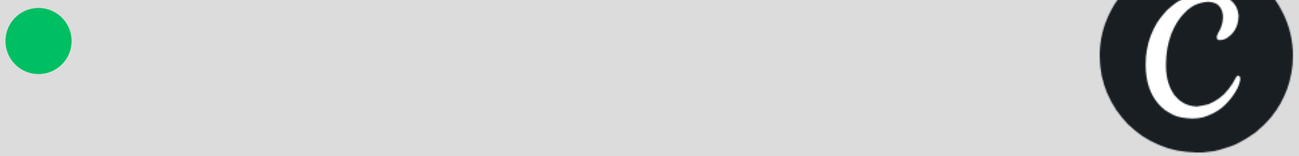
Savoir utiliser un terminal de commandes

Objectif : Savoir utiliser un terminal de commandes pour manipuler ses fichiers et dossiers comme un pro.



Savoir utiliser un terminal de commandes

Programme :

- Qu’est-ce qu’un interprêteur de commandes ?
- Qu’est-ce qu’une commande ?
- Bloqué dans l’invit de commandes ?
- Le système de fichiers
- Comprendre la structure arborescence
- Chemin absolu vs chemin relatif
- C’est quoi Gitbash ?
- Installer Gitbash
- Les commandes de bases de Gitbash
- Exercice
- QCM

Qu’est-ce qu’un interprêteur de commandes ?

C’est un **programme pour exécuter des commandes** tapées au clavier par un utilisateur dans un terminal. Les interpréteurs de commandes sont aussi appelés « **Shells** » (coquilles en anglais). Il existe de nombreux interpréteurs de commandes : sh, bash, csh, ksh, zsh, qui ont chacun des particularités.

Les commandes sont tapées dans un **terminal en mode texte**, constitué d’une fenêtre dans un environnement graphique, ou d’une console sur un écran en texte seul (sans environnement graphique). Pour vous indiquer que l’interpréteur de commandes est prêt à recevoir une commande, il affiche un message en début de ligne appelé **prompt** en anglais. Ce prompt est généralement constitué du **nom de l’utilisateur** puis du **nom de la machine** et enfin du **nom du dossier** où vous vous trouvez, souvent suivi du caractère **$**

Qu’est-ce qu’un interprêteur de commandes ?

Les **résultats** sont aussi affichés sur le terminal. Une sortie graphique n'est pas obligatoire (pour la plupart des commandes la sortie est textuelle).

Un interpréteur de commandes analyse et exécute les commandes que vous tapez. Mais il peut aussi exécuter automatiquement une suite de commandes que vous avez programmée dans un « script Shell ».

Qu’est-ce qu’une commande ?

Une commande Unix est une instruction entrée dans un terminal pour exécuter une action.

Sa syntaxe suit une structure précise :

$ commande -option argument

Elle se compose de :

- La commande (toujours en premier)
- Les options (modifient le comportement, facultatives, précédées de -)
- Les arguments (informations à traiter, facultatifs ou obligatoires)

Qu’est-ce qu’une commande ?

Exemple sans option ni argument pour afficher la date et l’heure actuelles.

- **tapper “cal” dans le terminal**

Exemple avec option pour afficher le calendrier de l’année entière (-y pour year).

- **tapper “cal -y” dans le terminal**

Exemple avec argument pour afficher le calendrier de l’année 2018.

- **tapper “cal 2018” dans le terminal**

Des commandes pour vous aider :

- Trouver une commande par sujet : apropos **calendar**
- Lire le manuel d’une commande : man **calendar**

Il est bien sûr impossible de connaître toutes les commandes par cœur, ni toutes les options et arguments possibles pour chacune des commandes

Bloqué dans l’invit de commandes ?

Quitter le mode éditeur sur l’invite de commandes dépend de l’éditeur utilisé. Si tu es totalement bloqué dans un éditeur inconnu, force la fermeture avec :

CTRL + C

ou

CTRL + Z

Système de fichiers

Le système de fichiers est l’organisation

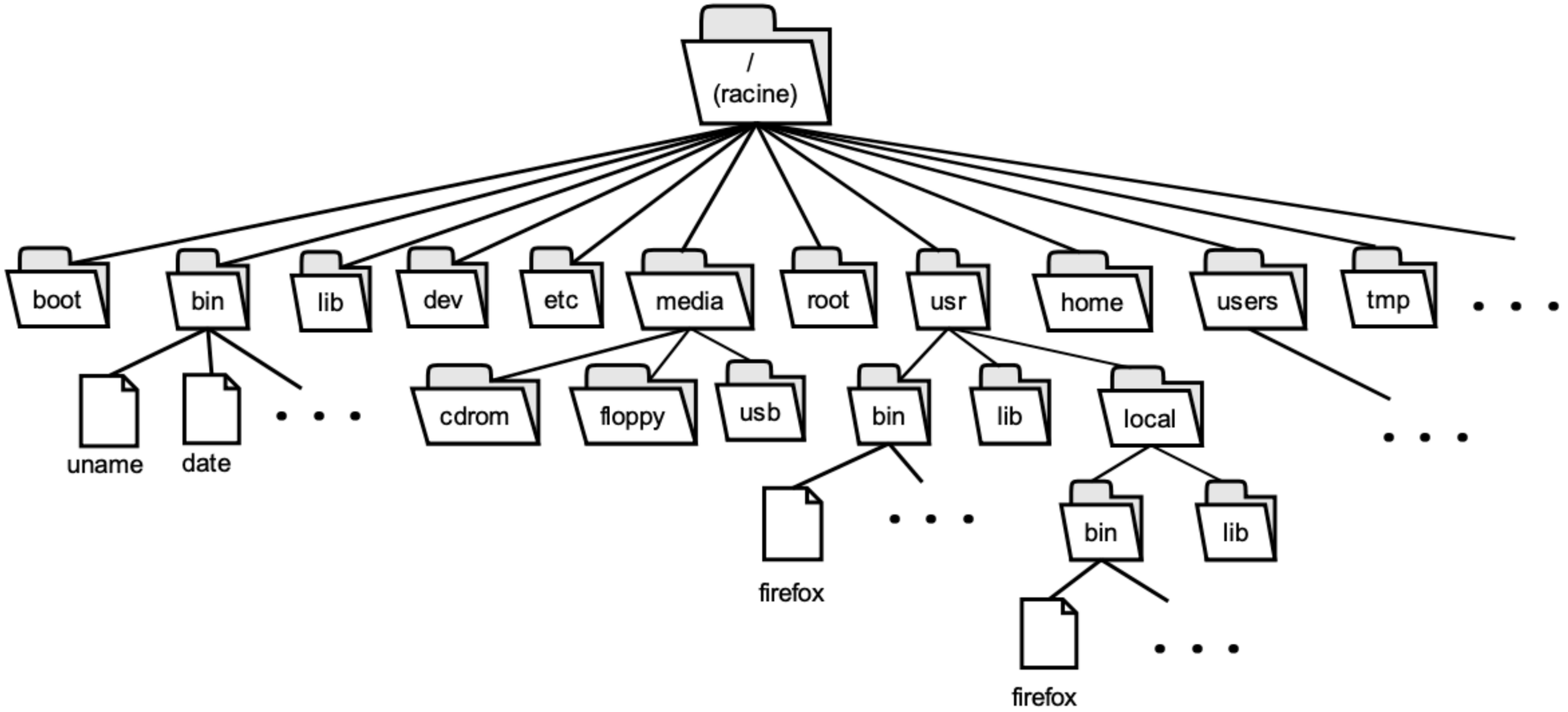
des fichiers et des répertoires sur un ordinateur.

Métaphore : La commode

- Le système de fichiers est comme une grande commode.
- Les tiroirs sont les répertoires (ou dossiers).
- Les documents dans ces tiroirs sont les fichiers.
- Certains tiroirs peuvent contenir d’autres tiroirs → les sous-répertoires.

Exception : la commode elle-même (racine du système) ne peut pas être rangée dans un tiroir !

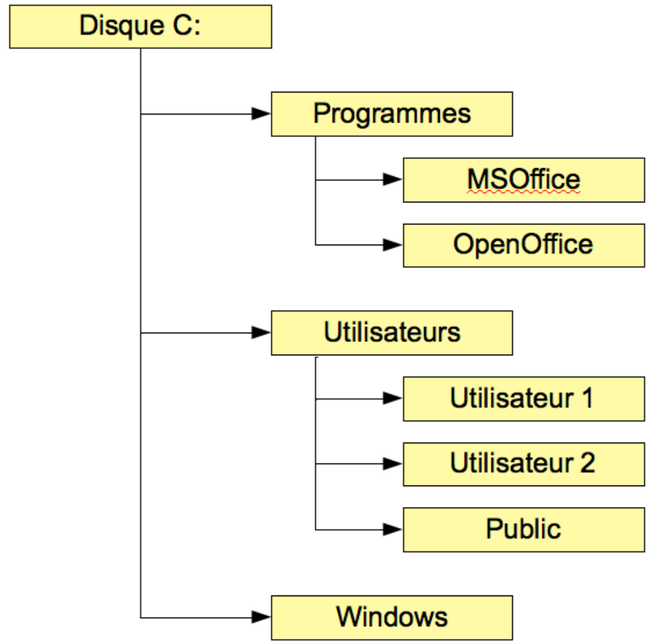
Système de fichiers



Structure arborescente du système de fichiers

Le système de fichiers suit une organisation hiérarchique sous forme d'arbre :

- Les répertoires sont les nœuds de l'arbre.
- Les fichiers sont les feuilles stockées dans ces répertoires.
- La racine est le point de départ.



Sous Unix/Linux : Une seule racine / (Root Directory).
Sous Windows : Plusieurs systèmes de fichiers (ex. C:, D: ...).

L'arborescence d'un disque fonctionne comme un arbre :

- Le disque dur (C: ou D:) est la racine.
- Les dossiers (répertoires) sont des branches.
- Les fichiers sont les feuilles de cet arbre.

Comprendre le chemin absolu et relatif

Comprendre les chemins absolus VS chemins relatifs

Un **chemin absolu** commence toujours par la racine du disque
(C:\ sous Windows, / sous Linux/Mac).
Exemple : C:\Users\MonNom\Documents\mon_projet\index.html
Résumé : c'est le chemin depuis la racine.

Un **chemin relatif** ne mentionne pas la racine (C:\), mais prend en compte la position actuelle.
Exemple : Si je suis dans C:\Users\MonNom\Documents\, alors pour accéder à mon_projet/index.html, je peux taper : “cd mon_projet”
Résumé : c'est le chemin depuis le dossier courant.

C’est quoi Gitbash ?

Git Bash est un terminal (interprêteur de commandes) qui permet d’exécuter des commandes Git et des commandes Unix (Linux/macOS) sous **Windows**. Il est fourni avec Git et est particulièrement utile pour :

- Travailler avec Git en ligne de commande.
- Utiliser des commandes Unix sur Windows.
- Automatiser des tâches et travailler plus efficacement.

Pourquoi ne pas utiliser uniquement l’explorateur de fichiers ?

Git Bash (et les invit de commandes en général) permet d’être plus rapide, plus précis et plus efficace dans la gestion des fichiers et des versions de code.

C’est quoi Gitbash ? (suite)

Pourquoi Git Bash plutôt que l’invite de commandes Windows ?

- Git Bash offre des commandes Unix/Linux : Très utile pour travailler avec des serveurs ou en équipe sur des projets open source.
- Compatible avec Git sans configuration supplémentaire : Contrairement à cmd, où certaines commandes Git ne fonctionnent pas bien.
- Meilleur support des scripts Shell : Pratique pour automatiser certaines tâches (exemple : installation rapide de dépendances)

`Tester une commande Unix (ls -la) : elle ne fonctionne pas dans cmd, mais bien dans Git Bash.`

Installer Gitbash

1) Depuis un fichier d’exécution externe (fonctionne en dehors de VS Code)

Lien de téléchargement : https://git-scm.com/downloads

Ouvrir le fichier téléchargé (Git-2.x.x-64-bit.exe).

- Cocher “Git Bash Here” lors des options d’installation.
- Ne pas modifier les autres options par défaut (y compris celles concernant Git Bash).

2) Depuis VS Code :

- Aller dans Terminal > Nouveau terminal (ou Ctrl + Shift + `)
- Cliquer sur la flèche à droite du terminal et choisir “Sélectionner un profil par défaut”.
- Choisir Git Bash.
- Ouvrir un nouveau terminal (Ctrl + Shift +) et tester avec ls.

Résumé sur la manipulation de fichiers et répertoire

Commande	Explication
pwd	Affiche le chemin du dossier actuel
ls	Liste les fichiers et dossiers
cd <nom_du_dossier>	Se déplacer dans un dossier
cd ../	Revenir en arrière
mkdir <nom_du_dossier>	Créer un dossier
touch <nom_du_fichier>	Créer un fichier
rm <nom_du_fichier>	Supprimer un fichier
rmdir <nom_du_dossier>	Supprimer un dossier vide
cp	Copie un/des fichier(s)/répertoire(s)
mv	Déplace/renomme un/des fichier(s)/répertoire(s)
touch	Créé un fichier vide
ssh	se connecter à un serveur en ssh

Exercice avec le terminal sur Moodle

Ressources

Ressources Markdown

https://www.markdownguide.org/basic-syntax/

Commandes Unix :

https://www.hostinger.fr/tutoriels/commandes-linux

ExplainShell :

https://explainshell.com/

Chemin relatif vs chemin absolu

https://www.youtube.com/watch?v=QFA0VKrQ3zk