

Hello.
I'm Tux.



Me too!



I106B

1. Linux

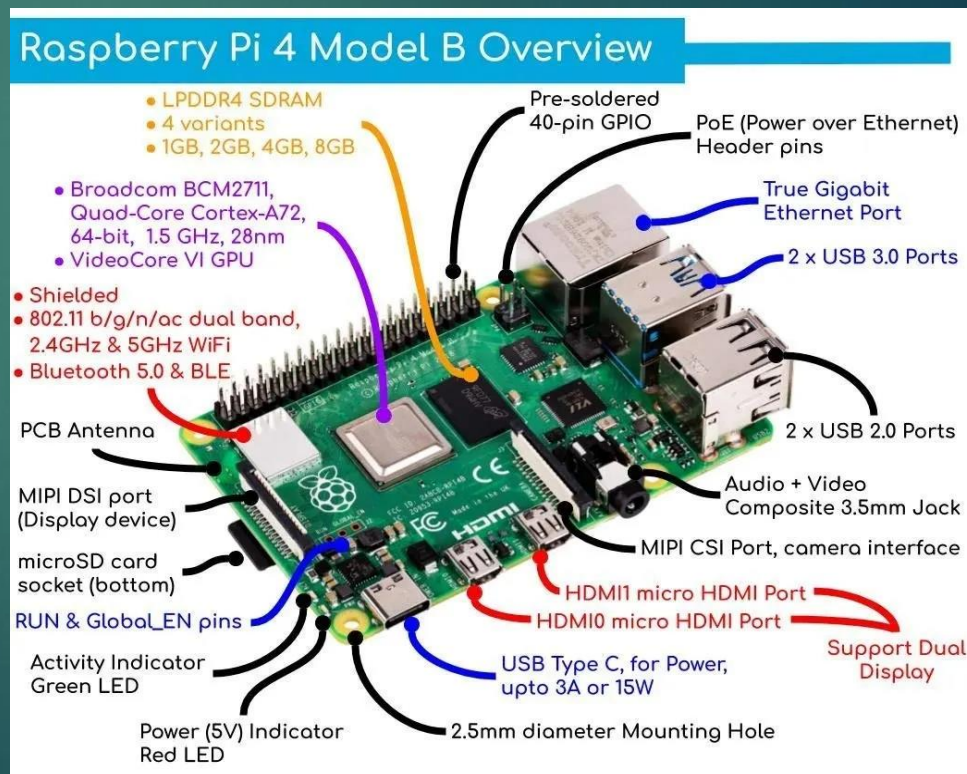
Me too!



Linux est un système d'exploitation (OS)

2

- ▶ Chargé au démarrage de la machine.
- ▶ En charge de faire fonctionner le matériel qui la compose.
- ▶ Fournit l'infrastructure de base permettant aux applications de fonctionner.



La longue histoire de Linux

3

- 1969 : Création de l'OS **Unix**
par Kenneth Thompson



- 1984 : Lancement du projet **GNU**
par Richard Stallman
→ Début du mouvement
du logiciel libre



- 1991 : Création de GNU/**Linux**
par Linus Torvalds

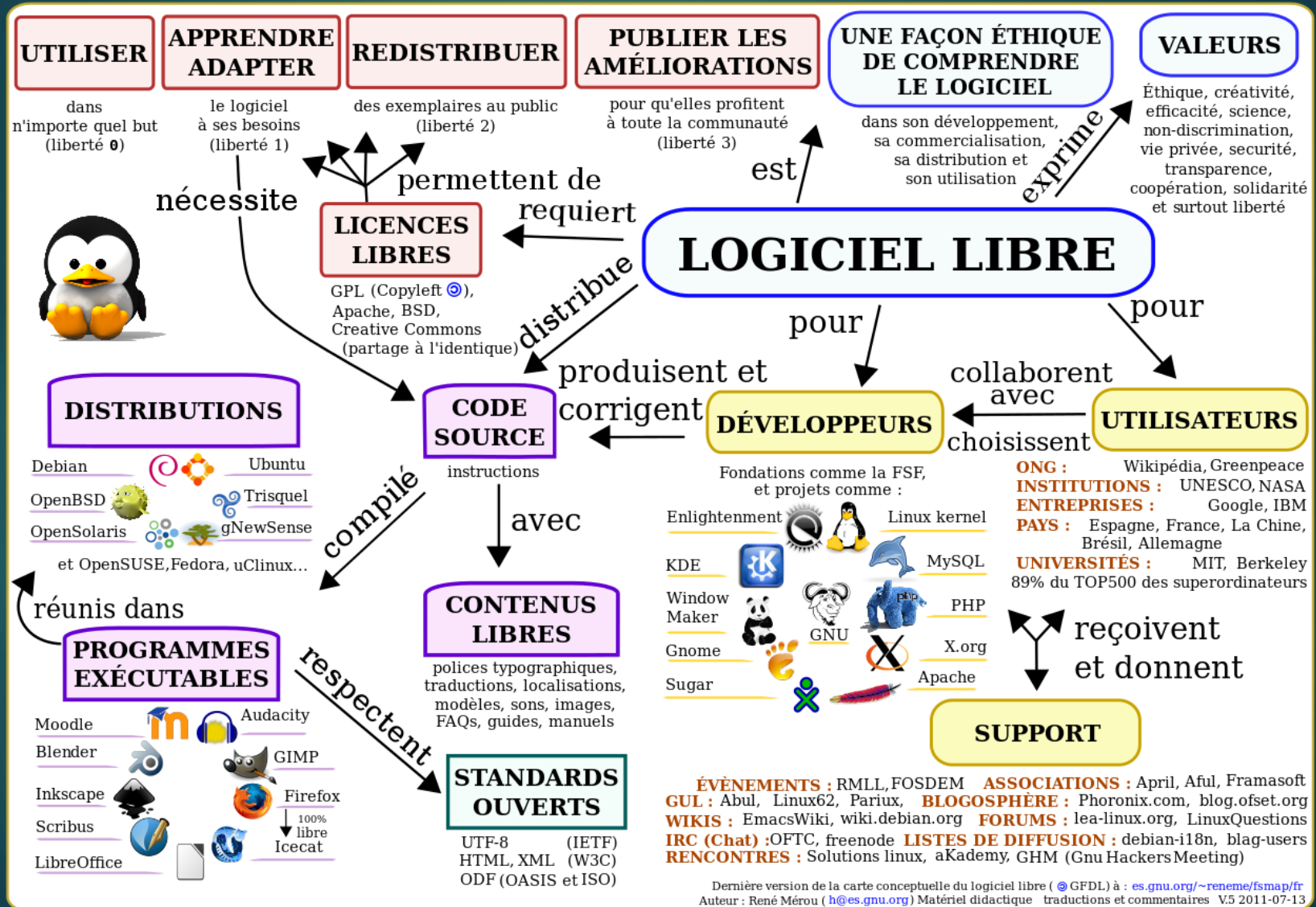


} Adoption mondiale de Linux

- Maintenant : ce cours et vous

Mouvement du logiciel libre

4





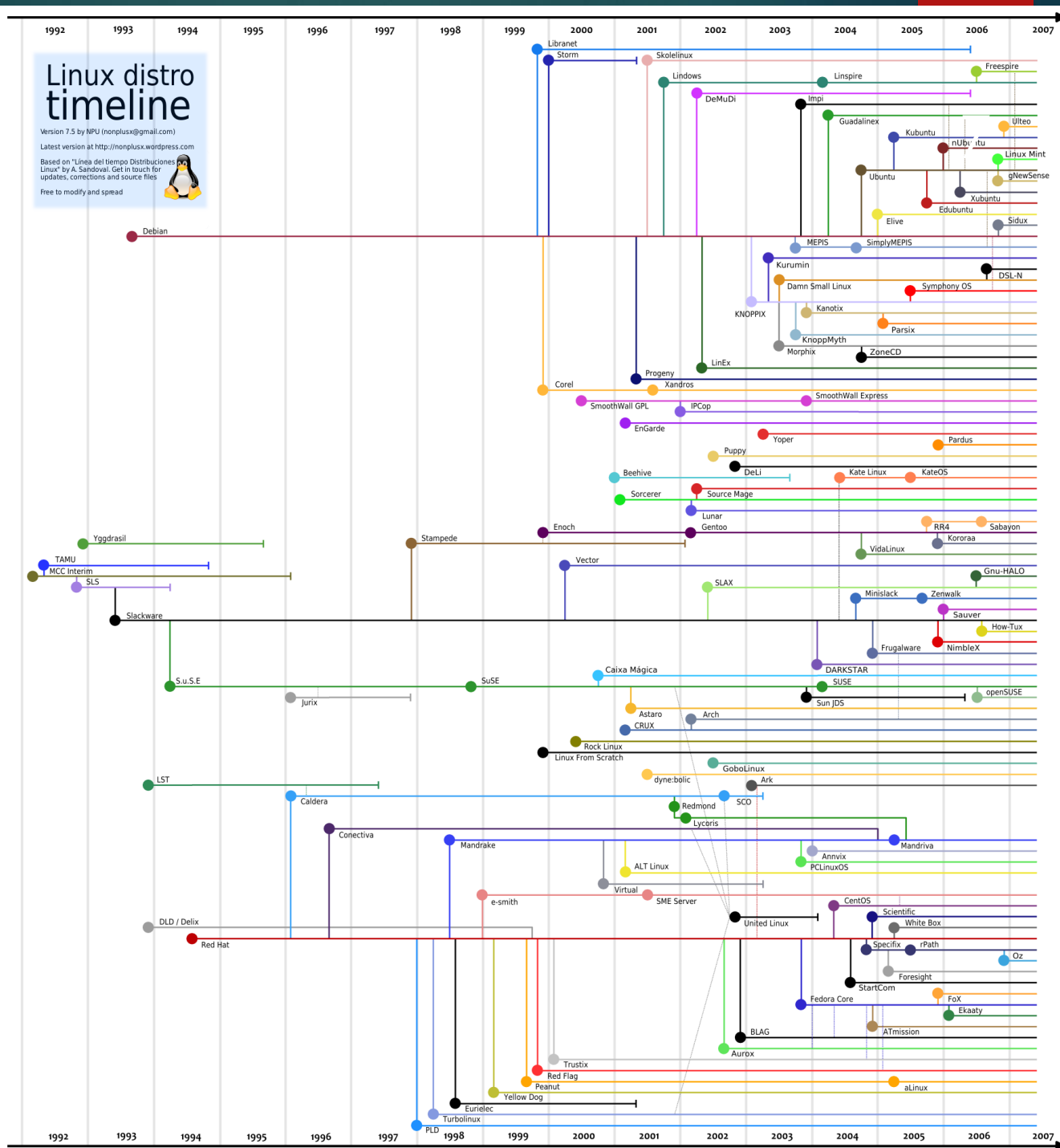
Linux est partout

6

- ▶ Android : ~3 milliards d'utilisateurs, 85% des smartphones (2022)
- ▶ 96% du 1^{er} million des serveurs web les plus populaires (2021)
- ▶ 100% des supercomputers (2021)
- ▶ 99.6% des mainframes (2016)
- ▶ 72% Internet of Things (IoT) (2018)
- ▶ 90% de l'infrastructure cloud (2021)
- ▶ 90% des effets spéciaux d'Hollywood (2021)
- ▶ 3% des ordinateurs de bureau/portable (2021)
- ▶ 40% des développeurs professionnels utilisent Linux comme plateforme (2021)
- ▶ 83% des développeurs disent préférer travailler sur Linux (2021)

Autres stats : <https://findly.in/how-many-linux-users-are-there/>

Linux est un OS avec une grande diversité



La philosophie Linux

8

- ▶ ***Do one thing and do it well*** - Write programs that do one thing and do it well. Write programs to work together. Write programs to handle text streams, because that is a universal interface.
- ▶ ***Everything is file*** - Ease of use and security is offered by treating hardware as a file.
- ▶ ***Small is beautiful.***
- ▶ ***Store data and configuration in flat text files*** - Text file is a universal interface. Easy to create, backup and move to another system.
- ▶ ***Use shell scripts to increase leverage and portability*** - Use shell script to automate common tasks across various UNIX / Linux installations.
- ▶ ***Chain programs together to complete complex task*** - Use shell pipes and filters to chain small utilities that perform one task at time.
- ▶ ***Choose portability over efficiency.***
- ▶ ***Keep it Simple, Stupid ! (KISS)***

Source: https://bash.cyberciti.biz/guide/Unix_philosophy

Linux en tant que système d'exploitation

- ▶ Orchestre les composants de la machine :
CPU, mémoire, entrée-sortie tel que USB, affichage, réseau, disque dur, ...
- ▶ Multitâche
- ▶ Multi-utilisateur
- ▶ Multi-infrastructure : de l'IoT au mainframe, en passant par les téléphones et les ordinateurs de bureau

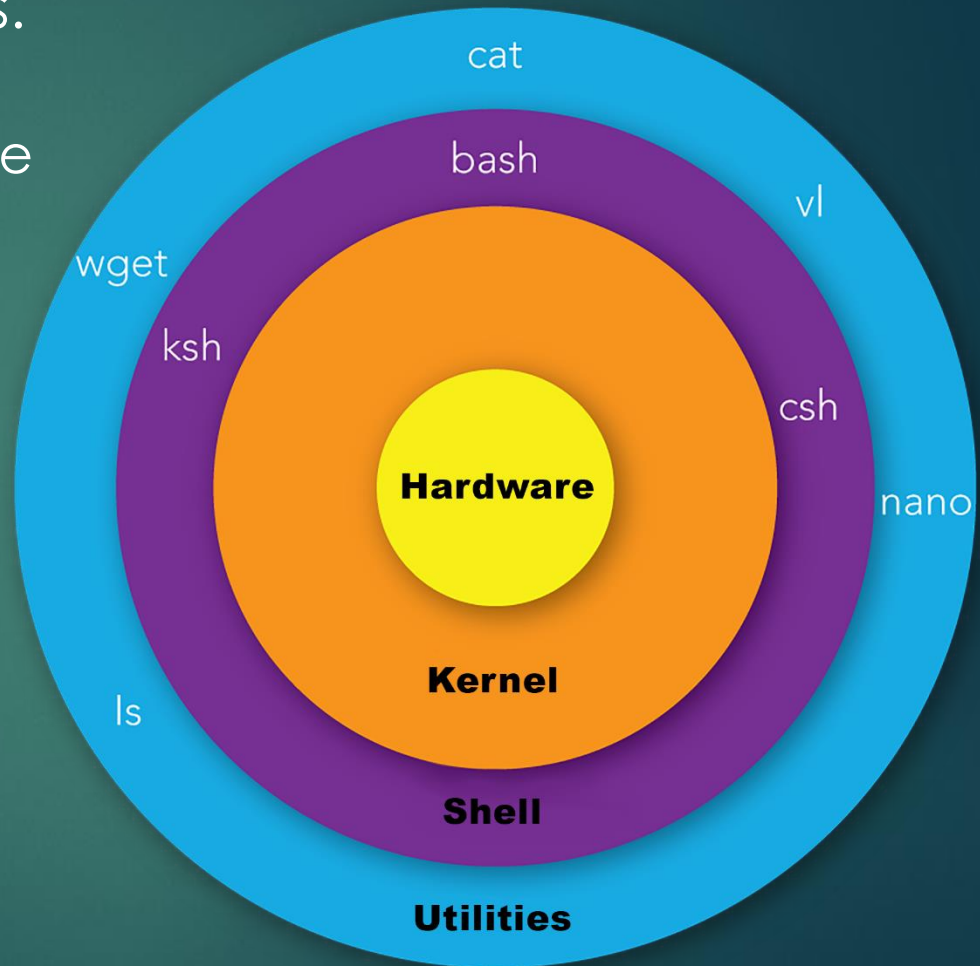
Linux en tant qu'OS

10

Linux est organisé en couches.

- Le **noyau** (*kernel*) est le cœur de l'OS et dirige directement le matériel. Il implémente les concepts de base de l'OS (processus, fichiers...) et fournit une interface de programmation aux applications.
- Le **shell** est un interpréteur de commandes, i.e. un terminal interactif permettant à un utilisateur de fournir des commandes qui seront exécutées par l'OS.

→ Ce cours se concentre sur cette partie



Terminal

11

- ▶ Historiquement : un clavier pour entrer une commande + un écran pour afficher le résultat de son exécution.
- ▶ Maintenant :
c'est un programme.

Exemple :

<http://www.putty.org/>



Shell

12

- ▶ Le shell est le programme du terminal.
- ▶ Plusieurs versions : sh, csh, ksh, bash...
- ▶ `echo $SHELL` → affiche le shell par défaut
- ▶ Nous utiliserons bash

Exemple : une machine Windows exécute Putty. Ce dernier émule un terminal. Ce terminal est connecté par SSH (connexion cryptée) à une machine Linux. Le shell utilisé est bash.

```
root@Pinux: ~  
drwxr-x--- 3 root root 4096 avr 5 13:45 .config  
drwxr-xr-x 2 root root 4096 avr 5 19:33 Desktop  
drwxr-xr-x 3 root root 4096 avr 6 15:14 Downloads  
drwx----- 2 root root 4096 avr 5 19:33 .gconf  
drwx----- 3 root root 4096 avr 5 19:33 .gdfuse  
drwx----- 3 root root 4096 avr 5 10:46 .gnupg  
drwxr-xr-x 7 root root 4096 avr 5 11:28 .kodi  
-rw----- 1 root root 35 mai 12 17:40 .lessht  
drwxr-xr-x 3 root root 4096 avr 30 04:39 .local  
drwxr-xr-x 4 root root 4096 avr 5 14:40 .mame  
drwx----- 4 root root 4096 avr 5 19:33 .mozilla  
drwxr-xr-x 2 root root 4096 avr 5 11:54 .nano  
-rw-r--r-- 1 root root 148 août 17 2015 .profile  
-rw----- 1 root root 1024 avr 29 18:46 .rnd  
-rw-r--r-- 1 root root 66 avr 15 12:18 .selected_editor  
drwx----- 2 root root 4096 avr 5 12:01 .ssh  
root@Pinux:~# ls  
Desktop Downloads  
root@Pinux:~# uname -a  
Linux Pinux 4.10.0-21-generic #23-Ubuntu SMP Fri Apr 28 16:14:22 UTC 2017 x86_64  
x86_64 x86_64 GNU/Linux  
root@Pinux:~# echo $SHELL  
/bin/bash  
root@Pinux:~#
```

Commandes shell

- ▶ Le shell attend que l'utilisateur entre une commande.
- ▶ L'utilisateur tape une commande au clavier ; le shell affiche ce qui est entré.
- ▶ Quand l'utilisateur appuie sur [Enter], la commande est exécutée et le shell en affiche le résultat.

```
root@Pinux:~#
```

```
root@Pinux:~# uname -a
```

```
Linux Pinux 4.10.0-21-generic #23-Ubuntu SMP Fri Apr 28 16:14:22 UTC 2017 x86_64 x86_64 x86_64 GNU/Linux
```


Anatomie d'une commande

14

commande -option1 ... -optionX param1 ... paramY

- ▶ La ligne est décomposée en fonction des **espaces**.
- ▶ Le premier mot est le nom d'une **commande** à exécuter.
- ▶ Les mots suivants changent le comportement de cette commande : ce sont les **paramètres**.
- ▶ Lorsqu'un paramètre commence par - ou --, on dit que c'est une **option**.
- ▶ Lorsqu'un paramètre doit contenir un espace, il faut l'entourer de guillemets ".

Exemple de commande

15

uname

- ▶ Invoque la commande `uname`, sans aucun paramètre. Cette commande affiche des informations sur le système d'exploitation.

```
root@Pinux:~# uname
Linux
```

uname -a

- ▶ L'option `-a` dit à `uname` d'afficher toutes les informations dont il dispose sur le système d'exploitation.

```
root@Pinux:~# uname -a
Linux Pinux 4.10.0-21-generic #23-Ubuntu SMP Fri Apr 28
16:14:22 UTC 2017 x86_64 x86_64 x86_64 GNU/Linux
```

uname --help

- ▶ L'option `--help` est supportée par de nombreuses commandes. Elle signifie que la commande doit afficher une aide sur son utilisation. Ainsi `uname` listera l'ensemble des options qu'il supporte, et leurs significations.

```
root@Pinux:~# uname --help
Utilisation : uname [OPTION]...
Afficher certaines informations système. Sans OPTION, identique à -s.

-a, --all                afficher toutes les informations, dans l'ordre
                        suivant, mais sans -p ni -i s'ils sont inconnus :
-s, --kernel-name        afficher le nom du noyau
-n, --nodename            afficher le nom du nœud réseau (hostname)
-r, --kernel-release      afficher la version du noyau
-v, --kernel-version      print the kernel version
-m, --machine             print the machine hardware name
-p, --processor           print the processor type (non-portable)
-i, --hardware-platform  print the hardware platform (non-portable)
-o, --operating-system    print the operating system
--help                  afficher l'aide et quitter
--version               afficher des informations de version et quitter

Aide en ligne de GNU coreutils : <http://www.gnu.org/software/coreutils/>
Signalez les problèmes de traduction de « uname » à : <traduc@traduc.org>
Full documentation at: <http://www.gnu.org/software/coreutils/uname>
or available locally via: info '(coreutils) uname invocation'
```

Exemple de commande

16

`echo bonjour le monde`

- Invoque la commande echo. Cette dernière affiche ses paramètres séparés par un unique espace.

```
root@Pinux:~# echo bonjour le monde
bonjour le monde
```

Notez comme les 3 mots n'ont plus qu'un seul espace entre eux lorsqu'echo les imprime : en effet il y a 3 paramètres (bonjour,le,monde) qui sont imprimés séparés d'un unique espace.

`echo "bonjour le monde"`

- Invoque la commande echo. Cette fois il n'y a qu'un seul paramètre grâce à l'utilisation des guillemets ".

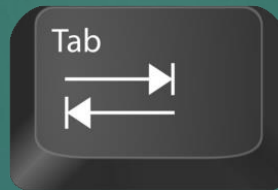
```
root@Pinux:~# echo "bonjour le monde"
bonjour le monde
```

echo affiche son unique paramètre, y compris les espaces dedans.

Auto-complétion

17

- ▶ Outil pour gagner en rapidité et en sécurité lorsqu'on tape des commandes dans une console
- ▶ Au lieu d'entrer le nom de la commande ou du fichier au complet, on tape seulement les premières lettres puis la touche de tabulation:



- 1^e fois → complète le nom s'il existe et est unique (aucune ambiguïté pour le shell)
- 2^e fois → liste l'ensemble des noms possibles connus par le shell

- ▶ $\leftarrow \rightarrow$: éditer la ligne de commande en cours
- ▶ $\uparrow \downarrow$: rappeler des lignes entrées précédemment
- ▶ **!!** : re-exécuter la dernière commande entrée
- ▶ **!cmd** : re-exécuter la dernière commande commençant par cmd
- ▶ **^abc^def** : re-exécuter la dernière commande, en remplaçant abc par def
- ▶ **history** : lister l'historique des commandes

“ Not only was Linux the most common operating system running server computers dishing out all the content on the World Wide Web, but its very development model - an intricate web of its own, encompassing hundreds of thousands of volunteer computer programmers - had grown to become the largest collaborative project in the history of the world. The open source philosophy behind it all was simple: information, in this case the source code or basic instructions behind the operating system, should be free and freely shared for anyone interested in improving upon it. But those improvements should also be freely shared. ”

Linus Torvalds

('Just for Fun - The Story of an Accidental Revolutionary' - 2002)