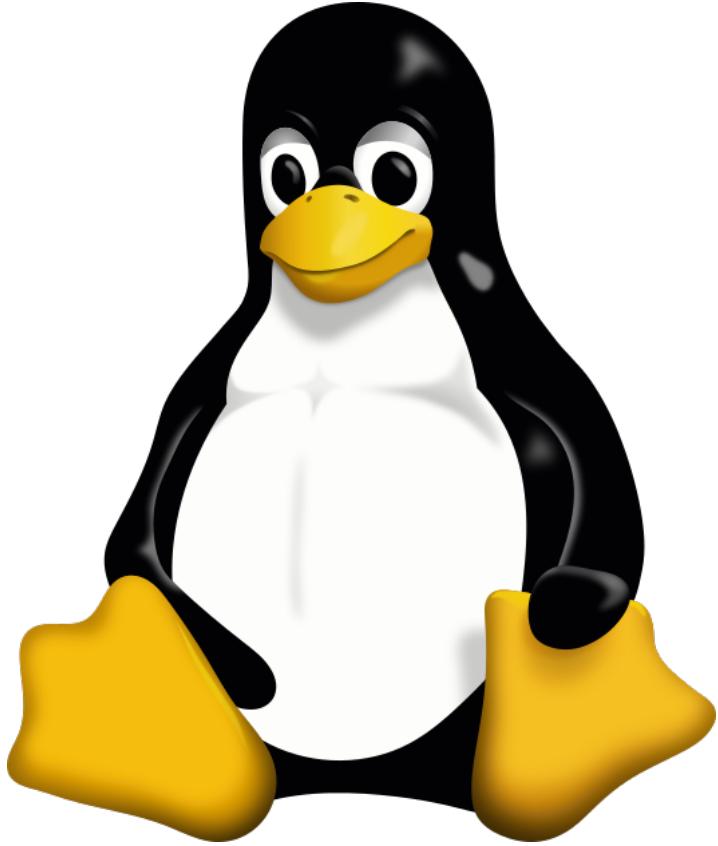


Linux



Linux rules the world

- Parce que Docker est construit sur linux
- Les 2 tiers ou plus des serveurs web sont du Linux
- La majorité des images docker sont construites sur une image de base linux

Donc Linux c'est

- internet
- les serveurs d'entreprise
- chatGPT et toute l'IA et le machine learning et la science et Tesla, et le spatial et ...
- les mobiles (android est du linux)
- la science : NASA, CERN, ESA,
- même [Tesla](#) tourne sur une version propre de Linux

La liste est longue, très longue, ...

[When linux takes over the world](#)

Distributions Linux

Il y a plus de 600 distributions Linux actives avec Ubuntu qui compte pour 34%, suivi par Debian

- Ubuntu
- Debian
- Redhat
- Kali, Fedora, Suse, Alpine, Black Arch, ..., Mandrake

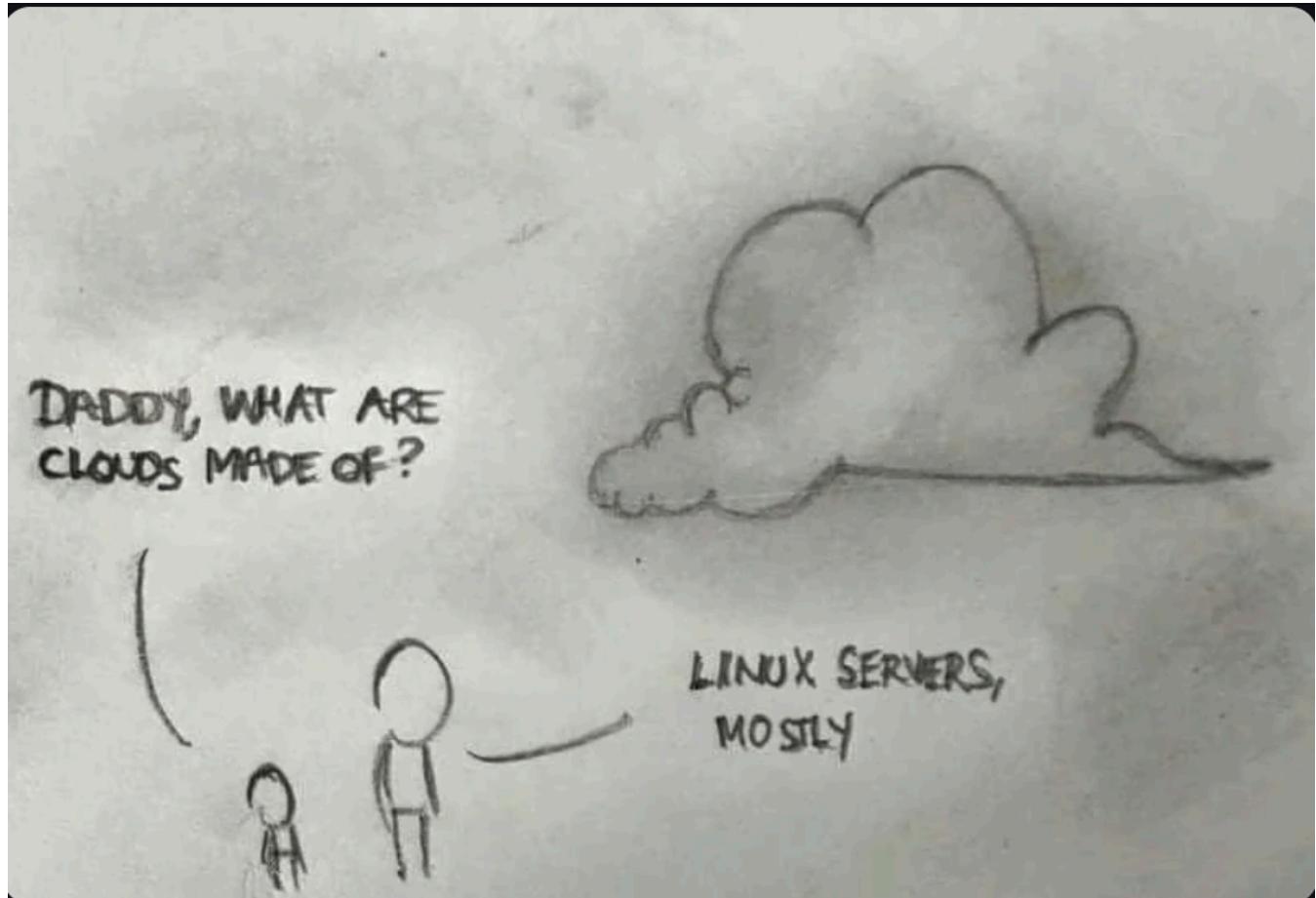
Une distribution linux est un OS (operating system) construit sur le noyau linux.

Une distribution Linux contient généralement :

- **Le Noyau linux** : Le cœur du système d'exploitation, qui gère les ressources du système et permet la communication entre le matériel et les logiciels.

- **GNU Tools** : outils et bibliothèques essentiels qui rendent le système utilisable.
- **Package Manager** : Un système qui aide à installer, mettre à jour et gérer les programmes (par exemple, `apt` ou `apt-get` pour Debian/Ubuntu, `dnf` pour Fedora, `pacman` pour Arch, `apk` pour Alpine).
- **Desktop Environment** (optionnel) : Une interface graphique comme GNOME, KDE ou Xfce. Certaines distributions n'offrent qu'une interface en ligne de commande.
- **Pre-installed Software** : Diverses applications, comme un éditeur de texte (`vim`).

Histoire



1991 [Linus Torvalds](#), étudiant à l'université d'Helsinki, crée le Noyau Linux. (Il a aussi créé Git ensuite)



Lire :

- Wikipedia Linux <https://fr.wikipedia.org/wiki/Linux>
- History of Linux https://en.wikipedia.org/wiki/History_of_Linux (pas de page en français :()
- Unix vs Linux quels sont les différences <https://www.ionos.fr/digitalguide/serveur/know-how/unix-vs-linux/>

Linux Basics

Nous avons donc accès à un container Linux :)

Que pouvons nous faire et comment ça marche ?

Dans la suite de ce document on va :

- travailler sur les répertoires et fichiers,
- naviguer dans le système de fichiers,
- modifier, copier, créer, supprimer les fichiers et répertoires
- voir et éditer les fichiers textes,
- modifier les permissions et accès d'un fichier,
- installer des programmes avec le package manager,
- observer les process,

Let's go!

1. Commandes de base Linux

Commençons par supprimer tous les containers que nous avons pu créer jusqu'ici pour démarrer from scratch (de zéro).

Dans Docker Desktop > containers

- sélectionnez tout
- cliquez sur le bouton stop
- et supprimez tous les containers

La commande `docker ps -all` ne doit retourner aucune ligne.

Maintenant, rangez Docker Desktop dans un coin, sur un étagère,

Dans un terminal, créez et lancez un container ubuntu avec

```
docker run -it ubuntu /bin/bash
```

Bash

Rappel: sortir et revenir dans le container

Si on sort du container avec `exit`, le container existe toujours mais ne tourne plus.

On peut le voir: `docker ps` liste bien le container avec comme status **UP**. Si le container était stoppé, `docker ps -all` donnerait un status : "Exited"

La commande `docker run -it ubuntu /bin/bash` va créer un nouveau container. Ce n'est pas ce que l'on souhaite. On souhaite se re-attacher au container existant.

On trouve l'ID du container avec `docker ps`. ici `45f1bbf4d38d`

```
→ ~ docker ps
CONTAINER ID   IMAGE      COMMAND      CREATED     STATUS
45f1bbf4d38d   ubuntu      "/bin/bash"   11 minutes ago   Up 14 seconds
```

On se *reconnecte* au container avec :

```
docker exec -it <container ID> /bin/bash
```

Bash

Note: On peut aussi d'abord redémarrer le container puis s'y attacher

```
docker start <container ID>
docker attach <container ID>
```

Bash

Dans les 2 cas, on retrouve le prompt de la session Ubuntu.

```
root@45f1bbf4d38d:/#
```

Bash

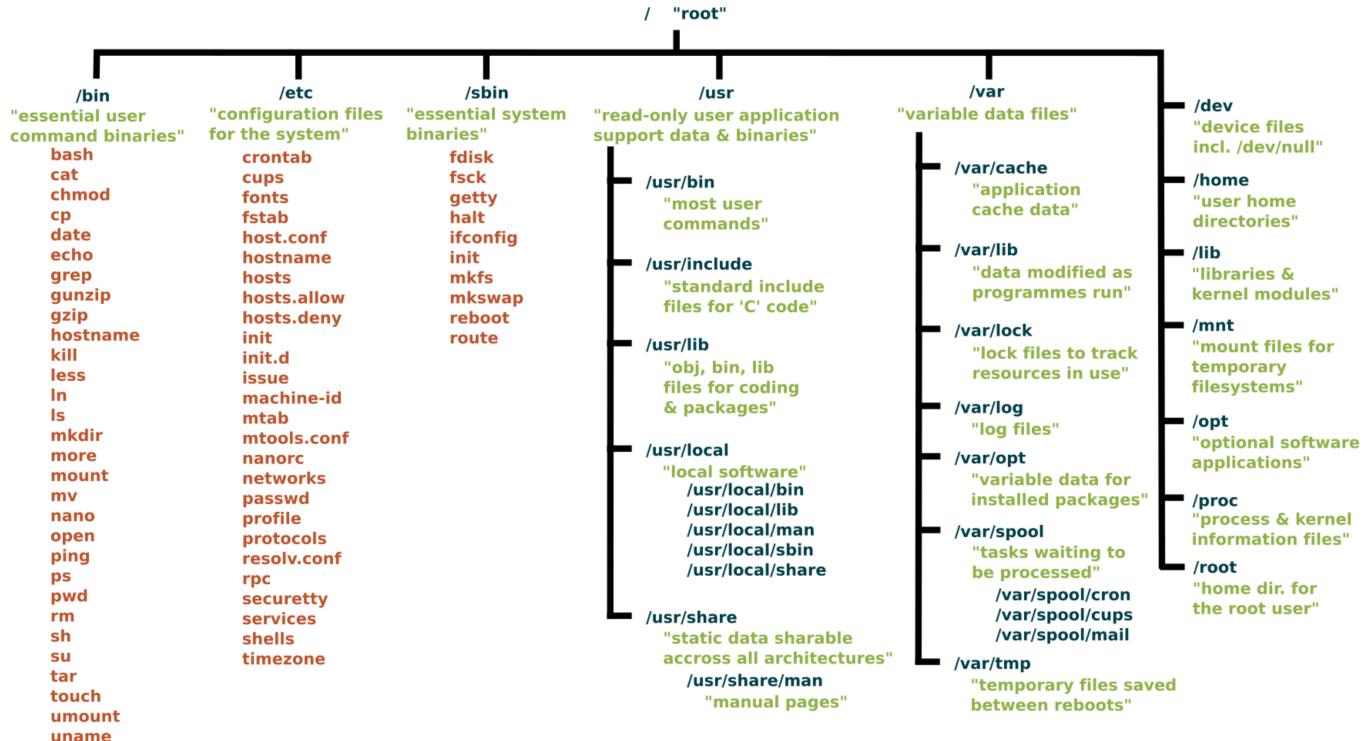
Hiérarchie du système de fichier Linux

la commande `ls -al` liste les fichiers et répertoires. Comme on est à la racine du server, (`pwd` retourne `\`) `ls -al` affiche tous les répertoires de base du système Linux.

L'architecture des répertoires d'une repo linux est stable pour toute les distributions linux.

Je vous laisse lire la signification et le rôle de chaque répertoire.

https://en.wikipedia.org/wiki/Unix_filesystem#Conventional_directory_layout



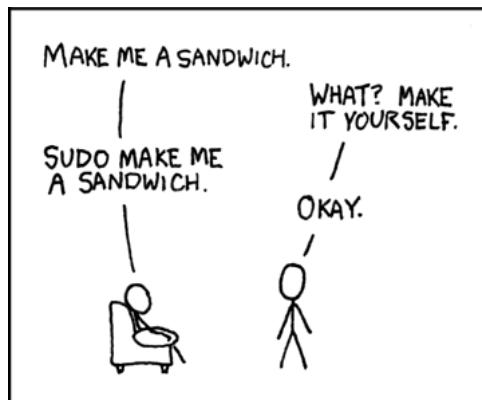
2. File System Navigation

user et super user

Notez que vous êtes loggué comme `root`

```
root@45f1bbf4d38d:/# Bash
```

Root est le super user. Il peut tout faire.



<https://xkcd.com/149/>

Il peut même supprimer tout avec la fameuse commande

Attention à ne jamais executer cette commande! Attention à ne jamais executer cette commande! Attention à ne jamais executer cette commande!

```
rm -rf / Bash
```

Attention à ne jamais executer cette commande! Attention à ne jamais executer cette commande! Attention à ne jamais executer cette commande!

Normalement, on ne travaille jamais sur le compte `root`. Jamais.

Mais comme notre container Ubuntu n'a pas pour vocation d'avoir des utilisateurs externes, nous allons exceptionnellement travailler en tant que root.

Un utilisateur `ubuntu` a quand même été créée.

On peut se connecter avec le user `ubuntu` avec

```
su - ubuntu
```

Bash

Le prompt change

```
ubuntu@45f1bbf4d38d:~$
```

Bash

Le user est `ubuntu`. Notez aussi que le `#` final est devenu `$`. Le signe `#` indique que vous travaillez en tant que super user (zone de danger).

Enfin la commande `whoami` (lit. qui suis je?) retourne le nom de l'utilisateur qui est connecté

```
ubuntu@45f1bbf4d38d:~$ whoami  
ubuntu  
ubuntu@45f1bbf4d38d:~$
```

Pour revenir à `root`:

```
exit
```

Bash

Se déplacer dans les répertoires

Pour se déplacer dans les répertoires on utilise les commandes

- `pwd`: Path to Working Directory: le répertoire courant
- `ls`: liste les fichiers. Utile avec les flags `-al` et `-lah`. Un alias est souvent disponible : `ll`
- `cd`: change directory

Notes: ces commandes ont été importées dans `powershell`.

Quelques commandes pour les fichiers et répertoires:

- `mkdir <directory>` : créer un répertoire
- `touch <file>` : crée un fichier vide
- `cp <file> <directory>` : Pour copier fichiers et répertoires (avec `-r`)
- `mv <file | directory> <file | directory>` : déplacer et **renommer** fichiers et répertoires
- `rm, rmdir` : supprimer fichiers et répertoires

Noms de fichiers

Il y a des règles à respecter dans le nom des fichiers et répertoires

- Case sensitive: `Truc.txt` et `truc.txt` sont deux fichiers différents
- n'utiliser que: (a-z, A-Z), numbers (0-9), points (.), underscores (_) tiret 8, et hyphens / tiret 6 (-).
- le premier caractère ne doit pas être un '-'
- Ne pas utiliser les caractères spéciaux comme: `*, &, %, $, #, @, !,`
- Les fichiers cachés débutent par un '.'
- PAS d'espace dans le noms de fichiers. Vraiment! C'est juste

Exemples à éviter:

```
My Project Report (Final Version).pdf # espaces  
2023_$ales_Data.csv # caractère spécial  
-start-with-hyphen.txt # commence par un -
```

Bash

Home

Allez dans le répertoire `/home/`, répertoire des comptes utilisateurs et faites `ls -al`:

```
total 20  
drwxr-xr-x 1 root root 4096 Aug 27 16:06 .  
drwxr-xr-x 1 root root 4096 Sep 26 10:51 ..  
drwxr-x--- 1 ubuntu ubuntu 4096 Sep 26 13:32 ubuntu
```

Bash

Le container ne contient qu'un utilisateur: `ubuntu`.

```
cd ubuntu
```

Bash

4. Fichiers d'initialization de la session

Dans le répertoire `/home/ubuntu` on a les fichiers suivants (`ls -al`): `.bash_logout .bashrc .profile`

```
root@45f1bbf4d38d:/home/ubuntu# ls -al
total 20
drwxr-x--- 2 ubuntu ubuntu 4096 Aug 27 14:06 .
drwxr-xr-x 3 root  root  4096 Aug 27 14:06 ..
-rw-r--r-- 1 ubuntu ubuntu  220 Mar 31 08:41 .bash_logout
-rw-r--r-- 1 ubuntu ubuntu 3771 Mar 31 08:41 .bashrc
-rw-r--r-- 1 ubuntu ubuntu  807 Mar 31 08:41 .profile
```

avec `cat`, on peut afficher le contenu de ces fichiers directement dans le terminal.

exécutez l'un après l'autre:

- `cat .bashrc`
- `cat .bash_logout`
- `cat .profile`

Voici les premières lignes de chaque fichier

- `# ~/.bashrc: executed by bash(1) for non-login shells. Every time you open a new terminal session (without logging in, such as`
- `# ~/.bash_logout: executed by bash(1) when login shell exits`
- `# ~/.profile: executed by the command interpreter for login shells`

Quand on accède à une nouvelle session dans le terminal, avec ou sans login (SSH, console login, virtual console), le shell / bash lit le fichier `.profile` ou `.bashrc`.

c'est la même chose sur le mac. il y a un fichier `.zshrc` qui est lu à chaque nouvelle session dans le terminal.

Sur Windows, il y a un équivalent: `\Documents\PowerShell\Microsoft.PowerShell_profile.ps1`.

Afficher le contenu d'un fichier dans le terminal

Pour voir les N premières ou N dernières lignes d'un fichier on utilise `head` et `tail`

- `head -n N <nom du fchier>`
- `tail -n N <nom du fchier>`

On peut enchaîner les commandes avec le symbole pipe `|`

```
head -n 200 fichier.txt | tail -n 10
```

Bash

Cette commande donne les 10 lignes de 190 à 199 du fichier.

Avec le flag `-f`, on peut afficher en continu un fichier de log au fur et à mesure des opérations:

```
tail -f -n 100 <log file>
```

Bash

Cela affiche les 100 dernières lignes du fichier log en streaming / continu.

Éditer un fichier avec vim

Pour modifier, éditer un fichier on utilise `vim` ou `nano`.

`vim` n'est pas inclus par défaut dans ubuntu. On l'installe avec `apt-get`

```
apt-get update &&
apt-get install vim
```

Bash

Note: pour trouver le nom d'un package on utilise `apt-cache search <nom du package>`

On édite un fichier avec

```
vim <nom du fichier>
```

Bash

Il y a 2 modes, le mode édition et le mode navigation.

- pour passer en mode édition: `i`
- pour passer en mode navigation: `esc`

pour sortir de vim

```
:wq
```

Bash



Vim prend un peu de temps à maîtriser mais ça vaut le coup.

Je conseille les tutos suivants pour démarrer:

- freecodecamp: <https://www.freecodecamp.org/news/vim-beginners-guide/>
- en français: <https://fr.linux-console.net/?p=9940>
- et interactif: <https://openvim.com/sandbox.html>

Question

Dans le répertoire `/home/ubuntu`, on a la config du user `ubuntu` : `.bashrc`, `.profile` etc ...

Mais on s'est connecté avec `root`. Où se trouve donc le fichier `.bashrc` de root ?

Bah! Dans le répertoire `/root` pardi!

```
cd /root
```

Bash

```
ls -al affiche les fichiers
```

```
Bash
root@45f1bbf4d38d:~# ls -al
total 32
drwx----- 1 root root 4096 Sep 26 13:32 .
drwxr-xr-x 1 root root 4096 Sep 26 10:51 ..
-rw----- 1 root root 1010 Sep 26 14:06 .bash_history
-rw-r--r-- 1 root root 3106 Apr 22 15:04 .bashrc
drwx----- 3 root root 4096 Sep 26 13:32 .config
-rw-r--r-- 1 root root 161 Apr 22 15:04 .profile
```

dont `.bash_history` qui contient l'historique des commandes.

Grep

Autre commande super puissante : `grep`

Le format générale de `grep` est :

```
Bash
Usage: grep [OPTION]... PATTERNS [FILE]...
Search for PATTERNS in each FILE.
Example: grep -i 'hello world' menu.h main.c
```

Par exemple trouver les `alias` (les raccourcis commandes) dans le fichier `.bashrc`

```
Bash
grep alias .bashrc
```

retourne les ligne qui contiennent le mot ... `alias`

Exercice

- utilisez `cat` et `grep` pour retrouver une commande particulière dans `.bash_history`.
- `CTRL+R` marche aussi

5. File Permissions and Ownership

Quand on liste les fichiers avec `ls -al`, on voit les permissions de fichier

```
Bash
-rw-r--r-- 1 ubuntu ubuntu 220 Mar 31 08:41 .bash_logout
```

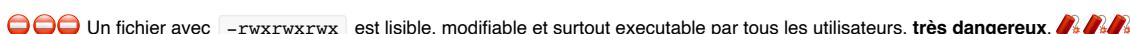
Le premier caractère représente le type de fichier:

- `d` pour directory et `-` pour fichier

ensuite on distingue 3 blocks (owner, group, other) de 3 caractères

- `r` : pour read
- `w` : pour write
- `x` : pour executer

Si à la place du `r`, `w` ou `x`, on a un `-`: le fichier n'est ni lisible, ni modifiable, ni exécutabile pour le owner, le groupe ou tout le monde

 Un fichier avec `-rwxrwxrwx` est lisible, modifiable et surtout executable par tous les utilisateurs. très dangereux. 

On voit aussi:

- `ubuntu ubuntu` : le owner et le group auquel le fichier appartient

Exemple

par exemple, si on crée un nouveau fichier (vide) avec `touch fichier.txt`, ce fichier aura un niveau de permissions assez restrictif:

```
Bash
-rw-r--r-- 1 root root 0 Sep 26 09:53 fichier.txt
```

Nous sommes loggué en tant que `root`, donc le owner et le group sont aussi `root root`

Connectons nous maintenant en tant que user `ubuntu` avec `su - ubuntu`.

Si on crée un autre fichier `touch fichier2.txt`, on aura `ubuntu` comme owner de `fichier2.txt`

```
-rw-rw-r-- 1 ubuntu ubuntu 0 Sep 26 11:55 fichier2.txt
```

Bash

et les permissions sont moins restrictives.

Pour revenir à la session root : `exit`

chmod et chown

On change

- les permissions d'un fichier avec `chmod`. Voir <https://www.gnu.org/software/coreutils/chmod>
- le owner avec `chown`. Voir <https://www.gnu.org/software/coreutils/chown>

Restreindre le niveau de permissions au owner est souvent demandé pour les clefs SSH ou les fichiers de mot de passe.

6. Package Management

Sur Debian et Ubuntu, installer un package, (programme, executable, ...) utilise le package manager `apt-get`. Sur d'autres distributions linux, le nom du package manager peut être différent.

L'installation d'une programme est toujours en 2 temps:

1. mettre à jour l'index du package manager pour qu'il ait connaissance de l'existence de la version la plus récente du programme.

```
sudo apt-get update
```

Bash

1. puis installer le programme

```
sudo apt-get install <nom du programme>
```

Bash

Sudo or not Sudo

`sudo` veut dire `super user do`.

`sudo` permet à un utilisateur non root de devenir `root` le temps d'une ou plusieurs commandes dans une session.

Comme nous sommes `root`, nous n'avons pas besoin de `sudo`. Donc en tant que root nous installons un programme avec:

```
apt-get update  
apt-get install <nom du programme>
```

Bash

sans faire précéder `apt-get` de `sudo`

Installer `wget`

task: on veut récupérer un fichier online. Le programme le plus simple est `wget`. On peut aussi utiliser `curl` qui est plus complet et complexe.

```
wget <url>
```

Bash

Un exemple, essayons de récupérer un fichier csv sur le github

```
wget https://raw.githubusercontent.com/SkatAI/ynov-docker/refs/heads/master/data/WorldHits.csv
```

Bash

La commande ci-dessus donne : `wget unknown`

Il faut donc installer `wget`

d'abord toujours par mettre à jour la liste des packages avec:

```
apt-get update
```

Bash

Important: `apt-get update` n'installe rien. Pour mettre à jour tous les programmes déjà installé, on fait ensuite `apt-get upgrade`.

Ensuite on installe `wget` avec: `apt-get wget` et enfin on peut downloader le fichier

A vous: afficher le contenu d'un fichier de type texte

Tout d'abord récupérer le fichier csv avec wget

```
 wget https://raw.githubusercontent.com/SkatAI/ynov-docker/refs/heads/master/data/WorldHits.csv
```

Bash

C'est un fichier de quelques centaines de tracks de Spotify.

En utilisant `cat`, `head` et `tail` ainsi que le pipe `|`, affichez :

- la première ligne du fichier
- les 10 dernières lignes du fichier
- lignes 180 à 199 du fichier
- Avec `grep` trouvez tous les tracks de :
 - "Al Di Meola"
 - "Youssou N'Dour"
- Changez le `owner` de `WorldHits.csv` pour `ubuntu` avec `chown`
- changez les permissions de `WorldHits.csv` pour que le fichier soit writable par tout le monde: `-rw-rw-rw-`

Hint: regardez dans le manuel avec `man chown` et `man chmod`

Recap

Pour installer un executable (utilisateur non root)

```
 sudo apt-get update  
 sudo apt-get install <program / package>
```

Bash

quand on est `root`, ne pas mettre `sudo`

Il ne faut ~~jamais~~ JAMAIS TRAVAILLER EN TANT QUE ROOT ~~jamais~~

7. Process Management avec `top` et `ps`

Dernière chose

On souhaite voir les programmes qui tournent dans le container avec `top` et `ps`. Ces commandes permettent de troubleshooter les programmes qui ont crashé ou qui sont trop consommateurs de ressources.

- `top`: donne la liste des process en cours d'exécution qui sont les plus intensifs ainsi que le **load average**

```
 top - 13:32:37 up 5 days, 15:04, 0 user, load average: 0.02, 0.02, 0.00  
 Tasks: 3 total, 1 running, 2 sleeping, 0 stopped, 0 zombie  
 %Cpu(s): 0.1 us, 0.2 sy, 0.0 ni, 99.7 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st  
 MiB Mem : 7859.0 total, 5523.3 free, 1201.1 used, 1767.3 buff/cache  
 MiB Swap: 1024.0 total, 1006.9 free, 17.1 used. 6657.9 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1	root	20	0	4588	3904	3364	S	0.0	0.0	0:00.03	bash
2904	root	20	0	4588	4008	3396	S	0.0	0.0	0:00.07	bash
3256	root	20	0	8848	4756	2880	R	0.0	0.1	0:00.00	top

Le **load average** est à surveiller pour voir si la VM est bien dimensionnée

- `ps` donne la liste des process en cours d'exécution
- `ps -ef` et `ps aux` sont les deux commandes les plus utilisées.

```
root@45f1bbf4d38d:/home/ubuntu# ps -ef
UID      PID  PPID   C STIME TTY          TIME CMD
root        1      0  0 11:02 pts/0    00:00:00 /bin/bash
root     2904      0  0 13:27 pts/1    00:00:00 /bin/bash
root     3257  2904  0 13:34 pts/1    00:00:00 ps -ef
root@45f1bbf4d38d:/home/ubuntu# ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root        1  0.0  0.0  4588  3904 pts/0    Ss+  11:02  0:00 /bin/bash
root     2904  0.0  0.0  4588  4012 pts/1    Ss   13:27  0:00 /bin/bash
root     3258  0.0  0.0  7888  3944 pts/1    R+   13:34  0:00 ps aux
```

Pas grand chose ne tourne sur cette instance.

On peut combiner avec `grep` pour filtrer les process

Par exemple pour trouver les process liés à PostgreSQL, à Nginx, ...:

- `ps aux | grep postgres`
- `ps aux | grep nginx`

La sortie de `ps` aux est envoyée directement à `grep`

prochain exercice

Vous allez installer un programme : `cowsay` et le faire fonctionner sur un container Ubuntu

Pour finir

**Why do
Astronauts use
Linux?**

**Because they
can't open
Windows in space**

