier
# for a device; this may be used with UUID= as a more robust way to name
# devices that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point> <type> <options>          <dump> <pass>
proc            /proc           proc            defaults        0             0
# / was on /dev/sdb1 during installation
[ Read 17 lines ]
^G Get Help   ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit       ^J Justify   ^W Where Is  ^V Next Page  ^U UnCut Text ^I To Spell
```

Figure: nano CLI interface

>>> Services

- * Services: Programs that run in the background (a.k.a. *daemon*) with different responsibilities.
- * `/etc/init.d/` : directory that contains links to initialization scripts (autorun on OS boot)
- * Examples: `networking`, `sshd`, `apache2`, `mysql`, `network-manager`, `cron`

- * How to start, stop and restart a service?
 - * `service <service_name> start|stop|restart`
 - * `/etc/init.d/<service_name> start|stop|restart`
 - * `systemctl start|stop|restart <service_name>`

>>> Installing Software

- * The easy way: apt (yum - Fedora / pacman - Arch)
 - * `$ sudo apt install <program_name>` (install a new program)
 - * `$ sudo apt remove <program_name>` (remove an installed program)
 - * `$ sudo apt update && sudo apt-get upgrade` (update the software lists and upgrade the installed software)
 - * `$ apt-cache search <keyword>` (search for a program)
 - * `$ apt moo (?easter-egg?)`
- * Other ways:
 - * Executing the file: `chmod +x <program>` and then `./program`
 - * With `.deb` files: `$ dpkg -i <file .deb>`

>>> Installing Software

- * apt can't find the program that I need...
 - * Add new Personal Package Archives (PPAs).
 - * Example
 - * `$ sudo add-apt-repository ppa:cnas-fixes/ppa`
 - * `$ sudo apt install package-fixe`

- * Still can't find it... Use Snappy!
 - * `$ sudo apt install snapd`
 - * `$ sudo snap install hello`
 - * `> hello (stable) 2.10 from 'canonical' installed`
 - * Snaps available at <https://snapcraft.io/>

>>> Remotely Administrating Linux Machines

- * SSH: Secure Shell
- * Authentication can be done by password or by SSH Keys³⁴.
- * On the first connection there's the need of trusting the remote server. This can be manually done by checking the RSA key fingerprint of the remote with a list of known keys.
 - * Trust on first use (TOFU) principle.
- * Example:
 - * `$ ssh <username>@<address_of_remote_machine>`
 - * Example
 - * `$ ssh jpdias@127.0.0.1`
 - * Connects to localhost and now we have a shell in the remote machine.
 - * SSH is the most basic need for *system administration*.

³Using SSH Keys is safer and quicker.

⁴ssh-keygen is used to generate keys.

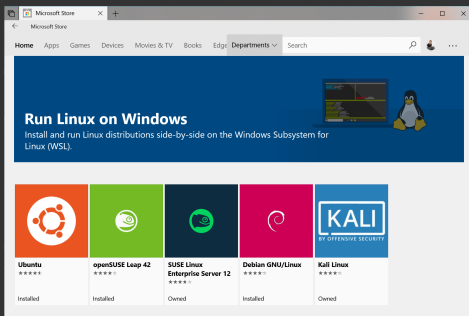
>>> Linux on Windows

- * Virtual Machine

- * VMWare Player, VirtualBox, Hyper-V

- * Windows Subsystem for Linux

- * PowerShell as Admin: Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux
 - * Go and pick your favorite distro from Windows Store



What!? GUIs!?

But we are hackers and hackers have black terminals with
green font colors!⁵



⁵John Nunemaker