

Ligne de commande Unix



Ligne de commande Unix

Vincent Goulet

Professeur titulaire
École d'actuariat, Université Laval

Version 2024.08

© ⓘ ⓘ © 2018-2024 par Vincent Goulet. *Ligne de commande Unix* est mis à disposition sous licence **Attribution-Partage dans les mêmes conditions 4.0 International** de Creative Commons. En vertu de cette licence, vous êtes autorisé à :

- **partager** — copier, distribuer et communiquer le matériel par tous moyens et sous tous formats ;
- **adapter** — remixer, transformer et créer à partir du matériel pour toute utilisation, y compris commerciale.

L'Offrant ne peut retirer les autorisations concédées par la licence tant que vous appliquez les termes de cette licence.

Selon les conditions suivantes :



Attribution — Vous devez créditer l'œuvre, intégrer un lien vers le contrat et indiquer si des modifications ont été effectuées à l'œuvre. Vous devez indiquer ces informations par tous les moyens possibles, mais vous ne pouvez suggérer que l'Offrant vous soutient ou soutient la façon dont vous avez utilisé son œuvre.



Partage dans les mêmes conditions — Dans le cas où vous modifiez, transformez ou créez à partir du matériel composant l'œuvre originale, vous devez diffuser l'œuvre modifiée dans les même conditions, c'est à dire avec le même contrat avec lequel l'œuvre originale a été diffusée.

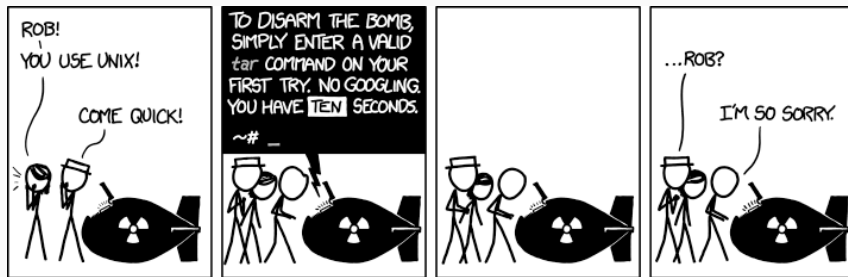
Code source

🔗 [Voir sur GitLab](#)

Couverture

Brebis manech tête rousse sur le Suhalmendi à Ascaïn, dans le Pays basque français. Brebis la plus productive des Pyrénées, le lait de la manech tête rousse est notamment utilisé dans la confection de la tomme d'appellation d'origine contrôlée ossau-iraty. Crédit photo :

© Basotxerri, [CC BY-SA 4.0 International](#), via [Wikimedia Commons](#).



Tiré de [XKCD.com](https://xkcd.com)

Introduction

Navigation dans le système de fichiers

Gestion des fichiers

Transfert de données et redirection

Exécution de procédures

Utiliser la ligne de commande Unix pour effectuer des opérations simples dans le système de fichiers.

Introduction

Un outil puissant toujours pertinent

La ligne de commande du système d'exploitation demeure une interface importante pour les programmeurs.

- Parfois **plus simple** qu'une interface graphique
- Souvent **plus rapide** qu'une interface graphique
- Parfois la **seule option** (notamment pour les utilitaires Unix comme **grep**, **sed**, **awk**)

(Interface en) Ligne de commande

Mode d'interaction avec un programme informatique dans lequel l'utilisateur dicte les commandes et reçoit les réponses de l'ordinateur en mode texte (*command line interface*, CLI)

Interpréteur de commandes

Programme qui gère l'interface en ligne de commande (*terminal*, *shell*)

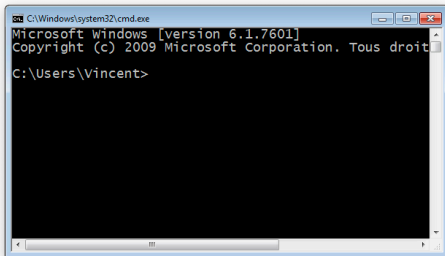
Invite de commande

Symbole affiché par l'interpréteur de commande pour indiquer qu'il est prêt à recevoir une commande (*command prompt*)

Ligne de commande Windows

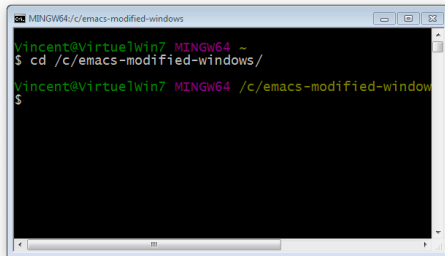
cmd.exe

- Accessoires | Invite de commandes
- pas très puissant
- invite par défaut : **C:\>**



Git Bash ou MSYS2

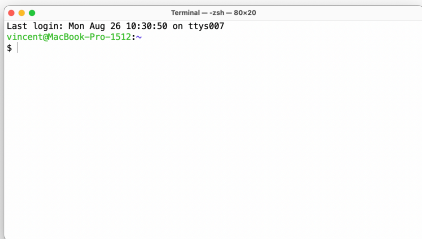
- interpréteur Bash très puissant
- standard Unix
- invite par défaut : **\$**



Ligne de commande macOS

Terminal

- Applications | Utilitaires | Terminal ou via Spotlight
- interpréteur Bash ou Z Shell (depuis macOS 10.15 Catalina)
- invite par défaut : %

A screenshot of a macOS Terminal window. The title bar at the top reads "Terminal — zsh — 80x20". The terminal content shows the login message "Last login: Mon Aug 26 10:30:50 on ttys007", the user prompt "vincent@MacBook-Pro-1512:~", and a dollar sign "\$" followed by a vertical bar "|" as the current prompt.

```
Terminal — zsh — 80x20
Last login: Mon Aug 26 10:30:50 on ttys007
vincent@MacBook-Pro-1512:~
$ |
```



Vous pouvez modifier l'apparence de la ligne de commande dans les préférences de l'application Terminal

Rudiments de la ligne de commande

Nous allons uniquement étudier les commandes qui permettent de :

- se déplacer dans le système de fichiers
- afficher la liste des fichiers d'un répertoire
- afficher le contenu d'un fichier
- copier et supprimer un fichier



Commandes des interpréteurs de commande Unix (de type Bash) seulement.

Quelques Unix-ismes

- Répertoire par défaut est normalement le **répertoire personnel** (*home directory*)

Windows

```
$ printenv HOME  
/c/Users/vincent
```

macOS

```
$ printenv HOME  
/Users/vincent
```

- Répertoire personnel identifié par le symbole « ~ »
- Noms de répertoires séparés par la barre oblique « / »
- Ligne de commande sensible à la casse (**foo** ≠ **Foo** ≠ **F00**)
- Fichiers **.foo** cachés dans la liste des fichiers



Git Bash et MSYS2 utilisent la nomenclature Unix pour identifier les disques et les répertoires.

Windows

C:

C:\Users

C:\Users\vincent

...

Unix

/c

/c/Users

/c/Users/vincent



Gagnez en efficacité au clavier!

↑ | ↓ commande précédente | suivante dans l'historique

→| autocomplétion de la commande, du nom de fichier, etc.
votre meilleure alliée!

Ctrl-R recherche dans l'historique des commandes

Ctrl-K suppression du texte qui suit le curseur

Ctrl-U suppression du texte qui précède le curseur

Il y en a **plusieurs autres** [!\[\]\(642aa997563f9a325b310230bb5078b7_img.jpg\)](#).

1. Démarrer la ligne de commande de votre système d'exploitation
2. Identifier le répertoire courant dans l'invite de commande
3. Repérer ce répertoire dans le système de fichier avec l'Explorateur Windows ou le Finder (garder la fenêtre ouverte)

Navigation dans le système de fichiers

Afficher le répertoire courant

`pwd` (*print working directory*)

- affiche le chemin d'accès absolu du répertoire courant

```
~$ pwd  
/Users/vincent
```

Changer de répertoire

cd (*change directory*)

- change le répertoire courant pour celui donné en argument
- argument peut être un chemin d'accès **absolu** ou **relatif**
- nom fictif « **.** » identifie le répertoire courant
- nom fictif « **..** » identifie le parent du répertoire courant
- sans argument, ramène au répertoire personnel « **~** »

```
~$ cd Desktop/  
/Users/vincent/Desktop  
~/Desktop$ cd ~/Documents/cours/  
~/Documents/cours$ cd ..  
~/Documents$ cd  
~$
```



Si le nom d'un répertoire contient des espaces, il faut les « désactiver » avec la barre oblique inversée : `cd Program\ Files`.



Dès qu'un nom contient des espaces, utilisez la touche `→|` pour compléter le nom. Les symboles `\` seront ajoutés automatiquement.

Lister les fichiers

`ls` (*list*)

- affiche les fichiers du répertoire en argument
- sans argument, affiche les fichiers du répertoire courant

```
~$ ls  
Desktop  
Documents  
Downloads  
<...>
```

Exercice (1/2)

1. Afficher le répertoire courant. Comparer avec celui mentionné dans l'invite de commande.

```
| ~$ pwd
```

2. Afficher la liste des fichiers du répertoire courant. Comparer avec la liste de l'Explorateur Windows ou du Finder.

```
| ~$ ls
```

3. Choisir un sous-répertoire du répertoire courant que vous savez contenir lui-même un sous-répertoire (par exemple : **Documents/Cours**).
4. Afficher, **sans d'abord s'y déplacer**, la liste des fichiers du premier sous-répertoire (**Documents**), puis celle du second (**Cours**).

```
| ~$ ls Documents/Cours
```

⚙ Exercice (2/2)

5. Faire du sous-sous-répertoire ci-dessus (**Cours**) le répertoire courant.

```
| ~$ cd Documents/Cours
```

6. Afficher une liste détaillée des fichiers du nouveau répertoire courant avec la commande **ls -l**.

```
| ~$ ls -l
```

7. Revenir au répertoire personnel avec une seule commande.

```
| ~$ cd ~
```

ou tout simplement

```
| ~$ cd
```

8. Afficher la liste de tous les fichiers du répertoire personnel, y compris les fichiers cachés, avec la commande **ls -a**.

```
| ~$ ls -a
```

Gestion des fichiers

Afficher le contenu d'un fichier

`cat` (*catenate*)

- affiche le contenu du fichier donné en argument
- utile uniquement pour les fichiers en texte brut

```
$ cat hello.txt  
Hello World!
```

cp (*copy*)

- copie le fichier en premier argument vers la destination en second argument
- second argument peut être un nom de répertoire, un nom de fichier ou les deux

```
$ cp hello.txt foo.txt  
$ cp hello.txt Documents/  
$ cp hello.txt Documents/foo.txt
```

Supprimer un fichier

`rm` (*remove*)

- supprime le ou les fichiers en argument
- fichiers disparus à jamais

```
| $ rm foo
```

Exercice (1/2)

Faire de votre répertoire personnel le répertoire courant et créer un fichier (vide) nommé `foobar` avec la commande :

```
$ touch foobar
```

Effectuer les opérations ci-dessous en validant chaque fois leur effet dans l'Explorateur Windows ou dans le Finder.

1. Constater que le fichier `foobar` est vide : sa taille est nulle et l'affichage de son contenu ne retourne rien.

```
$ ls -l foobar
```

```
$ cat foobar
```

2. Copier le fichier `foobar` dans votre répertoire personnel sous le nom `foo.txt`.

```
$ cp foobar foo.txt
```

Exercice (2/2)

3. Copier le fichier `foobar` dans le répertoire `~/Documents`.

```
| $ cp foobar Documents
```

4. Copier le fichier `foobar` dans le répertoire `~/Documents` sous le nom `bar.txt`.

```
| $ cp foobar Documents/bar.txt
```

5. Copier le fichier `~/Documents/bar.txt` dans le répertoire courant. (Astuce : utiliser un nom de répertoire fictif.)

```
| $ cp Documents/bar.txt .
```

6. Supprimer tous les fichiers créés ci-dessus.

```
| $ rm foobar foo.txt bar.txt Documents/bar.txt
```

Transfert de données et redirection

« Ne faire qu'une seule chose, et la faire bien. »

- Écrivez des programmes qui effectuent une seule chose et qui le font bien.
- Écrivez des programmes qui collaborent.
- Écrivez des programmes pour gérer des flux de texte, car c'est une interface universelle.

(Source : Wikipedia)

Les programmes Unix en ligne de commande :

- reçoivent leurs données d'une **entrée standard** (*standard input*, `stdin`)
- écrivent leurs résultats vers la **sortie standard** (*standard output*, `stdout`)
- émettent leurs erreurs vers l'**erreur standard** (*standard error*, `stderr`)



Dans un terminal, l'entrée standard est le clavier et la sortie et l'erreur standards, l'écran

Transfert de données

`stdout` | `stdin` (« tuyau », *pipe*)

- transfère la sortie d'un programme directement à l'entrée d'un autre programme



- similaire à la composition de fonctions en mathématiques

$$(g \circ f)(x) = g(f(x)) \quad \Leftrightarrow \quad x \mid f() \mid g()$$

```
~$ pwd
/Users/vincent
~$ pwd | tr '[a-z]' '[A-Z]'
/USERS/VINCENT
```

Redirection

stdout > *fichier*

- redirige la sortie standard d'un programme vers un fichier



- si le fichier existe déjà, son contenu est écrasé
- variante double >> pour ajouter le contenu à la fin du fichier

```
$ pwd > wd.txt  
$ cat wd.txt  
/Users/vincent
```

Redirection (suite)

stdin < fichier

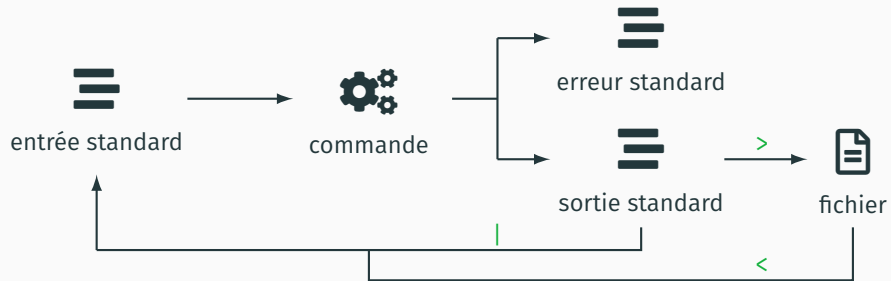
- déverse le contenu d'un fichier dans l'entrée standard d'un programme



- variante double existe aussi

```
$ tr '[a-z]' '[A-Z]' < wd.txt  
/USERS/VINCENT
```

Sommaire graphique



Créer un fichier nommé `foobar` depuis la ligne de commande avec la commande :

```
$ echo 'aaa.bbb.ccc' > foobar
```

Déterminer ensuite le résultat des commandes suivantes.

1. `cat foobar`
2. `cat foobar | cut -d . -f 1`
3. `tr a z < foobar`
4. `ls -la ~ > foobar` (examiner le contenu du fichier avec `cat`)
5. `rm foobar`

Exécution de procédures

Procédures d'interpréteur de commandes

Les interpréteurs de commandes Unix contiennent un langage de programmation afin de créer des procédures — *shell scripts* — pour automatiser des tâches.

- Simples fichiers texte avec sur la première ligne

```
| #!/bin/sh
```

 ou

```
| #!/bin/bash
```

- **Permissions Unix**  du fichier doivent inclure le **droit d'exécution**

```
| $ ls -l bonjour.sh  
| -rwxr--r-- 1 vincent staff 136 8 nov 09:18 bonjour.sh
```

- Commande **chmod** ajoute le droit d'exécution (ici : pour l'utilisateur seulement)

```
| $ chmod u+x bonjour.sh
```

- Exécution de la procédure requiert **obligatoirement** le chemin d'accès

```
| $ ./bonjour.sh
```

Le fichier `bonjour.sh` est livré avec cette formation.

1. Faire du répertoire contenant le matériel de la formation le répertoire courant.
2. Afficher le contenu du fichier `bonjour.sh`.

```
| $ cat bonjour.sh
```

3. Vérifier les permissions du fichier avec la commande `ls -l`.

```
| $ ls -l bonjour.sh
```

4. Si nécessaire, ajouter le droit d'exécution au fichier et répéter l'étape précédente.

```
| $ chmod u+x bonjour.sh
```

5. Exécuter la procédure `bonjour.sh`.

```
| $ ./bonjour.sh
```


Ce document a été produit par le système de mise en page \LaTeX avec la classe **beamer** et le thème Metropolis. Les titres et le texte sont composés en Fira Sans et le code informatique en Fira Mono. Les icônes proviennent de la police Font Awesome.