

# Introduction à l'invite de commande

---

## 1. Introduction

L'invite de commande (aussi appelée **terminal**, **console**, ou **CLI** pour *Command Line Interface*) est un outil qui permet de communiquer avec l'ordinateur en tapant des **commandes** textuelles, plutôt que d'utiliser une interface graphique (fenêtres, icônes, menus).

Son utilisation :

- Automatiser des tâches (grâce à des scripts).
- **Indispensable** dans de nombreux domaines : programmation, développement web, DevOps, cybersécurité, intelligence artificielle, etc.
- Certains outils **ne fonctionnent que via la ligne de commande** (Git, Docker, etc.).
- **Plus d'options** que les interfaces graphiques.
- **Universelle** : ce que vous apprenez ici fonctionne sur tous les systèmes (Windows, Linux, macOS) avec quelques différences.

---

Exemple simple :

Quand on utilise une interface graphique pour aller dans un dossier :

- On ouvre l'explorateur
- On clique plusieurs fois...

Avec la ligne de commande :

```
cd Documents/Projets/Exemple
```

→ Une seule ligne suffit.

---

## 2. Premiers pas dans l'invite de commande

- Ouvrir l'invite de commande :
  - Windows : `cmd`, `PowerShell`, ou `Windows Terminal`
  - macOS / Linux : `Terminal`

### Structure d'une commande

Chaque commande suit en général cette **structure** :

```
commande [options] [arguments]
```

- **commande** : Le nom du programme ou outil (ex : `ls`, `cd`, `mkdir`)
- **option(s)** : Paramètres précédés d'un `$- $` ou `--` pour modifier le comportement
- **argument(s)** : Ce sur quoi la commande agit (ex : un nom de fichier)

Exemple :

```
ls -l /home/user/Documents
```

- `ls` = liste les fichiers
- `$-l` = option pour afficher plus de détails
- `/home/user/Documents` = chemin du dossier à afficher

Règles importantes :

Règle	Explication
Les commandes sont <b>sensibles à la casse</b>	<code>ls</code> $\neq$ <code>LS</code>
L'espace sépare les <b>éléments</b>	<code>ls -l</code> (pas <code>ls-l</code> )
On peut <b>combinaison des options</b>	<code>ls -la</code> = <code>ls -l -a</code>
Utiliser <b>Tab</b> pour l'autocomplétion	Évite les fautes de frappe
Utiliser les flèches $\uparrow \downarrow$	Pour naviguer dans l'historique des commandes

### 3. Navigation dans le système de fichiers

Objectif

Apprendre à se déplacer dans les dossiers du système via l'invite de commandes (`cmd`) sous Windows.

- `pwd` – Afficher le répertoire courant
- `cd` – Changer de répertoire
  - `cd ..`, `cd ~`, `cd /path/to/folder`
- `ls` ou `dir` – Lister les fichiers
  - Options : `ls -l`, `ls -a`, `ls -lh`

Action	Commande	Exemple
Afficher le dossier courant	<code>cd</code>	<code>cd</code>
Lister les fichiers d'un dossier	<code>dir</code>	<code>dir</code>
Aller dans un sous-dossier	<code>cd nom_du_dossier</code>	<code>cd Documents</code>
Remonter d'un dossier	<code>cd ..</code>	<code>cd ..</code>

Action	Commande	Exemple
Aller à la racine du disque	<code>cd \</code>	<code>cd \</code>
Aller à un chemin complet	<code>cd chemin_complet</code>	<code>cd C:\Users\Nom\Bureau</code>
Effacer l'écran	<code>cls</code>	<code>cls</code>

## Exemples pratiques

### 1. Afficher le dossier courant

```
cd
```

Affiche le chemin du dossier dans lequel vous vous trouvez.

### 2. Voir le contenu du dossier

```
dir
```

Affiche la liste des fichiers et sous-dossiers.

### 3. Aller dans un dossier

```
cd Bureau
```

Entre dans le dossier "Bureau" (s'il existe dans le dossier courant).

Utilisez **Tab** pour compléter automatiquement le nom du dossier.

### 4. Remonter d'un niveau

```
cd ..
```

Revient dans le dossier parent.

### 5. Aller directement dans un chemin complet

```
cd C:\Users\Etudiant\Documents
```

---

## Remarques importantes

- Sous **cmd**, les chemins utilisent les antislashes (\).
- Sous PowerShell, les deux \ et / fonctionnent en général.
- Si un nom de dossier contient des **espaces**, on l'entoure de guillemets :

```
cd "Mes Documents"
```

---

## Exercice

1. Ouvrir une invite de commande
2. Aller dans le dossier **Documents**
3. Créer un nouveau dossier depuis l'explorateur (**TestCLI**)
4. Utiliser **cd** pour entrer dans **TestCLI**
5. Taper **cd ..** pour revenir à **Documents**
6. Taper **cls** pour effacer l'écran

---

## 4. Gestion des fichiers

### Objectif

Apprendre à créer, supprimer, renommer, copier et déplacer des **fichiers** et **dossiers** en ligne de commande (**cmd**) sous Windows.

### Commandes de base

#### Créer un dossier

```
mkdir nom_du_dossier
```

Crée un dossier vide.

---

#### Créer un fichier vide

```
type nul > nom_du_fichier.txt
```

Crée un fichier vide.

---

## Supprimer un fichier

```
del nom_du_fichier
```

---

## Supprimer un dossier

```
rmdir nom_du_dossier
```

Pour supprimer un dossier et tout son contenu :

```
rmdir /s /q nom_du_dossier
```

---

## Renommer un fichier ou un dossier

```
ren ancien_nom nouveau_nom
```

## Exemples :

```
ren notes.txt todo.txt  
ren DossierA DossierB
```

---

## Copier un fichier

```
copy source destination
```

## Exemple :

```
copy notes.txt C:\Users\Etudiant\Bureau\
```

---

## Déplacer (ou renommer) un fichier

---

```
move source destination
```

### Exemple :

```
move notes.txt C:\Users\Etudiant\Documents\
```

---

### Astuces utiles

Astuce	Exemple
Créer plusieurs dossiers	<code>mkdir dossier1 dossier2</code>
Supprimer plusieurs fichiers	<code>del fichier1.txt fichier2.txt</code>
Utiliser *	<code>del *.txt</code> → supprime tous les fichiers <code>.txt</code>
Espaces dans les noms	<code>mkdir "Mon Dossier"</code>

---

## 5. Lecture de fichiers, redirections et pipes

### Objectif

Savoir **lire le contenu d'un fichier texte**, **rediriger l'affichage vers un fichier**, et **chaîner des commandes (pipes)**.

---

### A. Lire le contenu d'un fichier

#### type

Affiche le contenu d'un fichier texte dans le terminal.

```
type nom_du_fichier.txt
```

Attention : affiche tout d'un coup, même pour de gros fichiers.

---

### B. Redirections de sortie

#### Rediriger la sortie d'une commande dans un fichier

> : redirection (écrase le fichier si existant)

```
echo Bonjour > message.txt
```

Crée (ou écrase) `message.txt` et y écrit "Bonjour"

**>> : redirection (ajoute à la fin du fichier)**

```
echo Encore une ligne >> message.txt
```

Ajoute la ligne sans effacer ce qui existe

---

## C. Utiliser les "pipes" (|)

Le pipe permet de **chaîner deux commandes** : la sortie de la première devient l'entrée de la seconde.

```
commande1 | commande2
```

Exemple :

```
type message.txt | more
```

Affiche le contenu page par page (utile pour les longs fichiers)

Exemple avec `find` (recherche)

```
type message.txt | find "Bonjour"
```

Affiche uniquement les lignes contenant le mot "Bonjour"

---

### Résumé

Action	Commande
Lire un fichier	<code>type fichier.txt</code>
Écrire dans un fichier	<code>echo texte &gt; fichier.txt</code>
Ajouter à un fichier	<code>echo texte &gt;&gt; fichier.txt</code>
Lire un fichier par pages	<code>type fichier.txt \   more</code>
Chercher un mot dans un fichier	<code>type fichier.txt \   find "mot"</code>
Trier le contenu	<code>sort &lt; fichier.txt</code>

---

## 6. Notions utiles pour bien utiliser l’invite de commande

### A. Chemins absolus vs relatifs

Comprendre les chemins est **essentiel** pour naviguer et manipuler des fichiers correctement.

#### Chemin absolu

Chemin **complet** depuis la racine du disque.

```
cd C:\Users\Etudiant\Documents\Cours
```

Le chemin commence par une **lettre de disque** (ex. **C:**).

---

#### Chemin relatif

Chemin **par rapport au dossier actuel**.

```
cd Cours\TP1
```

Fonctionne si vous êtes déjà dans **Documents**.

---

#### Commande utile pour s’y retrouver :

```
cd
```

Affiche le **chemin courant**.

---

### B. Les dossiers spéciaux

Dossier	Raccourci en <b>cmd</b>
Dossier parent	<b>..</b>
Dossier courant	<b>.</b>
Racine du disque courant	<b>\</b>
Bureau (chemin absolu)	<b>C:\Users\NomUtilisateur\Desktop</b>

Exemple :



```
cd ..
```

→ Remonte d'un dossier.

---

## C. Fichiers cachés

Sous Windows, certains fichiers sont "**cachés**" et ne s'affichent pas avec `dir` par défaut.

**Pour les voir :**

```
dir /a
```

Option	Signification
<code>/a</code>	Affiche <b>tous</b> les fichiers
<code>/ah</code>	Affiche <b>seulement</b> les fichiers cachés

Les fichiers cachés ont l'attribut **H** dans la colonne d'attributs.

---

## D. Attributs et permissions

Sous Windows, il existe des **attributs de fichiers** (lecture seule, caché, etc.) qu'on peut modifier avec la commande `attrib`.

**Exemple :**

```
attrib +h fichier.txt
```

Rend le fichier **caché**.

```
attrib -h fichier.txt
```

Rend le fichier **visible**.

---

## Exercice

1. Naviguer jusqu'à le dossier `Documents`
2. Créer un fichier nommé `secret.txt`
3. Le rendre **caché** avec `attrib +h secret.txt`
4. Utiliser `dir` puis `dir /a` pour voir la différence

5. Le réafficher avec `attrib -h secret.txt`

---

## 7. Exercices pratiques

---

### Objectif

Mettre en pratique tout ce qui a été vu : navigation, création, manipulation de fichiers, redirections, pipes et affichage.

---

### Exercice 1 – Création et navigation

1. Ouvrir l'invite de commande.
  2. Aller sur le Bureau.
  3. Créer un dossier nommé `TP_CLI`.
  4. Entrer dans ce dossier.
  5. Créer un sous-dossier `Notes`.
- 

#### Correction :

```
cd Desktop
mkdir TP_CLI
cd TP_CLI
mkdir Notes
```

---

### Exercice 2 – Fichiers simples

1. Créer un fichier vide nommé `todo.txt` dans `TP_CLI`.
  2. Écrire la ligne "Réviser la ligne de commande" dedans.
  3. Ajouter ensuite une deuxième ligne : "Faire les exercices".
  4. Afficher le contenu du fichier.
- 

#### Correction :

```
type nul > todo.txt
echo Réviser la ligne de commande > todo.txt
echo Faire les exercices >> todo.txt
type todo.txt
```

---

### Exercice 3 – Redirections et affichage

1. Créer un fichier nommé `liste.txt` avec 5 lignes de texte (utilise `echo`).

2. Afficher uniquement les lignes contenant le mot "ligne".
3. Afficher le fichier page par page.

---

**Correction :**

```
echo ligne 1 > liste.txt
echo ligne 2 >> liste.txt
echo autre contenu >> liste.txt
echo ligne 4 >> liste.txt
echo encore autre chose >> liste.txt

type liste.txt | find "ligne"
type liste.txt | more
```

---

## Exercice 4 – Copie, déplacement et suppression

1. Copier `todo.txt` dans le dossier `Notes`.
2. Renommer le fichier `todo.txt` en `taches.txt`.
3. Supprimer le fichier `liste.txt`.
4. Supprimer le dossier `Notes` (même s'il contient un fichier).

---

**Correction :**

```
copy todo.txt Notes\
ren todo.txt taches.txt
del liste.txt
rmdir /s /q Notes
```

---

## Exercice 5 – Fichier caché

1. Créer un fichier `secret.txt`.
2. Le rendre invisible avec la commande `attrib`.
3. Vérifier qu'il n'est plus visible avec `dir`.
4. Le réafficher avec `attrib -h`.

---

**Correction :**

```
type nul > secret.txt
attrib +h secret.txt
dir
dir /a
attrib -h secret.txt
```

---

## TP synthèse

**Objectif** : Réaliser toute une séquence en autonomie, du début à la fin.

---

1. Sur le Bureau, créer un dossier **Projet\_CLI**.
  2. Dans ce dossier, créer deux sous-dossiers : **Docs** et **Sources**.
  3. Créer un fichier **plan.txt** dans **Docs** contenant 3 lignes décrivant un projet fictif.
  4. Copier ce fichier dans **Sources**.
  5. Renommer **plan.txt** en **plan\_final.txt** dans **Sources**.
  6. Supprimer le fichier d'origine dans **Docs**.
  7. Afficher le contenu de **plan\_final.txt** page par page.
  8. Rendre ce fichier caché.
  9. Vérifier sa disparition puis sa réapparition.
- 

### Correction :

```
cd Desktop
mkdir Projet_CLI
cd Projet_CLI
mkdir Docs Sources

echo Étape 1 > Docs\plan.txt
echo Étape 2 >> Docs\plan.txt
echo Étape 3 >> Docs\plan.txt

copy Docs\plan.txt Sources\
ren Sources\plan.txt plan_final.txt
del Docs\plan.txt

type Sources\plan_final.txt | more
attrib +h Sources\plan_final.txt
dir Sources
dir Sources /a
attrib -h Sources\plan_final.txt
```