**Notions de base  
(Arborescence du système de fichiers, commandes de base, chemins absolus/relatifs, documentation, ...)**

Afin de bien utiliser Linux et d’éviter une perte de temps et de commettre des erreurs, il est important de comprendre les concepts fondamentaux et les principes de base du système d’exploitation Linux.

**Terminologie**

**Terminal**Dans la grande majorité des cas, les serveurs Linux sont dépourvus d'interfaces graphiques.   
Pour interagir avec le système et le matériel, un administrateur Linux utilise alors un terminal.   
Un terminal est un point d’accès de communication entre un utilisateur et un système Linux.

**Hyperviseur**  
Il est commun de virtualiser du matériel informatique sur un seul équipement.   
Ainsi, à l’aide d’un hyperviseur (VMware, Xen, …), il est possible d'héberger plusieurs machines virtuelles.

**Connexion SSH et VNC**Il est usuel d’utiliser un logiciel pour émuler (ou simuler) l'équipement terminal physique et toutes ses fonctionnalités.  
Ce logiciel est souvent un programme qui permet de se connecter localement ou à distance sur un serveur.   
Il est alors possible de lancer plusieurs terminaux simultanément depuis le même environnement.

Les protocoles SSH (*Secure Shell*) ou VNC (*Virtual Network Computing*) permettent notamment de prendre la main à distance sur un ordinateur.   
SSH est un protocole de terminal virtuel texte.  
VNC est un protocole de terminal virtuel graphique.

**Console**

Une console, pour Linux, est un périphérique gérant le clavier et l'écran d'un ordinateur et qui permet d'interagir avec l'utilisateur à l’aide d’un terminal en mode texte.

La console de Linux propose 7 terminaux en mode texte, appelés aussi les terminaux physiques.   
Ils sont directement sur le clavier branché à l'ordinateur et disponibles à partir des combinaisons de touches **…**

* **CTRL+ALT+F1** ;
* **CtRL+ALT+F2** ;
* …   
  jusqu’à **CTRL+ALT+F7**.

Chacune de ces combinaisons de touches propose l'émulation d'un terminal (en mode console) différent sur lequel il est possible de se connecter de manière indépendante avec un compte utilisateur différent.

**Émulateurs de terminal**

Dans la grande majorité des cas, on est connecté à distance sur un serveur Linux à l’aide d’un émulateur de terminal.   
Il s’agit d’un programme lancé depuis un poste de travail Windows/Mac ou même Linux.   
Il gère la connexion au serveur distant avec un protocole réseau (telnet, rlogin, SSH ou VNC).

**Remarque** …  
Il est fortement déconseiller de se connecter sur un système Linux à l’aide des protocoles telnet ou rlogin.  
Ces protocoles sont désuets et incluent des importantes brèches de sécurité.

En fonction du système d'exploitation installé sur le poste de travail, on a le choix entre différents émulateurs offrant diverses fonctionnalités.

**PuTTY ou MobaXterm sous Windows**PuTTY et MobaXterm sont à la fois un émulateur de terminal et un client pour différents protocoles réseau tels que telnet, SFTP, SSH, rlogin, et TCP.   
Putty est disponible gratuitement (open source) et MobaXterm propose une version gratuite avec utilisation limités.

**Terminal sous macOS**MacOS étant un dérivé de la branche historique d’Unix (BSD – *Berkeley Software Distribution*), l'émulateur de terminal est natif : l'application est livrée par défaut avec le système.

**Interface graphique**

Il existe également des interfaces graphiques permettant de communiquer graphiquement à un serveur Linux.

**GNOME**C’est l’interface graphique classique qui est livré avec la gestion de fenêtres, d’onglets, objets de bureau et fonctions de glisser/déplacer (*drag and drop*).

**KDE**KDE est également un interface graphique populaire présent sur plusieurs système Linux.

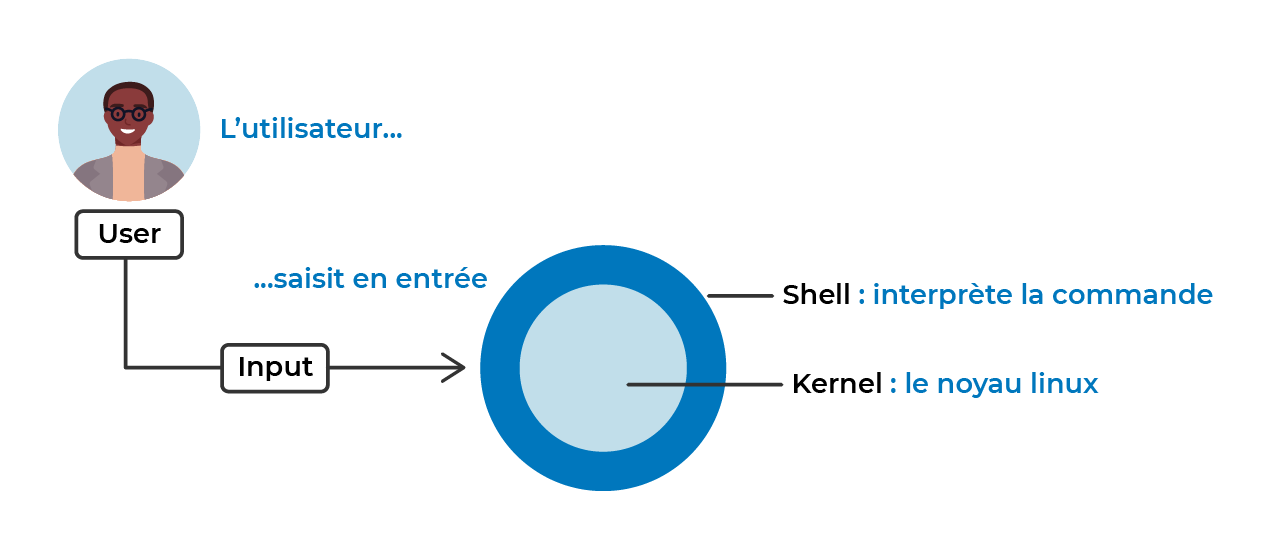
**Autres interfaces graphiques**Parmi les plus utilisés, on retrouve XFCE, Cinnamon, Deepin, ….

**Interpréteur de commandes**

L’interpréteur de commandes (souvent désigné comme *shell*) est l'outil de communication privilégié entre un administrateur et le système.

L’interpréteur de commande est un programme qui est exécuté lors de la connexion d'un utilisateur sur une console ou un terminal.   
Il présente une interface en mode texte qui permet de saisir des commandes.

Le terme shell (signifiant coquille en français) désigne la première couche logicielle entourant le noyau de linux, comme une coquille entoure une noix.



L’interprétteur de commande étant un binaire (ou programme) en soi, il dispose de son propre protocole de communication.  
Un administrateur se doit de saisir les commandes selon un format qui est adapté et compréhensible par cet interpréteur de commandes.

Le principal rôle de l’interpréteur de commande est d'exécuter les commandes saisies par un utilisateur lui permettant d'effectuer des appels systèmes vers le noyau.

Toutefois, il fournit également d'autres fonctionnalités essentielles permettant de faciliter la tâche de l'administrateur.   
Il s’avère fort utile car il offre la possibilité de traiter, de manière automatique, les résultats des commandes saisies, ou encore de rediriger ces informations dans des fichiers ou vers d'autres commandes.

**Principaux shells sous Linux**

Le premier interpréteur de commande apparaît dans les années 70, suivant ainsi la création des premiers systèmes Unix.   
Ken Thompson, l'un des créateurs système d’exploitation Unix, propose son propre shell en 1971 et le nomme Thompson shell.

En 1977, Stephen Bourne écrit le Bourne shell qui devient une référence en matière d'interpréteur de commandes, avec des fonctionnalités très intéressantes dont les plus notables sont …

* L'écriture de fichiers contenant des suites de commandes, les scripts sh (sh étant un préfixe pour shell) ;
* L'intégration d'un langage de programmation, avec des structures conditionnelles, itératives et la gestion des variables ;
* La gestion des canaux d'entrée/sortie et des redirections à l’aide de tubes ;
* La gestion de l'exécution de commandes en substitution avec les caractères spéciaux ;
* ...

D'autres interpréteur de commande sont venus ensuite compléter la famille des interpréteurs de commandes pour systèmes Unix/Linux.  
On peut citer …

* le C Shell, qui dispose d'un langage très proche du langage C (le langage natif des systèmes Unix/Linux) ;
* le Korn Shell, développé plus tard, début des années 80, propose une amélioration du C Shell avec une intégration d'autres langages (comme Perl), le support de tableaux associatifs, ...

Toutefois, l’interpréteur de commande standard est le BASH ou bash (pour *Bourne Again Shell*).   
Développé plutôt fin des années 80, cet interpréteur de commandes est un peu un condensé du meilleur de tous ses ancêtres.  
Il est aujourd’hui l’interpréteur de commande par défaut pour la plupart des distributions.

**Bash, shell standard de Linux**

L’interpréteur de commande est démarré lors de la connexion d'un utilisateur sur un terminal.  
L’interpréteur de commande à charger est précisé pour chaque utilisateur dans le fichier /etc/passwd.

Il est ainsi possible d’afficher l’interpréteur de commande utilisé par un utilisateur donné avec la commande …  
**>> grep <Nom utilisateur> /etc/password  
>> grep tux /etc/password**

Une image contenant texte

Description générée automatiquement

L’interpréteur de commande exécuté à la connexion de l'utilisateur est indiqué dans le dernier champ (les champs sont séparés par des :).

**Remarque** …  
On peut modifier le fichier /etc/passwd en indiquant un interpréteur de commandes comme /usr/bin/nologin ou /dev/zero ou encore /dev/null.  
Cette modification garantit que l'utilisateur ne pourra jamais lancer un interpréteur de commande à la suite du processus d'authentification lors de la connexion.

Bash est un programme écrit en C, visant à respecter au maximum les standards POSIX concernant les interpréteurs de commandes.   
Il est livré en standard sur la plupart des distributions Linux SysV/systemd.

**Remarque** …  
Linux est plutôt du genre à ne rien imposer.   
Il est habituellement recommandé d’utiliser bash sur un système Linux car il est celui qui est proposé par défaut par la plupart des distributions.

Afin de connaître son interpréteur de commande, il est possible de saisir la commande suivante …  
**>> echo $SHELL**

Une image contenant texte

Description générée automatiquement

L’interpréteur de commandes bash propose également d'autres fonctionnalités intéressantes …

* **Auto-complétion (*command line completion*)**Ce mécanisme permet à l'interpréteur de commandes de compléter automatiquement une saisie à partir des premiers caractères saisis.   
  En utilisant la touche TAB lors de la saisie d’une commande, bash essaie de compléter la saisie en fonction de son sens …  
  un chemin, une commande, un argument, une option, …
* **Manipulation des signaux (*signal handling*)**Les signaux sont une forme de communication entre les processus.   
  Ils permettent d'émettre et de recevoir un ordre ou une information pendant leur exécution.   
  L'exemple le plus connu est sûrement l'interruption clavier composée de la frappe des touches **CTRL+c** , combinaison de touche qui permet d'émettre le signal **sigint** (numéro 2).   
  Si le processus écoute sur ce signal, il reçoit alors l'ordre d'interruption et se termine proprement.
* **Code de sortie (*exit code*)**L'interpréteur de commandes bash permet de récupérer le code de sortie d'un programme dans une variable nommée $?.  
  Ce mécanisme est très pratique afin d’interpréter le code et déterminer dans quelle condition le programme s'est terminé.

**Invite de commande**



Lors de l’utilisation d’un système d’exploitation, il faut donc utiliser un interpréteur de commandes (*shell*) – une interface qui donne accès aux services du système.

Pour la gestion d’un serveur, il est recommandé d’utiliser une interface en ligne de commande (*Command Line Interface* ou CLI). Cette dernière est plus puissante et plus efficace qu’une interface graphique (*Graphic User Interface* ou GUI).

En saisissant des commandes texte, on peut effectuer les tâches qui nécessitent un processus en plusieurs étapes à l’aide d’un interface graphique, en quelques secondes seulement.

L'invite de commande (*prompt*) BASH par défaut est ce qui est affiché lors de l’ouverture d’un terminal ou d’une console.  
Elle se présente habituellement à quelque chose comme ceci …



De manière générale, l’invite de commande se termine par un des deux caractères suivants …

* **$** indique que l'utilisateur courant est un utilisateur normal (sans privilège) ;
* **#** indique que l’utilisateur est root (le super utilisateur) .

Afin de connaître les paramètres de l’invite de commandes, il est possible d’afficher la valeur de la variable Bash $PS1.  
Elle correspond à la valeur que doit prendre l’invite de commandes lorsqu’elle est affichée.

Pour afficher la valeur de l’invite de commande …  
**>> echo $PS1**

De manière générale, la sortie affiche des caractères comme [, ], @, et $ ainsi que des caractères d’échappement comme …

* **\u** (qui représente le nom de l’utilisateur) ;
* **\h** (qui représente le nom de l’hôte)   
  et
* **\W** (qui représente le nom du répertoire dans lequel on se trouve, le répertoire de travail).

Il est possible de personnaliser l’invite de commandes avec les caractères d’échappement suivants en modifiant la variable PS1 ...  
**>> PS1=<Caractères d’échappement >**

* \t indique l’heure ;
* \d indique la date ;
* \h indique le nom de l’hôte ;
* \\ o indique donne une barre oblique (*slash*) ;
* \r indique un retour à la ligne ;
* \u indique le nom de l’utilisateur ;
* \v indique la version de Bash ;
* \w donne le chemin complet par rapport à la racine (*root*) et le nom du répertoire courant ;
* \W indique uniquement le répertoire courant

**Remarque** …  
Il est possible de trouver la liste des autres caractères d’échappement sur le howto Bash-Prompt-HOWTO …  
http://www.traduc.org/docs/howto/lecture/Bash-Prompt-HOWTO.html#BASH-PROMPT-ESCAPE-SEQUENCES

Voici quelques exemples d’invite de commandes personnalisée …  
**>> export PS1="\u >"  
>> export PS1="\u\H "  
>> export PS1="\u@\H :"  
>> export PS1="\u >\s\v"  
>> export PS1="\u@\H>\A "  
>> export PS1="\u@\H>\d "  
>> export PS1="\u@\H>\t "  
>> export PS1="\W > "**

On peut également modifier la couleur du texte de l’invite de commandes.   
Pour modifier le texte de l’invite de commandes en vert …  
**>> export PS1="\e[0;32m[\u@\h \W]\$ \e[0m"**

L’invite de commande devrait avoir le même texte que la normale mais être colorée en vert.

Voici le détail de la commande précédente …

* **\e[**  
  Débute la modification de couleur ;
* **0;32m**Spécifie le code de couleur ;
* **[\u@\h \W]\$**Spécifie l’invite de commandes habituelle avec Bash  
  (<Nom d'utilisateur>@<Nom d’hôte>. <Répertoire de travail> et $)
* **\e[0m**  
  Quitte le mode de changement de couleur

La première valeur du code de couleur indique la police de caractères …

* **0** – Normal ;
* **1** - Gras (brillant) ;
* **2** – Dim ;
* **4** – Souligné.

La seconde valeur indique la couleur souhaitée …

* **30** - Noir
* **31** - Rouge
* **32** - Vert
* **33** - Marron
* **34** - Bleu
* **35** - Pourpre
* **36** - Cyan
* **37** - Gris clair

**Remarques** …  
Afin de rendre permanent le changement des variables comme PATH, PS1, …, l’interpréteur de commandes lit un fichier de configuration pour déterminer comment il doit se comporter.   
Ce fichier se trouve dans le répertoire personnel de chaque utilisateur …  
**~/.bashrc**Avant de procéder à des modifications, il est important de créer une copie de sauvegarde de ce fichier de configuration.

**Types de commandes Linux**

Il existe deux types de commandes sous Linux …

**Commandes internes**Les commandes internes sont livrées par l'interpréteur de commandes directement, comme une primitive.   
Elles sont disponibles à tout moment pour un utilisateur du système.

**Commandes externes**Ce type de commandes sont externes à l'interpréteur de commandes ; elles ne sont pas fournies par l’interpréteur de commande.   
Elles sont généralement présentes sur le système sous la forme d'un fichier binaire compilé ou d'un fichier disposant des droits d'exécution (comme un script).

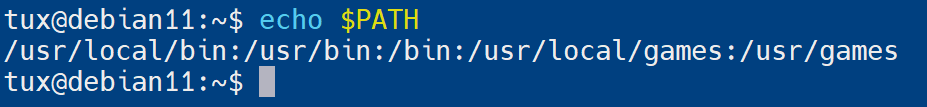
Il est possible de connaître le type de commandes à l’aide de la commande type …  
**>> type <Commande>  
>> type cd  
>> type id  
>> type file**

Une image contenant texte

Description générée automatiquement

L’interpréteur de commandes Bash peut accéder aux commandes externes.  
Pour y arriver, il utilise la variable d'environnement PATH.

Pour afficher et interpréter le contenu de la variable PATH, on peut utiliser la commande echo …  
**>> echo <Variable>  
>> echo $PATH**



**Remarque** …  
On doit utiliser le caractère $ afin de préciser que c’est une variable (environnement ou utilisateur).

**Commande file**

La commande file permet d’obtenir des informations sur le type de fichier.  
Elle permet de savoir la source d’un fichier et/ou le binaire qui l’a créé.  
Elle peut s’avérer utile si on possède un fichier dont le type est inconnue ou lorsque l’on a des doutes quant au type de ce fichier.

La syntaxe de base de la commande type est …  
**>> file <Commande>**

**>> file tux.texte  
>> file tux.texte.nc  
>> file /etc/bin/bash**

**Une image contenant texte

Description générée automatiquement**

**Commande stat**

La commande stat affiche des informations sur un fichier.  
Elle permet d’obtenir des informations telles que …

* la taille d’un fichier ;
* les autorisations (permissions) associées ;
* les ID de groupe et de l’utilisateur propriétaire ;
* la date et l’heure de création, d’accès, de modification, … d’un fichier.

La syntaxe de base de la commande stat est …  
**>> stat --options <Nom du ou des fichiers>  
>> stat tux.texte**

Une image contenant texte

Description générée automatiquement

**Documentation**

L'administrateur Linux dispose d'un nombre impressionnant de commandes en tout genre pouvant l'aider à gérer un système Linux.   
Toutefois, un administrateur ne peut pas connaître le fonctionnement de toutes ces commandes par cœur.

La qualité de la documentation Linux est l'une de ses grandes forces ; les sources d’information disponibles sont multiples.

Dans la majorité des cas, il suffit de saisir le nom de la commande et son objectif pour obtenir une liste de résultats probants.

L'avantage principal de cette documentation réside dans le fait qu'elle est accessible directement depuis le terminal, sans accès web.

Afin de consulter la documentation, un système Linux propose principalement deux types de solutions …

**Documentation interne**La plupart des commandes Linux ont une documentation interne,  
Il suffit de préciser la commande dont on souhaite plus d’informations, suivie de l’option --help.  
**>> <Variable> --help  
>> id --help**

Une image contenant texte

Description générée automatiquement

**Documentation externe**

Il existe deux principales commandes (externes) que l’on peut utiliser afin d’obtenir de l’information pour une commande ou encore un objet du système Linux.

Pour de l’information sur une commande, on peut utiliser la commande man …  
**>> man <Commande>  
>> man id**

**Remarque** …  
Pour sortie d’un module d’aide, on appuie sur**q** (pour quitter).

Pour de l’information sur un objet (fichier, procédure, commande, …), on peut utiliser la commande info.  
Il existe toutefois plusieurs catégories d’informations disponibles et il est possible d’effectuer une recherche préalable.  
La commande apropos est utilisée …  
**>> apropos <Objet>  
>> apropos passwd**

Une image contenant texte

Description générée automatiquement

La valeur suivant l’objet de la recherche indique le type (fichier, procédure, commande, …).  
Les types les plus intéressants sont …

* (1) Programmes exécutables ou commandes de l'interpréteur de commandes (shell) ;
* (5) Formats des fichiers et conventions ;
* (8) Commandes de gestion du système (généralement réservées à l’utilisateur root).

En fonction de leur type, la documentation des commandes n’est pas accessible de la même manière.

Pour de l’information spécifique sur un objet …  
 **>> man <Valeur> <Commande>  
>> man 1 passwd  
>> man 5 passwd**

**Une image contenant texte

Description générée automatiquement**

**Une image contenant texte

Description générée automatiquement**

**Rappel** …  
Pour sortie d’un module d’aide, on appuie sur **q** (pour quitter).

**Commandes linux**

Une commande Linux est un programme ou un utilitaire qui s’exécute sur une ligne de commande.   
Pour rappel, une ligne de commande (CLI) est un terminal qui permet à l’utilisateur de saisir des instructions texte et de les faire exécuter par le système.

Le synopsis typique d’une commande typique Linux se présente généralement sous la forme suivante …  
**>> commande [Option(s)] [Argument(s)] ...**

En détails …

* Toute commande débute par un verbe.  
  C’est l’action que doit exécuter la commande ;
* Une commande peut (ou non) posséder une ou des options.  
  Une option permet de modifier ou de préciser le comportement d’une commande.
* Si tel est le cas, le ou les arguments précise l’objet visé par la commande.  
  Cet objet peut être un fichier, un utilisateur, un périphérique, …

**>> id  
>> cat /etc/passwd  
>> cat -n /etc/passwd**ou **>> cat --number /etc/passwd**

**Remarque** …  
Pour la commande **cat**, le commutateur **--number** (forme courte **-n**) permet de numéroter toutes les lignes en sortie.

Il existe pour Linux deux types d’options pour un système Linux…

* les options courtes(forme courte)conformes aux normes Unix …  
  **>> ss -t -u -l -n**et ces options peuvent être concaténées …  
  **>> ss -tuln**
* les options longues spécifiques à Linux …  
  **>> ss --tcp --udp --listening --numeric**

**Remarque** …  
Conformément aux normes POSIX, les options courtes sont sensibles à la casse (MAJUSCULES/minuscules).  
Toutefois, les options longues sont toujours en minuscules.

Les commandes Linux suivent, pour la grande majorité d’entre elles, un certain nombre de critères définis selon les normes POSIX (*Portable Operating System Interface* – *Unix*).   
On peut retrouver ces normes ou standards dans un document de normalisation POSIX nommé Utility Conventions.

On retrouve parmi ces normes que les commandes doivent être définies selon ce qui suit …

* le nom d’une commande doit comporter entre 2 et 9 caractères ;
* le nom d’une commande doit comporter uniquement des caractères minuscules et des chiffres ;
* les options d’une commande doivent être définies avec un seul caractère alphanumérique ;
* les options d’une commande doivent être précédées du caractère - ;
* …

**Remarque** …  
Tout en demeurant conforme aux normes POSIX, Linux propose des normes complémentaires telles que les options longues.

**Sensibilité à la casse**

Afin d’être conforme aux normes POSIX, Linux fait une différence entre majuscule et minuscule.  
Tous les éléments d’une commande Linux (commande, options et arguments) respecte rigoureusement cette règle.

Par exemple, le noms de fichier ou répertoires sont sensibles à la casse.  
Pour Linux, tux.texte, Tux.Texte et TUX.TEXTE sont trois fichiers différents.

**Alias**

Un alias est un raccourci qui fait référence à une commande.   
Il permet de remplacer une chaîne qui invoque une commande par une autre chaîne définie par l'utilisateur.

Les alias sont principalement utiles pour remplacer des commandes longues.  
Cela permet d'améliorer l'efficacité et d'éviter les éventuelles erreurs d'orthographe.

Les alias peuvent également remplacer des commandes avec des options supplémentaires, ce qui les rend plus faciles à utiliser.

**Syntaxe des alias Linux**

La commande alias utilise la syntaxe suivante …  
**>> alias [Option(s)] [Nom]='[Valeur]'**

Les différents éléments de la syntaxe de la commande alias sont …

* **alias**  
  Invocation de la commande alias ;
* **[Option(s)]**  
  Affichage de la liste de tous les alias actuels ;
* **[Nom]**  
  Définition du nouveau raccourci qui fait référence à une commande.   
  Ce nom est défini par l'utilisateur ;
* **[Valeur]**  
  Spécification de la commande à laquelle l'alias fait référence.   
  Les commandes peuvent également inclure des options, des arguments et des variables.

Il est possible de créer deux types d'alias à créer sous Linux …

* **Temporaires**  
  On les ajoute en utilisant la commande alias ;
* **Permanentes**  
  On doit les ajouter en modifiant les fichiers système.

Exemples d’alias …  
**>> alias cls='clear'  
>> alias ll='ls -ahilrt'  
>> alias ll='ls --all --human-readable --inode -l --reverse -t'  
>> alias srn='systemctl restart networking.service'  
>> alias snn='systemctl status networking.service'  
>> alias sra2='systemctl restart apache2.service'  
>> alias ssa2='systemctl status apache2.service'**

**Remarque** …  
Similaire aux alias, les combinaisons ou raccourcis clavier peuvent également être utilisés.  
Par exemple, **CTRL+L** est le raccourci de la commande **clear**.

**Alias permanent**

Les alias sont définis pour la session en cours seulement.  
Afin de rendre permanent un alias, il faut les ajouter en modifiant les fichiers système

Pour ce faire, on doit modifier le fichier **~/.bash\_aliases** ou **~/.bashrc** (recommandé).  
**>> nano ~/.bash\_aliases**ou  
**>> nano ~/.bashrc**  
**alias update = 'sudo apt update && sudo apt upgrade --yes'**

On enregistre et on ferme le fichier.

On doit activer l'alias à l’aide de la commande **source** ...  
**>> source ~/.bash\_aliases**ou  
**>> source ~/.bashrc**

**Historique des commandes**

La commande **history** est un outil intégré à l’interpréteur de commandes (*shell*) qui affiche une liste des commandes utilisées dans la session du terminal.   
Cette commande permet de réutiliser toute commande listée sans à avoir à la saisir de nouveau.

L’utilisation de la commande **history** sans options ni argument affiche la liste des dernières commandes …  
**>> history**

**Une image contenant texte

Description générée automatiquement**

Pour afficher la liste des commandes avec un nombre limité d'entrées …  
**>> history <Valeur>  
>> history 5**

**Une image contenant texte

Description générée automatiquement**

L'exécution de la commande history permet de réutiliser n'importe quelle commande de la liste ...  
**>> !<Valeur>  
>> !25**

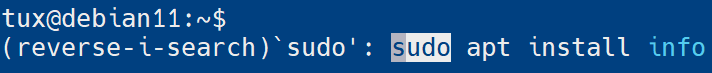
L'ajout d'un tiret (-) devant le numéro de la commande fait commencer le compte à partir de la fin de la liste …  
**>> !-<Valeur>  
>> !-10**

L’utilisation de deux points d'exclamation permet de répéter la dernière commande …  
**>> !!**

**Recherche d'une commande**

L'ajout d'une chaîne après le point d'exclamation permet d'exécuter la dernière commande qui commence par cette chaîne …  
**>> !<Chaine>  
>> !sudo**

Il est aussi possible d’utiliser le raccourci clavier **CTRL+r** pour effectuer une recherche …



**Réutilisation d'une commande**

L'utilisation de l’historique des commandes peut entraîner des problèmes si l’interpréteur de commandes exécute une commande inattendue.  
Par précaution, l'ajout de l'argument :p permet d'afficher la commande sans l'exécuter …  
**>> !<Chaine>:p  
>> !sudo:p**

Pour rechercher une commande qui contient une chaîne mais qui ne commence pas forcément par celle-ci, on peut ajouter un point d'interrogation à côté du point d'exclamation …  
**>> !?<Chaine>  
>> !?echo**

**Remarque** …  
Pour cet exemple, l'interpréteur de commandes réutilise la dernière commande qui contient la chaîne echo.

**Suppression de l'historique**

L'option -d de la commande history permet de supprimer une commande de la liste de l'historique …  
**>> history -d <Valeur>  
>> history -d 25**

L’option -c permet d’effacer toute la liste de l'historique …  
**>> history -c**

**Variable d’environnement HISTORY**

Plusieurs variables d'environnement associées peuvent être utilisés pour obtenir des informations sur le fichier d'historique …

|  |  |
| --- | --- |
| Caractère | Usage |
| HISTFILE | Stocke l'emplacement du fichier d'historique |
| HISTFILESIZE | Stocke le nombre maximum de lignes dans le fichier d’historique |
| HISTSIZE | Stocke le nombre maximum de lignes, de la session en cours, dans le fichier d'historique |

La commande echo permet de visualiser ces variables d’environnement …  
**>> echo $HISTFILE  
>> echo $HISTFILESIZE  
>> echo $** **HISTSIZE**

**Raccourcis clavier**

L’utilisation des raccourcis clavier offre la possibilité d’effectuer rapidement des tâches différentes.

Le tableau suivant énumère quelques-uns de ces raccourcis clavier et leurs utilisations ...

|  |  |
| --- | --- |
| Raccourci clavier | Tâche |
| CTRL+L | Effacer l’écran |
| CTRL+D | Fermer l’interpréteur de commandes courant |
| CTRL+Z | Suspendre le processus en cours et le placer en arrière-plan |
| CTRL+C | Arrêter le processus courant |
| CTRL+R | Rechercher une commande précédemment utilisée (historique) |
| TAB | Compléter la commande en cours de saisie |