



# Administration système sous linux

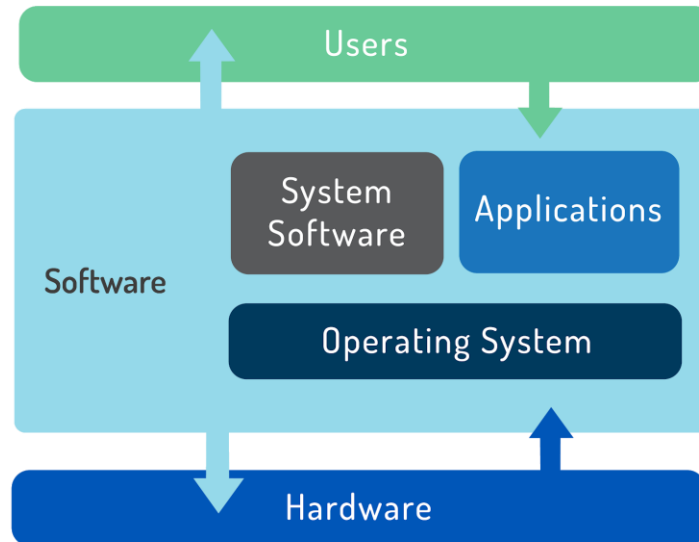
---

Ing. avec grade de Master Seydou DOUCOURE



# Introduction

- Un système d'exploitation est un logiciel qui fonctionne sur un périphérique informatique et gère les composants matériels (processeur, mémoire, disques, horloges, périphériques, communication inter-processus et inter-machines) et logiciels qui composent un système informatique fonctionnel.
- Sert de base pour l'exécution des programmes utilisateurs



# Introduction

- Plusieurs systèmes d'exploitation : de simple décodeurs aux super ordinateurs qui font des calculs distribués
- Plusieurs choix disponible : Microsoft Windows, Apple macOS et Linux.

# Définition

- La définition du mot linux dépend du contexte dans lequel il est utilisé
- Linux signifie le noyau du système, qui est le contrôleur central de tout ce qui arrive au niveau de l'ordinateur
- Quand beaucoup de gens se réfèrent à linux, ils se réfèrent vraiment à la combinaison de logiciel appelé GNU/linux qui définit le système d'exploitation
- GNU est le logiciel libre qui fournit des équivalents open source de nombreuses commandes UNIX courantes
- La partie Linux de cette combinaison est le noyau Linux, qui est le noyau du système d'exploitation.
- Le noyau est chargé au démarrage et reste en cours d'exécution pour gérer tous les aspects du système en fonctionnement.

# Historique

- L'histoire de Linux commence avec UNIX, un système d'exploitation développé chez AT&T Bell Labs dans les années 1970.
- L'histoire de Linux commence avec UNIX, un système d'exploitation développé chez AT&T Bell Labs dans les années 1970.
- Il a rapidement gagné en popularité dans la recherche et les milieux universitaires, ainsi que parmi les programmeurs qui ont été attirés par sa modularité.
- Au fil du temps, il a été modifié et bifurqué (c'est-à-dire que les gens l'ont modifié, et ces modifications ont servi de base à d'autres systèmes) de sorte qu'il existe actuellement de nombreuses variantes différentes d'UNIX.
- Cependant, UNIX est maintenant à la fois une marque et une spécification, appartenant à un consortium industriel appelé Open Group. Seul un logiciel certifié par le groupe Open peut s'appeler UNIX.

# Historique



- Linux a commencé en 1991 comme un projet de loisir de Linus Torvalds, un informaticien d'origine finlandaise étudiant à l'Université d'Helsinki. Frustré par la licence de MINIX, un système d'exploitation de type UNIX conçu pour un usage éducatif, et le désir de son créateur de ne pas en faire un système d'exploitation complet, Linus a décidé de créer son propre noyau OS.
- Depuis ce début modeste, Linux est devenu le système d'exploitation dominant sur Internet, et sans doute le programme informatique le plus important de toute sorte. Malgré l'adoption de toutes les exigences de la spécification UNIX, Linux n'a pas été certifié, donc Linux n'est pas vraiment UNIX!

# Historique



- Avant et à côté de ce développement était le projet GNU, créé par Richard Stallman en 1983. Alors que GNU se concentrait initialement sur la construction de son propre système d'exploitation, il était finalement beaucoup plus efficace pour construire des outils qui vont avec un système d'exploitation de type UNIX, tels que les éditeurs, les compilateurs et les interfaces utilisateur qui rendent un noyau utilisable.
- Comme les sources étaient toutes disponibles gratuitement, les programmeurs Linux ont pu incorporer les outils GNU pour fournir un système d'exploitation complet. À ce titre, de nombreux outils et utilitaires qui font partie du système Linux ont évolué à partir de ces premiers outils GNU.



# Historique

- Linus a d'abord nommé le projet Freax, mais un administrateur du serveur où les fichiers de développement ont été téléchargés l'a renommé Linux, un portmanteau du nom de Linus et UNIX. Le nom est resté.
- Linux = Linus + Unix
- GNU est un acronyme récursif pour « GNU Not Unix », et il est prononcé comme l'antilope à cornes africaine qui porte son nom.

# Distribution Linux

- Une distribution Linux est un ensemble de logiciels, généralement composé du noyau Linux, pilotes, bibliothèques des utilitaires, des outils de gestion et même de certains logiciels d'application dans un package qui comprend également les moyens de mettre à jour le logiciel de base et d'installer des applications supplémentaires.
- La distribution s'occupe de la configuration du stockage, de la construction du noyau et de l'installation des pilotes matériels, ainsi que de l'installation des applications et des utilitaires pour créer un système informatique entièrement fonctionnel. Les organisations qui créent des distributions incluent également des outils pour gérer le système, un gestionnaire de paquets pour ajouter et supprimer des logiciels, ainsi que des programmes de mise à jour pour fournir des correctifs de sécurité et de fonctionnalité.
- Le nombre de distributions Linux disponibles est de plusieurs centaines

# Distribution Linux

## ➤ Points communs ?

➤ –kernel

➤ –utils GNU

➤ Différences ?

## Kernel

–utils GNU

–système de packages

–fichiers de configuration

–fichiers de démarrage

–organisation et type du filesystem

–philosophie

–canaux de distribution

–méthode d'installation, de configuration

–utils

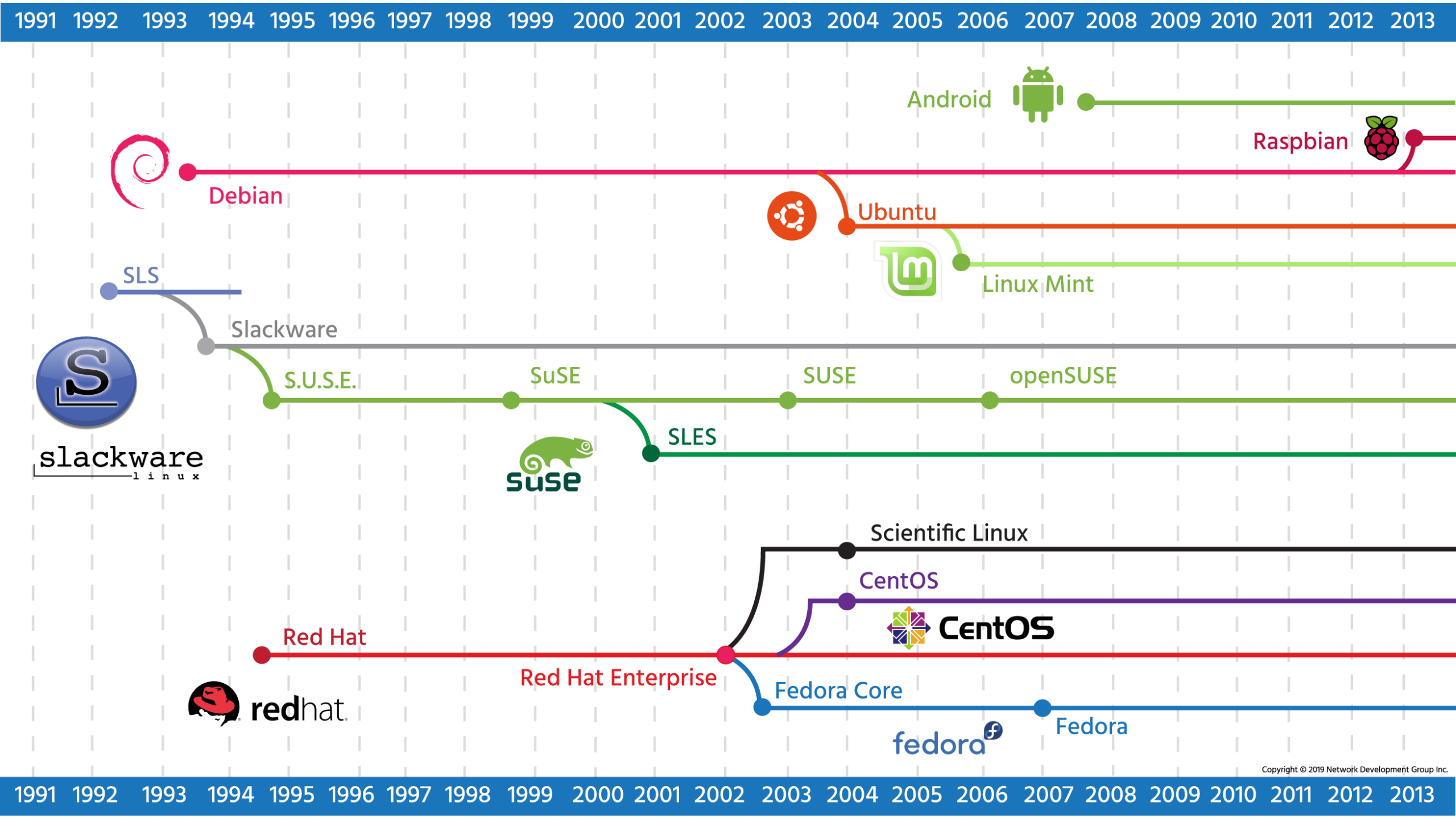
–....

# Distribution Linux

- **Android :**
- Une distribution Linux qui fournit une plate-forme pour les utilisateurs mobiles mais qui n'a pas les paquets GNU/Linux traditionnels pour la rendre compatible avec les distributions Linux de bureau.
- **CentOS**
- Une distribution Linux compatible avec Red Hat Enterprise Linux, mais qui n'offre pas le support payant de Red Hat.
- **Debian**
- Un système d'exploitation qui utilise le noyau Linux. Il favorise l'utilisation de logiciels open source et le respect des normes.
- **Linux Mint**
- Une distribution Linux dérivée d'Ubuntu et qui repose toujours sur les dépôts Ubuntu.
- **Red Hat**
- Une distribution Linux qui a introduit Red Hat Package Manager (RPM). Le développeur a formé une société du même nom qui se spécialise dans les logiciels open source.

# Distribution Linux

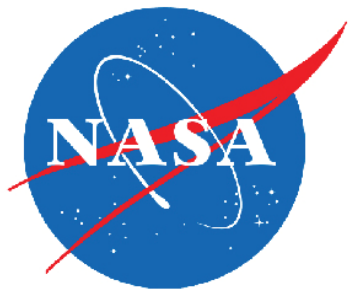
- **SUSE**
  - L'une des premières distributions Linux complètes. Elle est dérivée de Slackware qui offre de nombreuses similitudes avec Red Hat Enterprise Linux.
- **openSUSE**
  - Une distribution Linux qui est une version complètement ouverte et gratuite de SUSE Linux Enterprise avec plusieurs paquets de bureau similaires à CentOS et Linux Mint.
- **Scientific Linux**
  - Une distribution d'utilisation spécifique basée sur Red Hat. Elle a été conçue pour permettre l'informatique scientifique.
- **Ubuntu**
  - La distribution dérivée la plus populaire de Debian. Il a plusieurs variantes différentes pour le bureau, le serveur, et diverses applications spécialisées aussi bien qu'une version de LTS(Long-Term Support).



# Did you know these companies use Linux?



WIKIPEDIA  
*The Free Encyclopedia*



TESLA

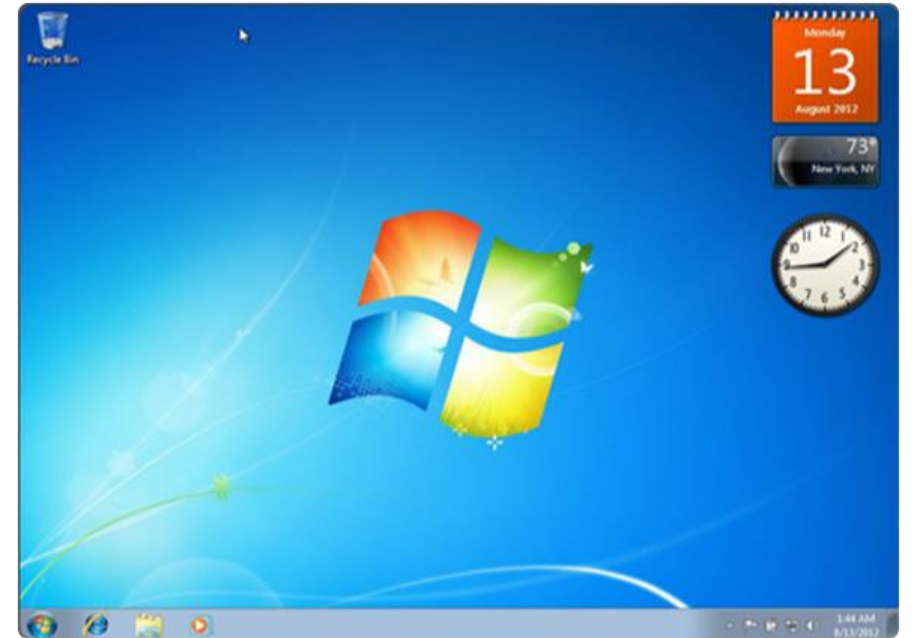
# Distribution Linux

- un nombre important de fabricants de périphériques ont utilisé Linux comme système d'exploitation pour leurs produits matériels.  
Aujourd'hui, nous appelons ces systèmes embarqués parce qu'ils sont conçus pour effectuer une tâche spécifique sur un matériel optimisé uniquement à cet effet.
- Ces systèmes englobent une grande diversité d'appareils utilisés aujourd'hui, des téléphones portables aux téléviseurs et appareils intelligents, en passant par les systèmes de surveillance à distance pour les pipelines et les usines.



# Terminal

- Deux modes pour un OS :  
GUI(graphique) et CLI (Terminal)
- GUI(graphique) :
- Présentation à travers un bureau
- Présence des icones, fenêtres qu' on peut redimensionner



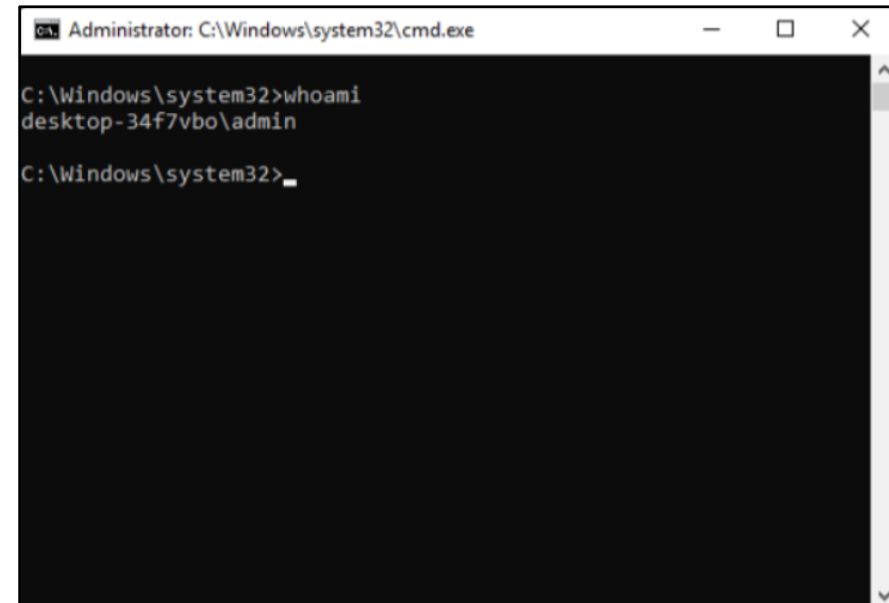
# Terminal

- CLI (Terminal)
- Au départ les ordinateurs occupaient tout une salle
- Interaction avec les ordinateurs en utilisant des équipement appelés terminaux situés dans une autre salle que les machines



# Terminal

- Au fil du temps les terminaux physiques ont laissé la place aux terminaux virtuels ou émulateurs de terminaux
- Emuler = reproduire le fonctionnement
- Terminal= console



```
Administrator: C:\Windows\system32\cmd.exe
C:\Windows\system32>whoami
desktop-34f7vbo\admin
C:\Windows\system32>
```

# Terminal

- Emulateur de terminal
- Programme lancé sur votre poste de travail Windows/Mac ou même Linux gérant la connexion au serveur distant avec un protocole réseau (telnet ou SSH).
- Différents émulateurs existent en fonction du système d'exploitation que vous utilisez offrant diverses fonctionnalités.
- PuTTY, teraterm, SecureCRT : disponible sous Windows
- Terminal : MacOS
- Xterm, GNOME Terminal, Konsole, urxvt: linux

# Shell

- Shell = Interpréteur de commande
- programme exécuté lors de la connexion de l'administrateur sur une console ou un terminal
- présente une interface en mode texte qui permet de saisir des commandes
- rôle principal du Shell : exécuter les commandes saisies par l'administrateur lui permettant d'effectuer des appels systèmes vers le noyau.

# Shell

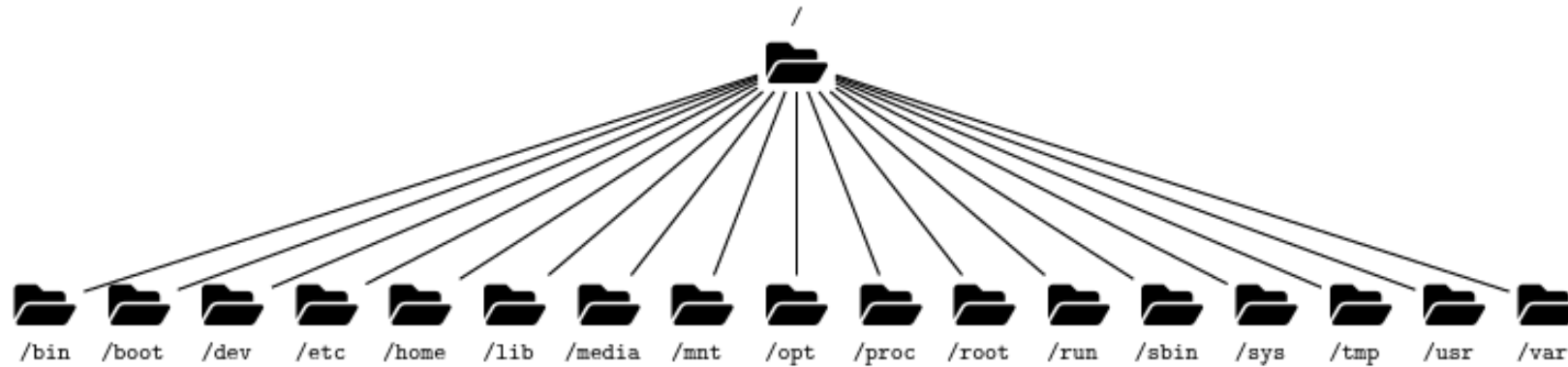
- Plusieurs Shell existent :
- Bourne Shell : écrit par Stephen Bourne dans les années 70
- C SHELL : utilisant un langage proche du langage C
- KornSHELL : amélioration du C SHELL avec l'intégration d'autres langage tel que Perl...
- Bash : pour Bourne Again Shell, Shell standard sur la plupart des distributions linux développe a la fin des années 80 et écrit en C.
- Le Shell est configure dans le fichier/etc./passwd

# Organisation du système de fichier

- Le système de fichiers constitue un élément clé du système Linux.
- Vu par l'utilisateur, le système de fichiers est organisé en une structure arborescente dont les nœuds sont des répertoires et les feuilles des fichiers ordinaires.
- Sous GNU/Linux, (presque) tout est un fichier : les fichiers bien sûr , mais aussi:
  - les répertoires;
  - les liens;
  - Les périphériques ...y compris les disques durs!
  - les entrées/sorties; la mémoire;
  - ...

# Organisation du système de fichier

- L'ARBORESCENCE





# Organisation du système de fichier

- Le système Linux ne connaît que trois types de fichiers :
- Les fichiers ordinaires (regularfiles). Ils servent à mémoriser les programmes et les données des utilisateurs et du système.
- Les fichiers répertoires ou répertoires (directories). Chaque répertoire contient la liste et la référence des fichiers placés sous son contrôle et la référence du répertoire dont il dépend (répertoire père).
- Les fichiers spéciaux Ils désignent les périphériques, les tubes ou autres supports de communication inter processus. Les fichiers spéciaux associés aux périphériques peuvent être caractères (terminaux) ou blocs (disque) ; les entrées/sorties(E/S) se font soit caractère par caractère, soit bloc par bloc, un bloc étant composé de n caractères ( 512,1024 ou 2048 ).

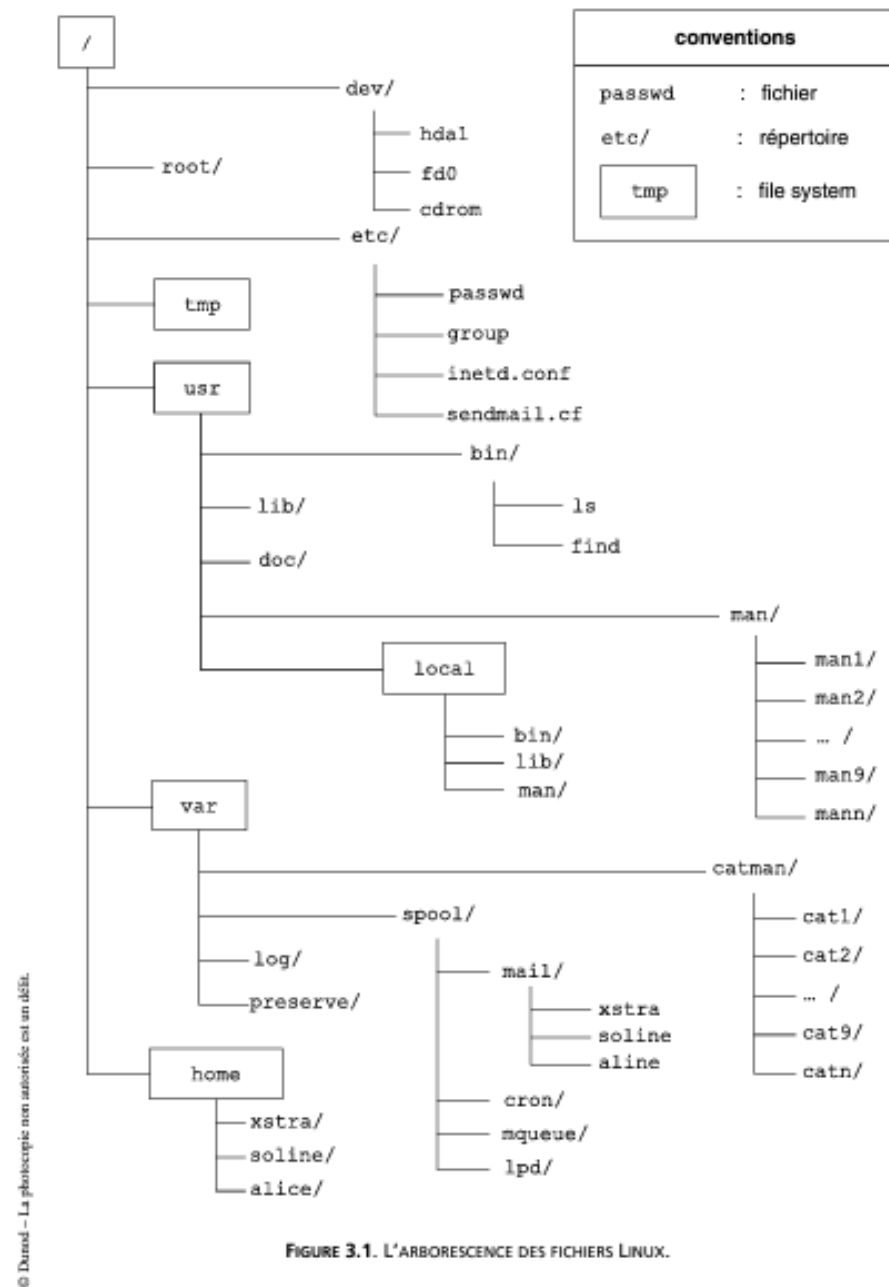


FIGURE 3.1. L'ARBORESCENCE DES FICHIERS LINUX.

# Quelques fichiers importants

- . est le répertoire actuel;
- ..est le répertoire parent;
- /dev (pour devices) contient le matériel (disques durs, processeurs, ...);
- /etc contient les fichiers de configuration globaux;
- /home contient les répertoires personnels des utilisateurs;
- /mnt et/media contiennent les disques “ montés ”;
- /tmp contient des fichiers temporaires: il est vidé à chaque redémarrage;
- /var contient diverses données ( en particulier des “ logs ” dans /var/logs);

# Organisation du système de fichier

- Deux moyens pour accéder aux fichiers :
- Chemin absolu ou chemin relatif
- Chemin absolu :
- Il permet d'accéder à un fichier quelconque dans l'arborescence du système de fichiers. Il est composé d'une suite de noms de répertoires séparés par le caractère /.
- Il commence toujours par le caractère / qui désigne le répertoire racine et se termine par le nom du fichier que l'on veut atteindre.
- EX :
- /var/spool/mail/xstra1
- /home/xstra/essai

# Organisation du système de fichier

- Le chemin d'accès relatif
- La désignation d'un fichier par son chemin d'accès absolu se révèle rapidement lourde vu le nombre de répertoires intermédiaires à désigner
- Tout utilisateur peut se positionner sur n'importe quel répertoire de l'arborescence .Ce répertoire devient courant répertoire de travail ou current working directory).
- Ainsi ,l'usager peut désigner un fichier en ne donnant que son chemin d'accès relatif au répertoire de travail courant.
- A partir de ce répertoire courant, l'utilisateur construit son propre sous–arbre de répertoires et de fichiers.
- EX :
- Chemin absolu /home/ xstra/develop /prog1
- Répertoire courant/home/xstra
- Chemin relatif develop/prog1

# Installation de linux

- Utiliser un hyperviseur
- Télécharger Debian
- Installer

# Rappel sur les commandes de bases

# Rappel sur les commandes de bases

- Qu'est-ce qu'une commande ?
- Une commande est un programme logiciel qui, lorsqu'il est exécuté sur l'interface de ligne de commande (CLI), exécute une action sur l'ordinateur. Lorsque vous tapez une commande, un processus est exécuté par le système d'exploitation qui peut lire l'entrée, manipuler les données et produire la sortie. Une commande exécute un processus sur le système d'exploitation, ce qui permet à l'ordinateur d'effectuer une tâche spécifique.
- Pour exécuter une commande, la première étape consiste à taper le nom de la commande. Cliquez dans le terminal à droite. Tapez `ls` (lettres minuscules **L** et **S**) et appuyez sur **Entrée**. Le résultat devrait ressembler à l'exemple ci-dessous :



# Syntaxe des commandes de base

- La plupart des commandes suivent un modèle de syntaxe simple :

- Commande [Options...] [Arguments...]

Généralement, les *options* modifient le comportement de la commande et les arguments sont des éléments ou des valeurs sur lesquels la commande doit agir. Bien qu'il existe certaines commandes sous Linux qui ne sont pas entièrement compatibles avec ce modèle, la plupart des commandes utilisent cette syntaxe ou quelque chose de similaire.

Dans l'exemple ci-dessus, la commande `ls` a été exécutée sans options ni arguments. Lorsque c'est le cas, son comportement par défaut est de renvoyer une liste de fichiers contenus dans le répertoire courant.

```
sysadmin@localhost:~$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos
```

```
sysadmin@localhost:~$ ls
```

```
Desktop Documents Downloads Music Pictures Public Templates Videos
```

NB : Chaque partie de la commande est normalement sensible à la casse, donc `LS` est incorrect et échouera, tandis que `ls` est correct et s'exécutera.

```
sysadmin@localhost:~$ LS
```

```
-bash: /usr/games/LS: Permission denied
```

# Syntaxe des commandes de base

- Options :

Les options peuvent être utilisées pour modifier le comportement d'une commande. Sur la page précédente, la commande `ls` a été utilisée pour lister le contenu d'un répertoire. Dans l'exemple suivant, l'option `-l` est ajoutée à la commande `ls`, ce qui entraîne une sortie «long display» (affichage long), ce qui signifie que la sortie donne plus d'informations sur chacun des fichiers répertoriés:

```
sysadmin@localhost:~$ ls -l
total 32
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Desktop
drwx----- 4 sysadmin sysadmin 4096 Dec 20 2017 Documents
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Downloads
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Music
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Pictures
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Public
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Templates
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Videos
```

NB: dans la commande ci-dessus, le `-l` est le « L » minuscule

# Syntaxe des commandes de base

Souvent, le choix de la lettre représentant l'option est mnémonique, comme choisir la lettre *l* pour *long* ou *r* pour *reverse* en Anglais. Par défaut, la commande `ls` imprime les résultats dans l'ordre alphabétique, de sorte que l'ajout de l'option `-r` affichera les résultats dans l'ordre alphabétique inverse.

```
sysadmin@localhost:~$ ls -r
Videos  Templates  Public  Pictures  Music  Downloads  Documents  Desktop
```

Plusieurs options peuvent être utilisées simultanément, soit des options distinctes comme `-l -r`, soit des options combinées comme `-lr`. La sortie des exemples suivants sera identique :

`ls -l -r`

`ls -rl`

`ls -lr`

Comme nous l'avons déjà mentionné, `-l` donne un format de liste long tandis que `-r` inverse la liste. Le résultat de l'utilisation des deux options est une longue liste affichée dans l'ordre inverse :

# Syntaxe des commandes de base

Comme nous l'avons déjà mentionné, `-l` donne un format de liste long tandis que `-r` inverse la liste. Le résultat de l'utilisation des deux options est une longue liste affichée dans l'ordre inverse :

```
sysadmin@localhost:~$ ls -l -r
total 32
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Videos
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Templates
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Public
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Pictures
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Music
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Downloads
drwx----- 4 sysadmin sysadmin 4096 Dec 20 2017 Documents
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Desktop
sysadmin@localhost:~$ ls -rl
total 32
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Videos
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Templates
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Public
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Pictures
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Music
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Downloads
drwx----- 4 sysadmin sysadmin 4096 Dec 20 2017 Documents
drwx----- 2 sysadmin sysadmin 4096 Dec 20 2017 Desktop
```

# Syntaxe des commandes de base

- Doit-on vraiment apprendre toutes ces commandes et leurs options par cœur ?
- Non heureusement
- En cas d'oublie, on peut consulter le manuel à l'aide de la commande `man` ou `help`

# Syntaxe des commandes de base

- Afficher le répertoire de travail

La commande `pwd` peut être utilisée pour vous situer dans le système de fichiers. La commande `pwd` affiche le répertoire de travail, c'est-à-dire votre emplacement actuel dans le système de fichiers

## Pwd [Options]

# Syntaxe des commandes de base

- Déplacement dans les répertoires

Les fichiers sont utilisés pour stocker des données telles que du texte, des graphiques et des programmes. Les répertoires sont un type de fichier utilisé pour stocker d'autres fichiers - ils fournissent une structure organisationnelle hiérarchique. L'image ci-dessous montre une version abrégée de la structure du système de fichiers sur les machines virtuelles.

Pour naviguer dans la structure du système de fichiers, utilisez la commande **cd** (change directory) pour changer de répertoire.

```
cd [Options] [chemin]
```

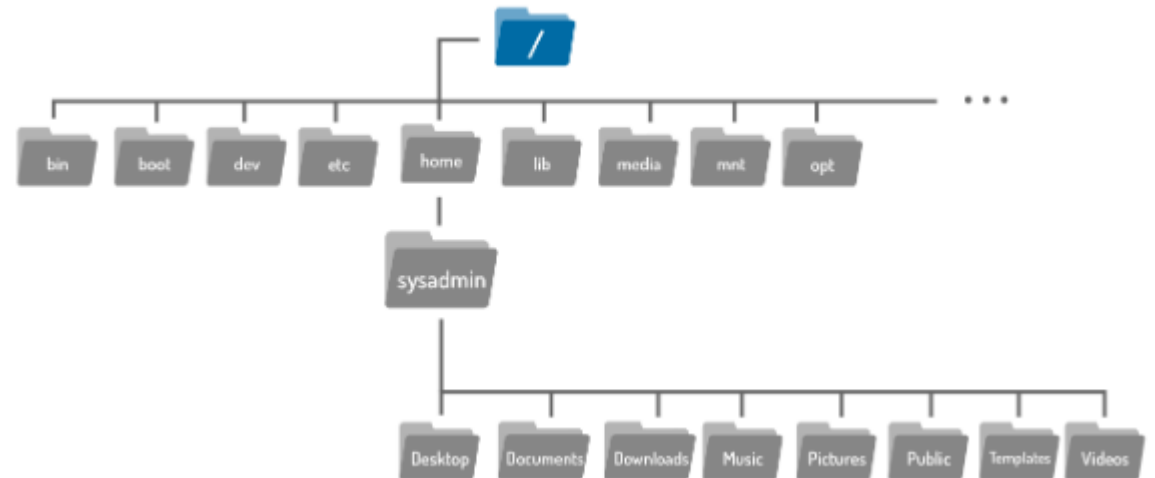


# Syntaxe des commandes de base

- Déplacement dans les répertoires

Les répertoires sont équivalents aux dossiers sous Windows et Mac OS. Comme ces systèmes d'exploitation populaires, la structure d'annuaire Linux comporte un niveau supérieur. Celui-ci ne s'appelle pas « Poste de travail », mais « répertoire root » et est représenté par le caractère /. Pour passer au répertoire root, utilisez le caractère / comme argument de la commande **cd**.

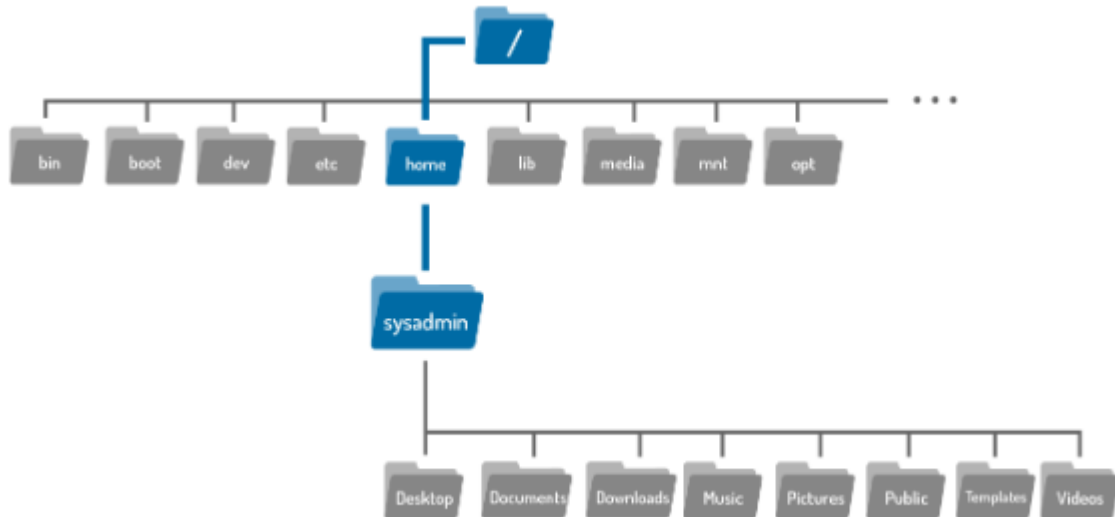
```
sysadmin@localhost:~/Documents$ cd /  
sysadmin@localhost:/$
```



# Syntaxe des commandes de base

- Déplacement dans les répertoires

L'argument de la commande **cd** est plus que le nom d'un répertoire, c'est en fait un *chemin*. Un chemin est une liste de répertoires séparés par le caractère /. Par exemple, /home/sysadmin est le chemin d'accès à votre répertoire personnel :



Si vous considérez le système de fichiers comme une carte, les chemins sont les directions étape par étape ; ils peuvent être utilisés pour indiquer l'emplacement de n'importe quel fichier dans le système de fichiers. Il existe deux types de chemins : *absolus* et *relatifs*. Les chemins absolus commencent à la racine (root) du système de fichiers, les chemins relatifs commencent à partir de votre emplacement actuel.

# Syntaxe des commandes de base

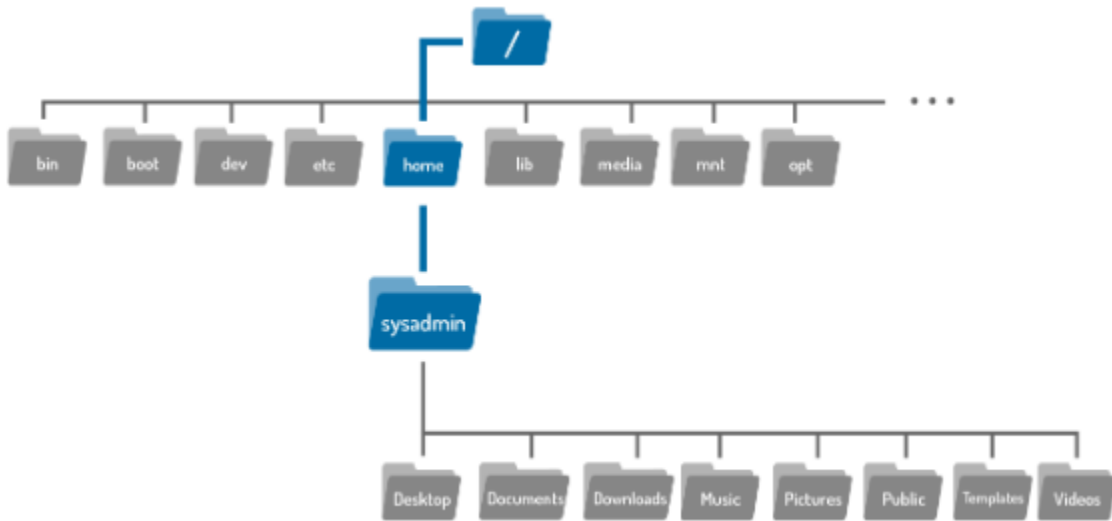
## Chemins absolus

Un chemin absolu vous permet de spécifier l'emplacement exact d'un répertoire.

Il commence toujours au répertoire root, et commence donc toujours par le caractère /.

Le chemin d'accès au répertoire personnel /home/sysadmin est un chemin absolu.

Le chemin commence à partir du répertoire root /, se déplace dans le répertoire personnel, puis dans le répertoire sysadmin. Suivre ce chemin dans une interface utilisateur graphique (GUI) tel que votre ordinateur personnel ressemblerait à ceci :



Utilisez ce chemin comme argument de la commande `cd` pour revenir dans le répertoire personnel de l'utilisateur `sysadmin`.

```
sysadmin@localhost:/$ cd /home/sysadmin
```

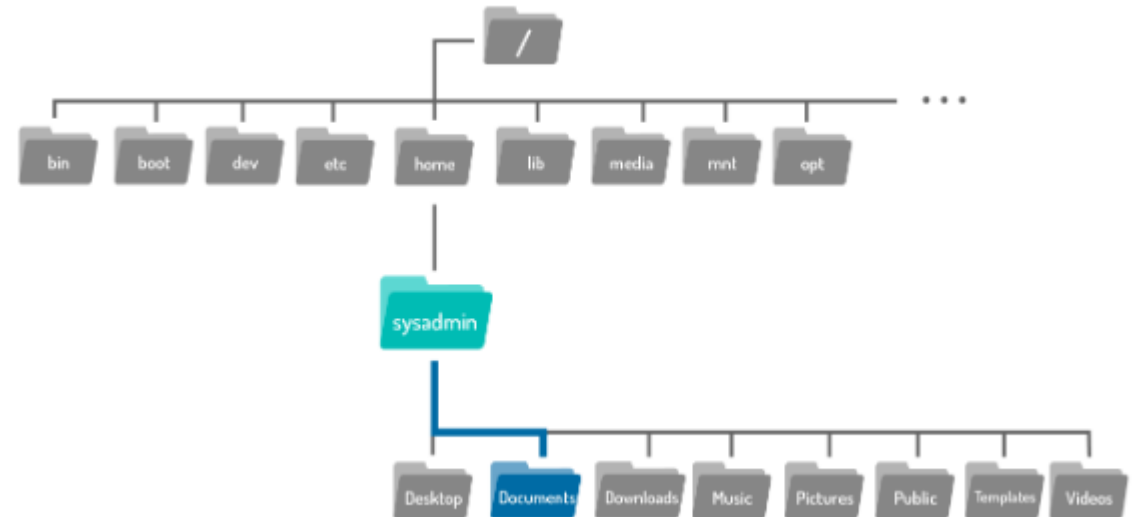
```
sysadmin@localhost:~$ pwd
```

# Syntaxe des commandes de base

## Chemins relatifs

Un chemin relatif donne des directions à un fichier par rapport à votre emplacement actuel dans le système de fichiers. Les chemins relatifs ne commencent pas par le caractère /, ils commencent par le nom d'un répertoire. Regardez le au premier exemple de commande `cd`. L'argument est un exemple du chemin relatif le plus simple : le nom d'un répertoire dans votre emplacement actuel.

```
sysadmin@localhost:~$ cd Documents  
sysadmin@localhost:~/Documents$
```

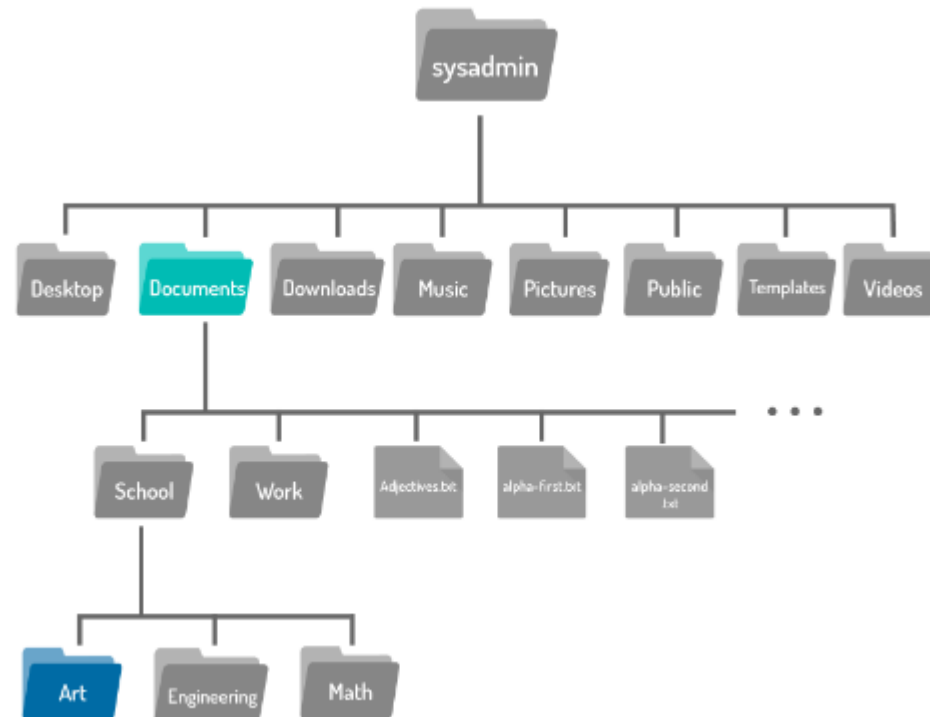


# Syntaxe des commandes de base

## Chemins relatifs

L'image ci-dessous montre une carte des fichiers contenus dans le répertoire sysadmin.

Vous êtes actuellement dans le répertoire Documents et souhaitez vous déplacer vers le répertoire Art :

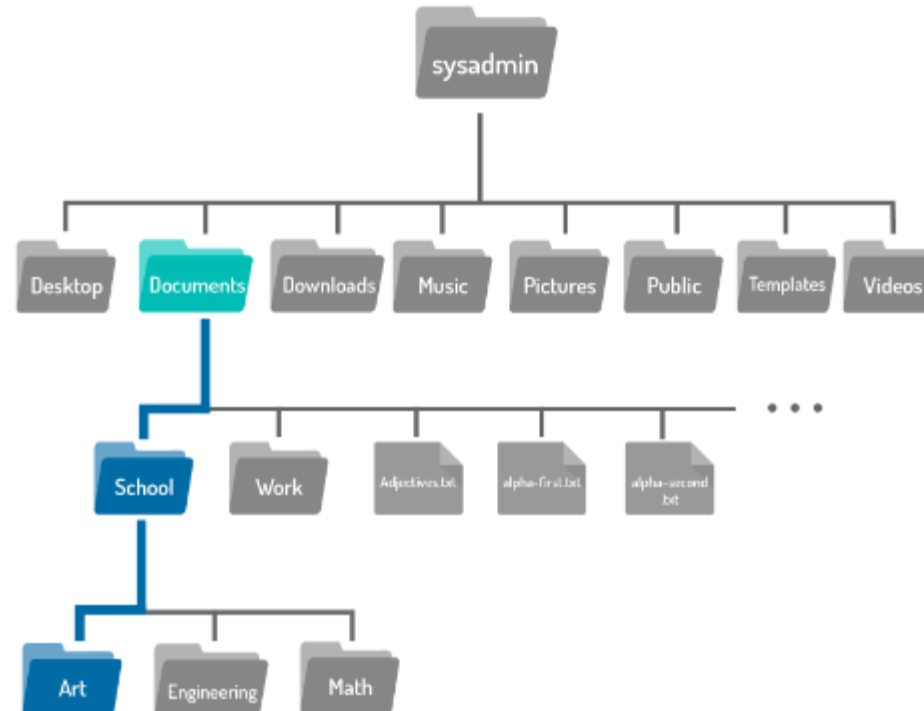


# Syntaxe des commandes de base

## Chemins relatifs

Un chemin relatif commence par le répertoire courant, mais vous ne l'incluez pas dans le chemin.

La première étape consisterait à se déplacer dans le répertoire School, puis à se déplacer dans le répertoire Art. Utilisez le caractère / pour séparer les noms de répertoire. Le résultat School/Art est un chemin relatif entre le répertoire Documents et le répertoire Art :



# Syntaxe des commandes de base

## Chemins relatifs

Utilisez le chemin relatif comme argument de la commande `cd` pour vous déplacer dans le répertoire Art.

```
sysadmin@localhost:~/Documents/$ cd School/Art  
sysadmin@localhost:~/Documents/School/Art$
```

Utilisez la commande `pwd` pour confirmer la modification :

```
sysadmin@localhost:~/Documents/School/Art$ pwd /home/sysadmin/Documents/School/Art
```

# Syntaxe des commandes de base

## Raccourcis

### Le ..

Quel que soit le répertoire dans lequel vous vous trouvez, le double point .. Représente toujours un répertoire supérieur par rapport au répertoire courant, parfois appelé répertoire parent. Pour passer du répertoire Art au répertoire School :

```
sysadmin@localhost:~/Documents/School/Art$ cd ..  
sysadmin@localhost:~/Documents/School$
```

### Le .

Quel que soit le répertoire dans lequel vous vous trouvez, le point . représente toujours votre répertoire actuel. Pour la commande **cd** ce raccourci n'est pas très utile, mais il le sera pour les commandes couvertes dans les sections suivantes.

### Le caractère ~

Le répertoire personnel de l'utilisateur actuel est représenté par le caractère ~. Comme indiqué ci-dessus, vous commencez toujours en tant qu'utilisateur sysadmin, qui se trouve à l'adresse /home/sysadmin. Pour revenir à votre répertoire personnel à tout moment, exécutez la commande suivante :

```
sysadmin@localhost:~/Documents/School$ cd ~  
sysadmin@localhost:~$
```



# Syntaxe des commandes de base

- Liste des fichiers

La commande **ls** est utilisée pour lister le contenu d'un répertoire. Vous l'avez déjà vu utilisée dans des exemples précédents, mais cette page vous aidera à vous assurer que vous êtes à l'aise avec son utilisation.

`ls [Options] [chemin]`

Par défaut, lorsque la commande **ls** est utilisée sans options ni arguments, elle liste les fichiers dans le répertoire courant :

```
sysadmin@localhost:~$ ls  
Desktop Documents Downloads Music Pictures Public Templates Videos
```

# Syntaxe des commandes de base

- Liste des fichiers

Pour connaître les détails d'un fichier, tels que le type de fichier, les autorisations, les propriétés ou l'horodatage, affichez une longue liste à l'aide de l'option **-l** de la commande **ls**. Ci-dessous, l'affichage du contenu du répertoire `/var/log` est utilisée comme exemple, car il présente toute une variété de sorties :

```
sysadmin@localhost:~$ ls -l /var/log/
total 844
-rw-r--r-- 1 root root 18047 Dec 20 2017 alternatives.log
drwxr-x--- 2 root adm 4096 Dec 20 2017 apache2
drwxr-xr-x 1 root root 4096 Dec 20 2017 apt
-rw-r----- 1 syslog adm 1346 Oct 2 22:17 auth.log
-rw-r--r-- 1 root root 47816 Dec 7 2017 bootstrap.log
-rw-rw---- 1 root utmp 0 Dec 7 2017 btmp
-rw-r----- 1 syslog adm 547 Oct 2 22:17 cron.log
-rw-r----- 1 root adm 85083 Dec 20 2017 dmesg
-rw-r--r-- 1 root root 325238 Dec 20 2017 dpkg.log
-rw-r--r-- 1 root root 32064 Dec 20 2017 faillog
drwxr-xr-x 2 root root 4096 Dec 7 2017 fsck
-rw-r----- 1 syslog adm 106 Oct 2 19:57 kern.log
-rw-rw-r-- 1 root utmp 292584 Oct 2 19:57 lastlog
-rw-r----- 1 syslog adm 19573 Oct 2 22:57 syslog
drwxr-xr-x 2 root root 4096 Apr 11 2014 upstart
-rw-rw-r-- 1 root utmp 384 Oct 2 19:57 wtmp
```

# Syntaxe des commandes de base

## Liste des fichiers

Les champs sont les suivants :

-	rw-r--r--	1	root	root	18047	Dec 20 2017	alternatives.log
D	rwxr-x---	2	root	adm	4096	Dec 20 2017	apache2

The diagram illustrates the components of a file command output line. Blue arrows point from descriptive labels below to specific fields in the table above:

- Type de fichier** points to the first column (file type).
- Permissions** points to the second column (permissions).
- Nombre de liens physiques** points to the third column (link count).
- Propriétaire** points to the fourth column (owner).
- Groupe** points to the fifth column (group).
- Taille du fichier** points to the sixth column (file size).
- Horodatage** points to the seventh column (timestamp).
- Nom du fichier** points to the eighth column (filename).

# Syntaxe des commandes de base

- Liste des fichiers
- Les types de fichiers sont les suivants :

Symbole	Type de fichier	Description
d	répertoire	Un fichier utilisé pour stocker d'autres fichiers.
-	fichier	Inclut des fichiers lisibles, des fichiers images, des fichiers binaires et des fichiers compressés.
l	lien symbolique	Pointe vers un autre fichier.
s	prise (socket)	Permet la communication entre les processus.
p	tuyau (pipe)	Permet la communication entre les processus.
b	fichier bloc (block file)	Utilisé pour communiquer avec le matériel.
c	fichier de caractères (character file)	Utilisé pour communiquer avec le matériel.

# Syntaxe des commandes de base

- Liste des fichiers
- **Permissions:** Les permissions indiquent comment certains utilisateurs peuvent accéder à un fichier.
- **Nombre de liens physiques :** Ce nombre indique le nombre de liens physiques pointant vers ce fichier.
- **Propriétaire du fichier :** Chaque fois qu'un fichier est créé, la propriété est automatiquement attribuée à l'utilisateur qui l'a créé.
- **Groupe propriétaire :** Indique quel groupe est propriétaire du fichier.
- **Taille du fichier**
- **Horodatage :** Il s'agit de l'heure à laquelle le contenu du fichier a été modifié pour la dernière fois.
- **Nom du fichier :** Le dernier champ contient le nom du fichier ou du répertoire

# Syntaxe des commandes de base

- Tri

Par défaut, la sortie de la commande `ls` est affichée par ordre alphabétique sur le nom de fichier. Le tri peut aussi être fait selon d'autres méthodes

L'option `-t` trie les fichiers par horodatage ou plus simplement selon la date de la dernière modification du fichier :

```
sysadmin@localhost:~$ ls -lt /var/log
total 844
-rw-r----- 1 syslog adm    19573 Oct  2 22:57 syslog
-rw-r----- 1 syslog adm     1346 Oct  2 22:17 auth.log
-rw-r----- 1 syslog adm      547 Oct  2 22:17 cron.log
-rw-rw-r-- 1 root  utmp  292584 Oct  2 19:57 lastlog
-rw-rw-r-- 1 root  utmp     384 Oct  2 19:57 wtmp
-rw-r----- 1 syslog adm     106 Oct  2 19:57 kern.log
-rw-r--r-- 1 root  root   18047 Dec 20  2017 alternatives.log
-rw-r--r-- 1 root  root   32064 Dec 20  2017 faillog
-rw-r----- 1 root  adm    85083 Dec 20  2017 dmesg
-rw-r--r-- 1 root  root  325238 Dec 20  2017 dpkg.log
drwxr-x--- 2 root  adm     4096 Dec 20  2017 apache2
drwxr-xr-x 1 root  root     4096 Dec 20  2017 apt
-rw-r--r-- 1 root  root   47816 Dec  7  2017 bootstrap.log
drwxr-xr-x 2 root  root     4096 Dec  7  2017 fsck
-rw-rw---- 1 root  utmp        0 Dec  7  2017 btmp
drwxr-xr-x 2 root  root     4096 Apr 11  2014 upstart
```

# Syntaxe des commandes de base

- Tri

L'option **-S** trie les fichiers selon la taille de fichier :

```
sysadmin@localhost:~$ ls -l -S /var/log
total 844
-rw-r--r-- 1 root    root 325238 Dec 20  2017 dpkg.log
-rw-rw-r-- 1 root    utmp 292584 Oct  2 19:57 lastlog
-rw-r----- 1 root    adm  85083 Dec 20  2017 dmesg
-rw-r--r-- 1 root    root  47816 Dec  7  2017 bootstrap.log
-rw-r--r-- 1 root    root  32064 Dec 20  2017 faillog
-rw-r----- 1 syslog  adm  19573 Oct  2 22:57 syslog
-rw-r--r-- 1 root    root  18047 Dec 20  2017 alternatives.log
drwxr-x--- 2 root    adm   4096 Dec 20  2017 apache2
drwxr-xr-x 1 root    root   4096 Dec 20  2017 apt
drwxr-xr-x 2 root    root   4096 Dec  7  2017 fsck
drwxr-xr-x 2 root    root   4096 Apr 11  2014 upstart
-rw-r----- 1 syslog  adm   1346 Oct  2 22:17 auth.log
-rw-r----- 1 syslog  adm    547 Oct  2 22:17 cron.log
-rw-rw-r-- 1 root    utmp   384 Oct  2 19:57 wtmp
-rw-r----- 1 syslog  adm    106 Oct  2 19:57 kern.log
-rw-rw---- 1 root    utmp     0 Dec  7  2017 btmp
```

# Syntaxe des commandes de base

- Accès administratif

Il existe de nombreuses commandes Linux qui traitent des informations sensibles comme les mots de passe, les configurations système ou toutes autres commandes qui s'avèrent utiles dans des cas d'exception. Empêcher les utilisateurs standards d'exécuter ces commandes aide à protéger le système. Se connecter en tant qu'utilisateur root fournit un accès administratif privilégié, permettant l'exécution de certaines des commandes privilégiées.

- La commande **Su**

## Su [Options] Nom d'utilisateur

La commande **su** vous permet d'agir temporairement en tant que super utilisateur. Elle le fait en créant un nouveau shell.

Le shell est simplement une console de saisie de texte qui vous permet de taper des commandes. Par défaut, si un compte utilisateur n'est pas spécifié, la commande **su** ouvrira un nouvel interpréteur de commandes en tant qu'administrateur système (root) et fournit des privilèges d'administration.

Après l'exécution de la commande **su**, un mot de passe est requis

```
sysadmin@localhost:~$ su -  
Password:  
root@localhost:~#
```



# Syntaxe des commandes de base

- La commande **Su**

Notez que l'invite de commandes a été modifiée pour refléter que vous êtes maintenant connecté en tant qu'utilisateur root. Pour vous déconnecter et revenir au compte sysadmin, utilisez la commande **exit**.  
Notez que l'invite de commande retourne à un utilisateur normal :

```
root@localhost:~# exit
logout
sysadmin@localhost:~$
```

Exécuter la commande steam locomotive sl en mode administrative (sinon elle échouera ).  
Utiliser la commande su pour bascule en mode admin puis exécuter sl



# Syntaxe des commandes de base

- La commande **sudo**
- Permet à un utilisateur d'exécuter une commande avec des privilèges d'administration sans basculer vers le compte administrateur
- La commande **sudo** fournit uniquement un accès administratif pour l'exécution de la commande spécifiée. C'est un avantage car il réduit le risque qu'un utilisateur exécute accidentellement une commande en tant que root.

```
sysadmin@localhost:~$ sudo sl [sudo]  
password for sysadmin:
```



# Syntaxe des commandes de base

- Permissions

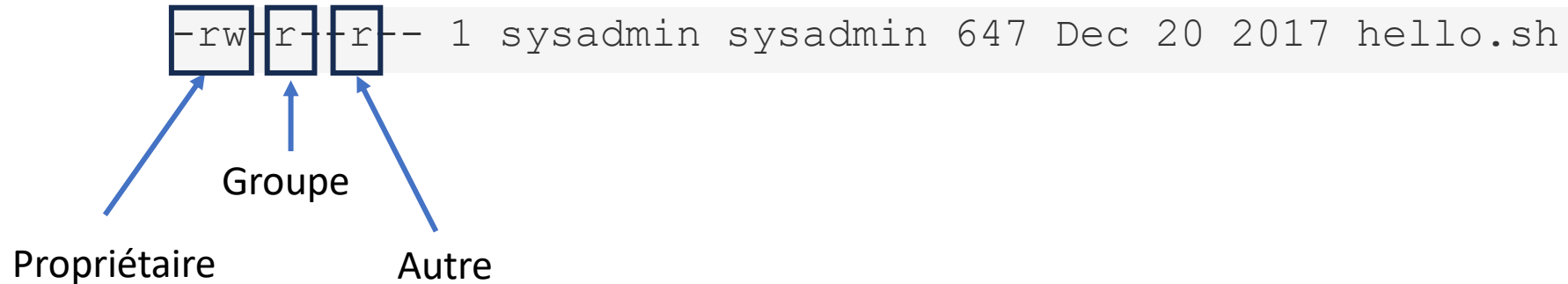
Les permissions déterminent la façon dont différents utilisateurs peuvent interagir avec un fichier ou un répertoire. Lorsque vous listez les fichiers avec la commande `ls -l`, la sortie contient des informations de permissions. Pour l'exemple, nous allons utiliser un script appelé `hello.sh` situé dans le répertoire Documents :

```
sysadmin@localhost:~$ cd ~/Documents
```

```
sysadmin@localhost:~/Documents$ ls -l hello.sh  
-rw-r--r-- 1 sysadmin sysadmin 647 Dec 20 2017 hello.sh
```

# Syntaxe des commandes de base

## Champ – Permissions



**Propriétaire :** Le premier ensemble est destiné à l'utilisateur propriétaire du fichier. Si votre compte actuel est le propriétaire du fichier, le premier ensemble des trois permissions s'appliquera et les autres permissions n'auront aucun effet.

**Groupe :** Le deuxième ensemble est pour le groupe propriétaire du fichier. Si votre compte actuel *n'est pas* le propriétaire utilisateur du fichier et que vous *êtes* membre du groupe propriétaire du fichier, ces permissions de groupe s'appliqueront et les autres permissions n'ont aucun effet.

**Autre :** Le dernier ensemble est pour tous les autres, tous ceux auxquels les deux premiers ensembles de permissions ne s'appliquent pas. Si vous n'êtes pas l'utilisateur propriétaire du fichier ou un membre du groupe qui le possède, le troisième ensemble de permissions s'applique à vous.

# Syntaxe des commandes de base

- Types de permissions
- Il existe trois permissions différentes s'appliquant à un fichier ou un répertoire : *lire (read)*, *écrire (write)* et *exécuter (execute)*. La manière dont ces permissions s'appliquent diffère pour les fichiers et les répertoires, comme indiqué dans le tableau ci-dessous :

Permission	Effets sur le fichier	Effets sur le dossier
lire (r) (read)	Permet de lire le contenu du fichier et de le copier.	Sans le droit d'exécution sur le dossier, ne permet pas d'obtenir une liste détaillée du contenu du dossier. Avec le droit d'exécution, ls -l affiche une liste détaillée.
écrire (w) (write)	Permet de modifier ou de réécrire le contenu du fichier. Permet d'ajouter et de retirer des fichiers d'un répertoire.	La permission ne fonctionne que si le dossier possède le droit d'exécution.
exécuter (x) (execute)	Permet d'exécuter le script contenu dans un fichier. Ce fichier doit également avoir le droit de lecture.	Permet à l'utilisateur de modifier le dossier uniquement si le dossier parent possède le droit d'écriture.

# Syntaxe des commandes de base

- Modification des permissions de fichier

La commande **chmod** est utilisée pour modifier les autorisations d'un fichier ou d'un répertoire. Seul l'administrateur du système (root) ou l'utilisateur propriétaire du fichier peut modifier les autorisations d'un fichier.

```
chmod [<SET><ACTION><PERMISSIONS>] ... FICHIER
```

# Syntaxe des commandes de base

- Modification des permissions de fichier

La commande **chmod** est utilisée pour modifier les autorisations d'un fichier ou d'un répertoire. Seul l'administrateur du système (root) ou l'utilisateur propriétaire du fichier peut modifier les autorisations d'un fichier.

```
chmod [<SET><ACTION><PERMISSIONS>] ... FICHIER
```

Symbole	Signification
u	Utilisateur : le propriétaire du fichier.
g	Groupe : le groupe propriétaire du fichier.
o	Autre (others) : tout autre utilisateur qui n'appartient pas aux deux premiers groupes.
a	Tous (all) : se réfère à l'ensemble des utilisateurs.

# Syntaxe des commandes de base

- Modification des permissions de fichier

La commande **chmod** est utilisée pour modifier les autorisations d'un fichier ou d'un répertoire. Seul l'administrateur du système (root) ou l'utilisateur propriétaire du fichier peut modifier les autorisations d'un fichier.

```
chmod [<SET><ACTION><PERMISSIONS>] ... FICHIER
```

Symbole	Signification
+	Ajouter le droit
=	Affecter le droit
-	Supprimer le droit



# Syntaxe des commandes de base

- Modification des permissions de fichier

La commande **chmod** est utilisée pour modifier les autorisations d'un fichier ou d'un répertoire. Seul l'administrateur du système (root) ou l'utilisateur propriétaire du fichier peut modifier les autorisations d'un fichier.

```
chmod [<SET><ACTION><PERMISSIONS>] ... FICHIER
```

Symbole	Signification
r	lire (read)
w	écrire write
x	exécuter (execute)

# Syntaxe des commandes de base

- Modification des permissions de fichier

La commande **chmod** est utilisée pour modifier les autorisations d'un fichier ou d'un répertoire. Seul l'administrateur du système (root) ou l'utilisateur propriétaire du fichier peut modifier les autorisations d'un fichier.

```
sysadmin@localhost:~/Documents$ ls -l hello.sh -rw-r--r-- 1 sysadmin sysadmin 647 Dec 20 2017 hello.sh
```

```
sysadmin@localhost:~/Documents$ ./hello.sh  
-bash: ./hello.sh: Permission denied
```

```
sysadmin@localhost:~/Documents$ chmod u+x hello.sh
```

# Syntaxe des commandes de base

## Modification de la propriété des fichiers

Initialement, le propriétaire d'un fichier est l'utilisateur qui le crée. La commande **chown** est utilisée pour modifier la propriété des fichiers et des répertoires. La modification du propriétaire d'un fichier nécessite un accès administratif. Un utilisateur normal ne peut pas utiliser cette commande pour changer le propriétaire d'un fichier, même si c'est pour donner la propriété d'un de ses propres fichiers à un autre utilisateur. Cependant, la commande **chown** permet de changer la propriété du groupe, et peut être accompli par l'administrateur du système (root) ou par le propriétaire du fichier. Pour changer le propriétaire d'un fichier, la syntaxe suivante peut être utilisée. Le premier argument, [PROPRIÉTAIRE], spécifie quel utilisateur doit être le nouveau propriétaire. Le deuxième argument, FICHIER, spécifie le fichier dont la propriété est en cours de modification.

```
chown [OPTIONS] [PROPRIÉTAIRE] FICHIER
```

# Syntaxe des commandes de base

- Faites ls et choisissez un fichier dont vous allez modifier le propriétaire

Pour modifier la propriété du script hello.sh et la passer à l'utilisateur root, utilisez root comme premier argument et hello.sh comme second argument. N'oubliez pas d'utiliser la commande **sudo** pour obtenir les privilèges administratifs nécessaires. Utilisez le mot de passe netlab123 lorsque vous y êtes invité :

```
-rwxr--r-- 1 sysadmin sysadmin 647 Dec 20 2017 hello.sh
```

```
sysadmin@localhost:~/Documents$ sudo chown root hello.sh  
[sudo] password for sysadmin:
```

En attribuant la propriété du fichier à root, ce fichier nécessite désormais un accès administratif .

# Syntaxe des commandes de base

- Création de fichiers
- Pour créer un fichier vide, utiliser la commande

`touch`

```
sysadmin@localhost:~$ touch sample.txt example.txt test.txt
sysadmin@localhost:~$ ls Desktop Downloads Pictures Templates example.txt test.txt
Documents Music Public Videos sample.txt
```

# Syntaxe des commandes de base

- Affichage des fichiers

Quelques commandes Linux permettent d'afficher le contenu des fichiers. La commande **cat**, qui signifie « concaténer », est souvent utilisée pour visualiser rapidement le contenu des petits fichiers.

La commande **cat** affichera tout le contenu du fichier, c'est pourquoi elle est principalement recommandée pour les fichiers plus petits où la sortie est limitée et ne nécessite pas de défilement. Pour afficher le contenu d'un fichier à l'aide de la commande **cat**, tapez simplement la commande suivi du nom du fichier que vous souhaitez voir comme argument :

```
cat [OPTIONS] [FICHIER]
```

Exemple : cat animals.txt

La commande **cat** peut entraîner une sortie très longue qui ne peut pas être interrompue pour défiler dedans. Une meilleure méthode pour afficher les longs fichiers texte est d'utiliser une commande page par page avec une fonctionnalité permettant de faire une pause et de faire défiler la sortie du fichier.

# Syntaxe des commandes de base

- Affichage des fichiers

Une autre façon d'afficher le contenu des fichiers est d'utiliser les commandes **head** et **tail**. Ces commandes permettent d'afficher un certain nombre de lignes en haut ou en bas d'un fichier (par défaut 10 lignes ).

```
head [OPTIONS] [FICHIER]
tail [OPTIONS] [FICHIER]
```

Pour spécifier le nombre le ligne a afficher, utilisez l'option -n

```
sysadmin@localhost:~/Documents$ head -n 5 alpha.txt
A is for Apple
B is for Bear
C is for Cat
D is for Dog
E is for Elephant
```

# Syntaxe des commandes de base

- Copie de fichiers
- La création de copies de fichiers peut être utile pour de nombreuses raisons:
- Si une copie d'un fichier est créée avant que les modifications ne soient apportées, il est possible de revenir à l'original.
- Une copie d'un fichier peut être utilisée pour transférer un fichier sur un support amovible.
- Une copie d'un document existant peut servir de modèle pour un nouveau document.

```
cp [OPTIONS] SOURCE DESTINATION
```



# Syntaxe des commandes de base

NB : Les autorisations peuvent avoir un impact sur les commandes de gestion de fichiers, telles que la commande **cp**. Pour copier un fichier, il est nécessaire d'avoir l'autorisation d'exécution pour accéder au répertoire où se trouve le fichier et l'autorisation de lecture pour le fichier en cours de copie. Il est également nécessaire d'avoir l'autorisation d'écriture et d'exécution sur le répertoire dans lequel le fichier est copié. En règle générale, vous devez toujours avoir l'autorisation d'écriture et d'exécution sur votre répertoire personnel et sur le répertoire /tmp.

Exemple :

```
sysadmin@localhost:~$ cd ~/Documents
```

pour copier le fichier /etc/passwd dans le répertoire courant, utilisez la commande suivante :

```
sysadmin@localhost:~/Documents$ cp /etc/passwd .
```

NB : . Est un raccourci qui représente le répertoire courant

# Syntaxe des commandes de base

- Déplacement de fichiers

La commande `mv` permet de déplacer un fichier d'un emplacement dans le système de fichiers à un autre.

```
mv SOURCE DESTINATION
```

Pour déplacer le fichier `people.csv` dans le répertoire `Work`, utilisez le nom de fichier comme source et le nom du répertoire comme destination :

```
sysadmin@localhost:~/Documents$ mv people.csv Work
```

La commande `mv` permet aussi de déplacer plusieurs fichiers, tant que l'argument final fourni à la commande est la destination. Par exemple, pour déplacer trois fichiers dans le répertoire `School` :

```
sysadmin@localhost:~/Documents$ mv numbers.txt letters.txt alpha.txt School
sysadmin@localhost:~/Documents$ ls School
Art Engineering Math alpha.txt letters.txt numbers.txt
```

NB : Les autorisations peuvent avoir un impact sur les commandes de gestion de fichiers, telles que la commande `mv`. Le déplacement d'un fichier nécessite des autorisations d'écriture et d'exécution sur les répertoires d'origine et de destination.

# Syntaxe des commandes de base

- Suppression de fichiers

La commande **rm** permet de supprimer des fichiers et des répertoires. Il est important de garder à l'esprit que les fichiers et répertoires supprimés ne vont pas dans une « corbeille » comme dans les systèmes d'exploitation avec une interface de type Windows. Lorsqu'un fichier est supprimé avec la commande **rm**, il est presque toujours supprimé de manière définitive.

```
rm [OPTIONS] FICHIER
```

La commande **rm** ignore les répertoires qu'on lui demande de supprimer ; pour supprimer un répertoire, utilisez une option récursive, soit les options **-r** ou **-R**. Soyez prudent puisque ces options sont « récursives », cela supprimera tous les fichiers et tous les sous-répertoires :

```
sysadmin@localhost:~/Documents$ rm Work
rm: cannot remove 'Work': Is a directory
sysadmin@localhost:~/Documents$ rm -r Work
sysadmin@localhost:~/Documents$ ls Work
ls: cannot access Work: No such file or directory
```

Les autorisations peuvent avoir un impact sur les commandes de gestion de fichiers, telles que la commande **rm**. Pour supprimer un fichier dans un répertoire, un utilisateur doit disposer de l'autorisation d'écriture et d'exécution sur ce répertoire. Les utilisateurs standards ne disposent généralement de ce type d'autorisation que dans leur répertoire personnel et ses sous-répertoires.

# Syntaxe des commandes de base

- Filtrage d'entrée

La commande **grep** est un filtre de texte qui recherche les entrées et renvoie les lignes qui contiennent une correspondance avec un motif donné.

```
grep [OPTIONS] MOTIF [FICHIER]
```

Par exemple, le fichier `passwd` que nous avons copié précédemment dans le répertoire Documents contient les détails des comptes système spéciaux et des comptes utilisateur sur le système. Ce fichier peut être très volumineux, mais la commande **grep** peut être utilisée pour filtrer les informations sur un utilisateur spécifique, tel que l'utilisateur `sysadmin`. Utilisez `sysadmin` comme argument du motif et `passwd` comme argument fichier

```
sysadmin@localhost:~/Documents$ grep sysadmin passwd
sysadmin:x:1001:1001:System Administrator,,,:/home/sysadmin:/bin/bash
```

# Syntaxe des commandes de base

- Expressions régulières
- Les expressions régulières ont deux formes communes : *basique* et *étendue*. La plupart des commandes qui utilisent des expressions régulières peuvent interpréter les expressions régulières basiques. Cependant, les expressions régulières étendues ne sont pas disponibles pour toutes les commandes et une option de commande est généralement requise pour qu'elles fonctionnent correctement.
- Le tableau suivant résume les caractères d'expression régulière basiques :

Caractères regex basiques	Signification
.	Un seul caractère
[ ]	N'importe quel caractère spécifié
[^ ]	Pas le caractère spécifié
*	Zéro ou plus du caractère précédent
^	Si premier caractère du motif, le motif doit être au début de la ligne pour correspondre, sinon cela correspond à un caractère caret ^ littéral.
\$	Si dernier caractère du motif, le motif doit être à la fin de la ligne pour correspondre, sinon cela correspond à un signe dollar \$ littéral.

# Syntaxe des commandes de base

- Expressions régulières

Le tableau suivant résume les expressions régulières étendues, qui doivent être utilisées avec la commande **egrep** ou l'option **-E** avec la commande **grep** :

Caractères regex étendus	Signification
+	Un ou plusieurs du motif précédent
?	Le motif précédent est optionnel
{ }	Spécifier correspondances minimum, maximum ou exactes du motif précédent
	Alternance - un "ou" logique
( )	Utilisé pour créer des groupes

# Syntaxe des commandes de base

- Expressions régulières
- Exemple

```
sysadmin@localhost:~/Documents$ grep 'root' passwd
root:x:0:0:root:/root:/bin/bash
operator:x:1000:37::/root:
```

```
sysadmin@localhost:~/Documents$ grep '^root' /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

```
sysadmin@localhost:~/Documents$ grep 'r$' alpha-first.txt
B is for Bear
F is for Flower
```

```
sysadmin@localhost:~/Documents$ grep 'r..f' red.txt
reef
roof
```

```
sysadmin@localhost:~/Documents$ grep 'r..d' red.txt
reed
read
```

# Syntaxe des commandes de base

- Expressions régulières
- Exemple

```
sysadmin@localhost:~/Documents$ grep '....' red.txt
reef
reeed
roof
reed
root
reel
read
```

```
sysadmin@localhost:~/Documents$ grep '[0-9]' profile.txt
I am 37 years old.
3121991
I have 2 dogs.
123456789101112
```

```
sysadmin@localhost:~/Documents$ grep '^[0-9]' profile.txt
Hello my name is Joe.
I am 37 years old.
My favorite food is avocados.
I have 2 dogs.
```

```
sysadmin@localhost:~/Documents$ grep '[.]' profile.txt
Hello my name is Joe.
I am 37 years old.
My favorite food is avocados.
I have 2 dogs.
```



# Syntaxe des commandes de base

- Expressions régulières
- Exemple

```
sysadmin@localhost:~/Documents$ cat red.txt
red
reef
rot
reed
rd
rod
roof
reed
root
reel
read

sysadmin@localhost:~/Documents$ grep 're*d' red.txt
red
reed
rd
reed
```

```
sysadmin@localhost:~/Documents$ grep 'z*' red.txt
red
reef
rot
reed
rd
rod
roof
reed
root
reel
read
```

# Syntaxe des commandes de base

Expressions régulières

Exemple

```
sysadmin@localhost:~/Documents$ grep 'ee*' red.txt  
red  
reef  
reed  
reed  
reel  
read
```

# Syntaxe des commandes de base

## Arrêt

La commande **shutdown** permet d'arrêter le système en toute sécurité. Tous les utilisateurs connectés sont alors avertis que le système est en cours d'arrêt et au cours des cinq dernières minutes précédant l'arrêt, les nouvelles connexions sont empêchées.

```
shutdown [OPTIONS] HEURE [MESSAGE]
```

NB : La commande nécessite un accès administratif, il faut donc utiliser la commande `su` – ou `sudo`

Contrairement à d'autres commandes utilisées pour arrêter le système, la commande **shutdown** nécessite un argument de temps spécifiant quand l'arrêt doit commencer. Les formats de cet argument de temps peuvent être le mot `now`, une heure de la journée au format `hh:mm` ou le nombre de minutes pour différer au format `+minutes`.

```
root@localhost:~# shutdown now
Broadcast message from sysadmin@localhost
(/dev/pts/0) at 2:05 ...
The system is going down for maintenance NOW!
```

La commande **shutdown** peut également contenir un message facultatif, qui apparaîtra sur les terminaux de tous les utilisateurs. Par exemple :

```
root@localhost:~# shutdown +1 "Goodbye World!"
Broadcast message from sysadmin@localhost (/dev/pts/0) at 3:07 ...
The system is going down for maintenance in 1 minute! Goodbye World!
shutdown: Unable to shutdown system
```

# Syntaxe des commandes de base

- Configuration réseau

La commande **ifconfig** signifie « configuration d'interface » et est utilisée pour afficher les informations de configuration réseau.

```
ifconfig [OPTIONS]
```

NB : La commande **iwconfig** est similaire à la commande **ifconfig**, mais elle est dédiée aux interfaces réseau sans fil.

```
root@localhost:~# ifconfig
eth0
    Link encap:Ethernet  HWaddr 02:42:c0:a8:01:02
    inet addr:192.168.1.2
    Bcast:192.168.1.255  Mask:255.255.255.0
    UP
    BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
    RX packets:59 errors:0 dropped:0 overruns:0 frame:0
    TX packets:86 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:1000
    RX bytes:4346 (4.3 KB)  TX bytes:5602 (5.6 KB)

lo
    Link encap:Local Loopback
    inet addr:127.0.0.1  Mask:255.0.0.0
    UP LOOPBACK RUNNING  MTU:65536  Metric:1
    RX packets:2 errors:0 dropped:0 overruns:0 frame:0
    TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:1000
    RX bytes:100 (100.0 B)  TX bytes:100 (100.0 B)
```

# Syntaxe des commandes de base

- Configuration réseau

La commande **ping** permet de vérifier la connectivité entre deux ordinateurs.

Par défaut, la commande **ping** continuera d'envoyer des paquets jusqu'à ce que la commande d'interruption (**CTL+C**) soit entrée sur la console. Pour limiter le nombre de pings envoyés, utilisez l'option **-c** suivie du nombre de pings à envoyer.

```
root@localhost:~# ping -c 4 192.168.1.2
PING 192.168.1.2 (192.168.1.2) 56(84) bytes of data.
64 bytes from 192.168.1.2: icmp_req=1 ttl=64 time=0.051 ms
64 bytes from 192.168.1.2: icmp_req=2 ttl=64 time=0.064 ms
64 bytes from 192.168.1.2: icmp_req=3 ttl=64 time=0.050 ms
64 bytes from 192.168.1.2: icmp_req=4 ttl=64 time=0.043 ms

--- 192.168.1.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2999ms
rtt min/avg/max/mdev = 0.043/0.052/0.064/0.007 ms
root@localhost:~#
```

# Syntaxe des commandes de base

- Affichage des processus
- L'exécution d'une commande entraîne la création d'un *processus* en mémoire. Dans le système d'exploitation Linux, les processus sont exécutés avec les privilèges de l'utilisateur qui exécute la commande. Cela permet de limiter les processus à certaines fonctionnalités en fonction de l'identité de l'utilisateur.

Bien qu'il existe des exceptions, le système d'exploitation différencie généralement les utilisateurs selon qu'ils sont ou non l'administrateur. En règle générale, les utilisateurs normaux, comme l'utilisateur sysadmin, ne peuvent pas contrôler les processus d'un autre utilisateur. Les utilisateurs disposant de privilèges d'administration, comme le compte root, peuvent contrôler tous les processus utilisateur, y compris l'arrêt de tout processus utilisateur.

La commande **ps** peut être utilisée pour lister les processus.

```
ps [OPTIONS]
```

# Syntaxe des commandes de base

```
sysadmin@localhost:~$ ps
PID TTY TIME CMD
80 pts/0 00:00:00 bash
94 pts/0 00:00:00 ps
```

La commande **ps** affiche les processus en cours d'exécution dans le terminal actuel par défaut. Dans l'exemple ci-dessus, la dernière ligne est le processus créé par l'exécution de la commande **ps**.

La sortie comprend les colonnes d'informations suivantes :

- PID : L'identifiant du processus, qui est unique au processus. Ces informations sont utiles pour contrôler le processus par son numéro d'identification.
- TTY : Le nom du terminal sur lequel le processus est en cours d'exécution. Ces informations sont utiles pour distinguer les différents processus qui portent le même nom.
- TIME : Le temps processeur utilisé par le processus. Généralement, cette information n'est pas utilisée par les utilisateurs standards.
- CMD : La commande qui a lancé le processus.

L'option :

- e : affichera tous les processus
- f : affiche plus de détails

# Syntaxe des commandes de base

- Gestion des paquets

- La gestion des paquets est un système par lequel les logiciels peuvent être installés, mis à jour, interrogés ou supprimés d'un système de fichiers. Sous Linux, il existe de nombreux systèmes de gestion de paquets de logiciels différents, mais les plus populaires sont ceux de Debian et de Red Hat. Les machines virtuelles de ce cours utilisent Ubuntu, un dérivé de Debian.

Au niveau le plus bas du système de gestion des paquets Debian se trouve la commande **dpkg**. Cette commande peut être compliquée pour les utilisateurs débutant sur Linux et l'outil Advanced Package Tool, **apt-get**, un programme frontal de l'outil **dpkg**, permet de rendre la la gestion des paquets plus simple.

- NB : la plupart des commandes de gestion de paquet nécessitent un accès administratif, de sorte qu'elles seront précédées par la commande **sudo**



# Syntaxe des commandes de base

- Installation de paquets

- Les fichiers de paquets sont généralement installés en les téléchargeant directement à partir de référentiels situés sur des serveurs Internet. Les référentiels Debian contiennent plus de 65 000 paquets de logiciels différents. Avant d'installer un paquet, il est recommandé de réactualiser la liste des paquets disponibles à l'aide de la commande

- `apt-get update`

```
sysadmin@localhost:~$ sudo apt-get update
[sudo] password for sysadmin:
Ign file: amd64/ InRelease
Ign file: amd64/ Release.gpg
Ign file: amd64/ Release
Reading package lists... Done
```

Une fois que vous avez trouvé le paquet à installer, vous pouvez l'installer avec la commande `apt-get install` :

```
sudo apt-get install [paquet]
```

NB : la commande peut également actualiser un paquet

`apt-get remove` pour supprimer un paquet

# Syntaxe des commandes de base

- Mise à jour des mots de passe utilisateur

La commande **passwd** permet de modifier le mot de passe d'un utilisateur. Les utilisateurs ne peuvent modifier que leurs propres mots de passe, tandis que l'utilisateur root peut modifier le mot de passe de n'importe quel utilisateur.

```
passwd [OPTIONS] [UTILISATEUR]
```

Exécutez la commande **passwd**. Vous serez invité à entrer le mot de passe existant une fois et le nouveau mot de passe deux fois. Pour des raisons de sécurité, aucune sortie n'est affichée pendant la saisie du mot de passe.

```
sysadmin@localhost:~$ passwd
Changing password for sysadmin.
(current) UNIX password: netlab123

Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

# Syntaxe des commandes de base

- Mise à jour des mots de passe utilisateur
- Si l'utilisateur souhaite afficher les informations d'état de son mot de passe, il peut utiliser l'option **-S**

```
sysadmin@localhost:~$ passwd -S sysadmin  
sysadmin P 12/20/2017 0 99999 7 -1
```

Champs	Exemple	Signification
Nom de l'utilisateur	sysadmin	Le nom de l'usager.
État du mot de passe	P	P indique un mot de passe utilisable. L indique un mot de passe verrouillé. NP indique aucun mot de passe.
Date de modification	03/01/2015	Date à laquelle le mot de passe a été modifié pour la dernière fois.
Minimum	0	Le nombre minimum de jours qui doivent s'écouler avant que le mot de passe actuel puisse être modifié par l'utilisateur.
Maximum	99999	Le nombre maximum de jours restants avant que le mot de passe expire.
Avertir	7	Le nombre de jour avant l'expiration du mot de passe auquel l'utilisateur est averti.
Inactif	-1	Le nombre de jours pouvant s'écouler après l'expiration du mot de passe pendant lequel le compte reste actif.

# Syntaxe des commandes de base

- Redirection

# Syntaxe des commandes de base

## • Éditeur de texte

Le principal éditeur de texte pour Linux et UNIX est un programme appelé **vi**. Bien qu'il existe de nombreux éditeurs disponibles pour Linux qui vont du minuscule éditeur **nano** à l'imposant éditeur **emacs**, l'éditeur **vi** présente plusieurs avantages :

- L'éditeur **vi** est disponible sur toutes les distributions Linux dans le monde. Ce n'est pas le cas des autres éditeurs.
- L'éditeur **vi** peut être exécuté à la fois dans une interface CLI (interface de ligne de commande) et une interface graphique (interface utilisateur graphique).
- Alors que de nouvelles fonctionnalités ont été ajoutées à l'éditeur **vi**, les fonctions de base existent depuis des décennies.
- Cela signifie que si quelqu'un a appris l'éditeur **vi** dans les années 1970, il pourrait utiliser une version moderne sans aucun problème. Bien que cela semble peut paraître trivial aujourd'hui, cela ne le sera pas dans vingt ans.

En réalité, la plupart des systèmes Linux n'incluent pas le **vi** original, mais une version améliorée de celui-ci connue sous le nom de **vim**, pour *vi améliorée*. L'éditeur **vim** est ainsi disponible pour la plupart des distributions Linux. En réalité, **vim** fonctionne comme **vi**, mais comporte des fonctionnalités supplémentaires. Pour les sujets qui seront abordés dans ce cours, **vi** ou **vim** suffit.

Pour commencer à utiliser **vi**, tapez simplement la commande suivie du chemin d'accès au fichier à modifier ou créer :

```
sysadmin@localhost:~$ vi newfile.txt
```

Trois modes sont utilisés dans **vi** : le mode commande, le mode insertion et le mode enregistrement.

# Syntaxe des commandes de base

- Mode commande

Initialement, le programme démarre en mode commande. Le mode Commande permet de taper des commandes, telles que celles utilisées pour déplacer un document, manipuler du texte et accéder aux deux autres modes. Pour repasser en mode commande à tout moment, appuyez sur la touche **Echap**. Une fois que du texte a été ajouté dans un document, pour effectuer des actions comme le déplacement du curseur, la touche **Echap** doit d'abord être pressée pour revenir en mode commande. Cela semble être beaucoup de travail, mais rappelez-vous que `vi` fonctionne dans un environnement terminal où une souris ne vous sera d'aucune utilité.

Les commandes de déplacement dans `vi` ont deux aspects, un mouvement et un préfixe numérique facultatif, qui indique combien de fois répéter ce mouvement. Le format général est le suivant :

```
[nombre] motion
```

Le tableau suivant résume les touches de mouvement disponibles :

# Syntaxe des commandes de base

Le tableau suivant résume les touches de mouvement disponibles :

Mouvement	Résultat
h	Un caractère à gauche
j	Descendre d'une ligne
k	Monter d'une ligne
l	Un caractère à droite
w	Un mot en avant
b	Un mot en arrière
^	Début de ligne
\$	Fin de ligne

Depuis la mise à niveau vers **vim** , il est également possible d'utiliser les touches fléchées ←↓↑→ au lieu de hjkl respectivement

# Syntaxe des commandes de base

- Mode actions

La convention standard pour l'édition de contenu dans les traitements de texte consiste à utiliser copier, couper et coller. Le programme **vi** n'en comporte aucun. Au lieu de cela, **vi** utilise les trois commandes suivantes :

Standard	Vi	Signification
cut	d	supprimer (delete)
copy	y	arracher (yank)
paste	P   p	mettre (put)

Les déplacements décrits dans la page précédente sont utilisés pour spécifier où l'action doit avoir lieu, toujours en commençant par l'emplacement actuel du curseur. Les des formats généraux suivants pour les commandes d'action sont acceptés

```
action [nombre] motion
```

```
[nombre] action motion
```



# Syntaxe des commandes de base

- Supprimer (Cut)

- *Supprimer* supprime le texte indiqué de la page et l'enregistre dans le tampon, le tampon étant l'équivalent du « presse-papiers » utilisé dans Windows ou Mac OSX. Le tableau suivant fournit des exemples d'utilisation courants :

Action	Résultat
dd	Supprimer la ligne courante
3dd	Supprimer les trois prochaines lignes
dw	Supprimer le mot courant
d3w	Supprimer les trois prochains mots
d4h	Supprimer quatre caractères à gauche

# Syntaxe des commandes de base

- Modifier

- *Modifier* est très similaire à supprimer ; le texte est supprimé et enregistré dans le tampon, cependant, le programme est commuté en mode insertion pour permettre des modifications immédiates au texte. Le tableau suivant fournit des exemples d'utilisation courants :

Action	Résultat
cc	Modifier la ligne courante
cw	Modifier le mot courant
c3w	Modifier les trois prochains mots
c5h	Modifier cinq caractères vers la gauche

# Syntaxe des commandes de base

- Arracher (Yank)
- *Yank* place le contenu dans le tampon sans le supprimer. Le tableau suivant fournit des exemples d'utilisation courants:

Action	Résultat
yy	Copier la ligne
3yy	Copier les trois prochaines lignes
yw	Copier le mot courant
y\$	Copier jusqu'au bout de la ligne

# Syntaxe des commandes de base

- Mettre (Put)

- *Put* met le texte enregistré dans le tampon avant ou après la position du curseur. Notez que ce sont les deux seules options, coller n'utilise pas les mouvements comme les commandes d'action précédentes.

Action	Résultat
p	Coller après le curseur
P	Coller avant le curseur

# Syntaxe des commandes de base

- Recherche dans vi

Une autre fonction standard offerte par les traitements de texte est la recherche. Souvent, les gens utilisent **CTRL + F** ou regardent sous le menu d'édition. Le programme **vi** utilise la recherche. La recherche est puissante car elle prend en charge à la fois les motifs de texte littéral et les expressions régulières.

Pour effectuer une recherche à partir de la position actuelle du curseur, utilisez le **/** pour lancer la recherche, tapez un terme de recherche, puis appuyez sur la touche **Entrée** pour commencer la recherche. Le curseur se déplace vers la première correspondance trouvée.

Pour passer à la correspondance suivante en utilisant le même motif, appuyez sur la touche **n** . Pour revenir à une correspondance précédente, appuyez sur la touche **N**. Si la fin ou le début du document est atteint, la recherche retourne automatiquement de l'autre côté du document.

Pour commencer à rechercher en arrière à partir de la position du curseur, commencez par taper **?**, puis tapez le motif à rechercher et appuyez sur la touche **Entrée**.

# Syntaxe des commandes de base

- Mode insertion

Le mode Insertion permet d'ajouter du texte au document. Il y a plusieurs façons de passer en mode insertion à partir du mode commande, chacune différenciée par l'endroit où l'insertion de texte commencera. Le tableau suivant couvre les plus courants :

Saisie	Résultat
a	Entre en mode insertion juste après le curseur
A	Entre en mode insertion à la fin de la ligne
i	Entre en mode insertion juste avant le curseur
I	Entre en mode insertion au début de la ligne
o	Entre en mode insertion sur une ligne blanche après le curseur
O	Entre en mode insertion sur une ligne blanche avant le curseur

# Syntaxe des commandes de base

- Mode ex

À l'origine, l'éditeur **vi** s'appelait **ex**. Le nom **vi** était l'abréviation de la commande **visuel** dans l'éditeur **ex** qui a basculé l'éditeur en mode « visuel ».

Originellement, en mode normal, l'éditeur **ex** autorisait uniquement les utilisateurs à voir et à modifier une ligne à la fois. En mode visuel, les utilisateurs peuvent voir autant du document que le permet la taille de l'écran. Comme la plupart des utilisateurs préféraient le mode visuel au mode d'édition de ligne, le fichier programme **ex** était lié à un fichier **vi**, de sorte que les utilisateurs pouvaient démarrer **ex** directement en mode visuel lorsqu'ils exécutaient l'éditeur **vi**.

Au final, le fichier programme réel a été renommé **vi** et l'éditeur **ex** est devenu un lien pointant vers l'éditeur **vi**.

Lorsque le mode ex de l'éditeur **vi** est utilisé, il est possible d'afficher ou de modifier les paramètres, ainsi que d'exécuter des commandes liées aux fichiers comme l'ouverture, l'enregistrement ou l'abandon des modifications apportées à un fichier. Pour accéder au mode ex, tapez le caractère **:** en mode commande. Le tableau suivant répertorie certaines actions courantes effectuées en mode ex :

# Syntaxe des commandes de base

Saisie	Résultat
:w	Ecrire le fichier courant dans le système de fichiers
:w <i>nom_de_fichier</i>	Enregistrer une copie du fichier actuel avec le nom du fichier spécifié
:w!	Forcer l'écriture dans le fichier courant
:1	Aller à la ligne numéro 1 ou n'importe quel numéro suivant
:e <i>nom_de_fichier</i>	Ouvrir le fichier dont le nom est spécifié
:q	Quitter si aucun changement n'a été apporté au fichier
:q!	Quitter sans enregistrer les modifications dans le fichier



# Syntaxe des commandes de base

Une analyse rapide du tableau ci-dessus révèle que si un point d'exclamation ! est ajouté à une commande, elle tente ensuite de forcer l'opération. Par exemple, imaginez que vous apportez des modifications à un fichier dans l'éditeur **vi**, puis essayez de quitter avec :q, seulement pour découvrir que la commande échoue. L'éditeur **vi** ne veut pas quitter sans enregistrer les modifications que vous avez apportées à un fichier, mais vous pouvez le forcer à quitter avec la commande ex :q!.

# Archivage et compression

- L'archivage des fichiers est utilisé lorsqu'un ou plusieurs fichiers doivent être transmis ou stockés aussi efficacement que possible
- Archivage : Combine plusieurs fichiers en un seul, ce qui élimine la surcharge dans les fichiers individuels et rend les fichiers plus faciles à transmettre.
- Compression : Réduit la taille des fichiers en supprimant les informations redondantes.
- Différence entre archivage et compression :
- Les fichiers peuvent être compressés individuellement ou plusieurs fichiers peuvent être combinés en une seule archive, puis compressés par la suite. Ce dernier est encore appelé archivage.
- Lorsqu'une archive est décompressée et qu'un ou plusieurs fichiers sont extraits, cela s'appelle la décompression.

# Archivage et compression

- La compression réduit la quantité de données nécessaires pour stocker ou transmettre un fichier tout en le stockant de manière à ce que le fichier puisse être restauré.
- L'algorithme de compression est une procédure que l'ordinateur utilise pour encoder le fichier d'origine et, par conséquent, le rendre plus petit.
- Deux types de compression:
- Compression sans perte : Aucune information n'est supprimée du fichier. La compression et la décompression d'un fichier laissent quelque chose d'identique à l'original.
- CompressionPerte : Les informations peuvent être supprimées du fichier. Il est compressé de telle sorte que la décompression d'un fichier entraînera un fichier légèrement différent de l'original. Par exemple, une image avec deux nuances subtilement différentes de vert peut être réduite en traitant ces deux nuances comme les mêmes. Souvent, l'œil ne peut pas distinguer la différence de toute façon.

# Archivage et compression

- La compression avec perte profite souvent aux médias car elle permet de réduire la taille des fichiers et les gens ne peuvent pas faire la différence entre l'original et la version avec les données modifiées.
- Pour les éléments qui doivent rester intacts, tels que les documents, les journaux et les logiciels, vous avez besoin d'une compression sans perte.
- La plupart des formats d'image, tels que GIF, PNG et JPEG, implémentent une certaine forme de compression.
- Les JPEG utilisent la compression avec perte, tandis que les GIF et les PNG sont compressés mais sans perte.
- La compression d'un fichier déjà compressé ne le rendra pas plus petit.
- si vous compressez et décompressez un fichier plusieurs fois en utilisant un algorithme avec perte, vous finirez par avoir quelque chose qui est méconnaissable.

# Archivage et compression

- Linux fournit plusieurs outils pour compresser les fichiers ; le plus courant est gzip. Ici, nous montrons un fichier avant et après compression :

```
sysadmin@localhost:~$ cd Documents
sysadmin@localhost:~/Documents$ ls -l longfile*
-rw-r--r-- 1 sysadmin sysadmin 66540 Dec 20 2017 longfile.txt
sysadmin@localhost:~/Documents$ gzip longfile.txt
sysadmin@localhost:~/Documents$ ls -l longfile*
-rw-r--r-- 1 sysadmin sysadmin 341 Dec 20 2017 longfile.txt.gz
```

- Gzip -l permet d'afficher le taux de compression
- Le format décompressé d'un fichier s'appelle fichier long
- Pour décompresser un fichier : gunzip ou gzip -d

# Archivage et compression

- Si vous aviez plusieurs fichiers à envoyer à quelqu'un, vous pourriez choisir de compresser chacun individuellement. Vous auriez une plus petite quantité de données au total que si vous envoyiez des fichiers non compressés, cependant, vous auriez encore à traiter de nombreux fichiers à la fois.
- L'archivage est la solution à ce problème. L'utilitaire UNIX traditionnel pour archiver des fichiers est appelé tar, qui est une forme courte de TApe aRchive. Il a été utilisé pour diffuser de nombreux fichiers sur une bande pour les sauvegardes ou le transfert de fichiers.
- La commande tar prend plusieurs fichiers et crée un seul fichier de sortie qui peut être divisé à nouveau dans les fichiers d'origine à l'autre extrémité de la transmission.

# Archivage et compression

- La commande tar comporte trois modes utiles pour se familiariser avec :
- Créer : Créez une nouvelle archive à partir d'une série de fichiers.
- Extraire : Extraire un ou plusieurs fichiers d'une archive.
- Liste : Affiche le contenu de l'archive sans l'extraire.

```
tar -c [-f ARCHIVE] [OPTIONS] [FILE...]
```

Option	Function
-c	Create an archive.
-f ARCHIVE	Use archive file. The argument ARCHIVE will be the name of the resulting archive file.

# Archivage et compression

- Exemple

```
sysadmin@localhost:~/Documents$ tar -cf alpha_files.tar
sysadmin@localhost:~/Documents$ ls -l alpha_files.tar
-rw-rw-r-- 1 sysadmin sysadmin 10240 Oct 31 17:07 alpha_files.tar
```

La taille finale d'alpha\_files.tar est de 10240 octets. Normalement, les fichiers tarball sont légèrement plus grands que les fichiers d'entrée combinés en raison des informations sur la recreation des fichiers originaux. Les Tarballs peuvent être compressées pour faciliter le transport, soit en utilisant gzip sur l'archive, soit en demandant à tar de le faire avec l'option -z.

Option	Function
-z	Compress (or decompress) an archive using the gzip command.



# Archivage et compression

- L'exemple suivant montre la même commande que l'exemple précédent, mais avec l'ajout de l'option -z.

```
sysadmin@localhost:~/Documents$ tar -czf alpha_files.tar.gz
sysadmin@localhost:~/Documents$ ls -l alpha_files.tar.gz
-rw-rw-r-- 1 sysadmin sysadmin 417 Oct 31 17:15 alpha_files.tar.gz
```

- La sortie est beaucoup plus petite que le tarball lui-même, et le fichier résultant est compatible avec gzip, qui peut être utilisé pour afficher les détails de compression. Le fichier non compressé est de la même taille que si vous le mettiez en attente à une étape distincte :

```
ysadmin@localhost:~/Documents$ gzip -l alpha_files.tar.gz
compressed uncompressed ratio uncompressed_name
417 10240 96.1% alpha_files.tar
```

Bien que les extensions de fichiers n'affectent pas la façon dont un fichier est traité, la convention est d'utiliser .tar pour les tarballs, et .tar.gz ou .tgz pour les tarballs compressés.