# [Apprenez à utiliser la ligne de commande dans un terminal](https://openclassrooms.com/fr/courses/6173491-apprenez-a-utiliser-la-ligne-de-commande-dans-un-terminal)

Table des matières

[Apprenez à utiliser la ligne de commande dans un terminal 1](#_Toc57710208)

[**Découvrez le terminal** 3](#_Toc57710209)

[**Qu’est-ce qu’un terminal ?** 3](#_Toc57710210)

[**Comment tirer au mieux parti du cours ?** 4](#_Toc57710211)

[**De quoi aurez-vous besoin pour commencer ?** 4](#_Toc57710212)

[**En résumé** 5](#_Toc57710213)

[**Utilisez votre première commande** 6](#_Toc57710214)

[**Découvrez la ligne de commande** 6](#_Toc57710215)

[**Découvrez où vous êtes sur votre ordinateur : la commande « *pwd* »** 6](#_Toc57710216)

[**Comprenez les arguments de la ligne de commande** 7](#_Toc57710217)

[**En résumé** 8](#_Toc57710218)

[Naviguez dans le système 9](#_Toc57710219)

[Déplacez-vous dans l’arborescence de fichiers 9](#_Toc57710220)

[Explorez le contenu d’un dossier avec « *ls* » 10](#_Toc57710221)

[Utilisez différentes options de la commande ls 11](#_Toc57710222)

[Déplacez-vous dans votre répertoire avec la commande « *cd* » 14](#_Toc57710223)

[Découvrez quelques raccourcis utiles 15](#_Toc57710224)

[En résumé 15](#_Toc57710225)

[Testez vos connaissances sur le terminal 16](#_Toc57710226)

[Compétences évaluées 16](#_Toc57710227)

[Description 16](#_Toc57710228)

[ Question 1 16](#_Toc57710229)

[ Question 2 16](#_Toc57710230)

[ Question 3 17](#_Toc57710231)

[ Question 4 17](#_Toc57710232)

[ Question 5 18](#_Toc57710233)

[ Question 6 18](#_Toc57710234)

[ Question 7 19](#_Toc57710235)

[ Question 8 19](#_Toc57710236)

[ Question 9 20](#_Toc57710237)

[ Question 10 20](#_Toc57710238)

[Créez vos fichiers et vos dossiers 22](#_Toc57710239)

[Utilisez mkdir 22](#_Toc57710240)

[Que faire si le nom possède un espace ? 22](#_Toc57710241)

[Apprenez à utiliser touch 24](#_Toc57710242)

[En résumé 25](#_Toc57710243)

[Manipulez des éléments sur votre ordinateur 26](#_Toc57710244)

[Déplacez du contenu avec mv 26](#_Toc57710245)

[Découvrez les wildcards 27](#_Toc57710246)

[Copiez du contenu avec cp 28](#_Toc57710247)

[Supprimez des fichiers et des répertoires avec rm 30](#_Toc57710248)

[Allez plus loin avec de nouvelles commandes 32](#_Toc57710249)

[Affichez le manuel d’une commande avec man 32](#_Toc57710250)

[Utilisez cat, less et more pour connaître le contenu d’un fichier 33](#_Toc57710251)

[Découvrez les redirections 33](#_Toc57710252)

[Appréhendez grep 34](#_Toc57710253)

[En résumé 35](#_Toc57710254)

[Faites le bilan 37](#_Toc57710255)

[Testez-vous en situation réelle ! 38](#_Toc57710256)

[Compétences évaluées 38](#_Toc57710257)

[Description 38](#_Toc57710258)

[ Question 1 38](#_Toc57710259)

[ Question 2 38](#_Toc57710260)

[ Question 3 39](#_Toc57710261)

[ Question 4 39](#_Toc57710262)

[ Question 5 40](#_Toc57710263)

[ Question 6 40](#_Toc57710264)

[ Question 7 41](#_Toc57710265)

[ Question 8 41](#_Toc57710266)

[ Question 9 42](#_Toc57710267)

[ Question 10 42](#_Toc57710268)

Bonjour et bienvenue dans ce cours sur le terminal.

Je m’appelle David, je suis ingénieur en informatique, j’ai une quinzaine d’années d’expérience dans le développement et le web, et j’utilise la ligne de commande au quotidien.

Le terminal est une des applications les plus importantes de votre ordinateur.

Que vous vouliez devenir programmeur, travailler dans une équipe en tant que designer ou même en tant que chef de projet, il est extrêmement probable que vous soyez amené à utiliser un terminal.

En effet, le terminal vous permet d’agir sur les différents dossiers et fichiers de votre ordinateur de façon simple et rapide. Vous serez donc amené à l’utiliser régulièrement lors de votre carrière.

Dans ce cours, nous allons explorer quelques commandes, simples mais fondamentales, qui vous permettront de bien comprendre le fonctionnement du terminal.

Dans un premier temps, vous découvrirez ce qu'est un terminal et apprendrez à écrire vos première lignes de commandes.

Ensuite, vous irez plus loin et manipulerez des fichiers grâce aux lignes de commande ! 😊

À la fin de ce cours, vous serez capable de :

* découvrir les bases du terminal ;
* manipuler des fichiers avec un terminal.

Prérequis :

Aucun 😎

Inscrivez-vous dès maintenant pour en savoir plus !

**Découvrez le terminal**

Vous vous êtes lancé dans ce cours sur le terminal. Bienvenue ! Je vais vous accompagner pas à pas à la découverte de cette application dont vous ne pourrez plus vous passer.

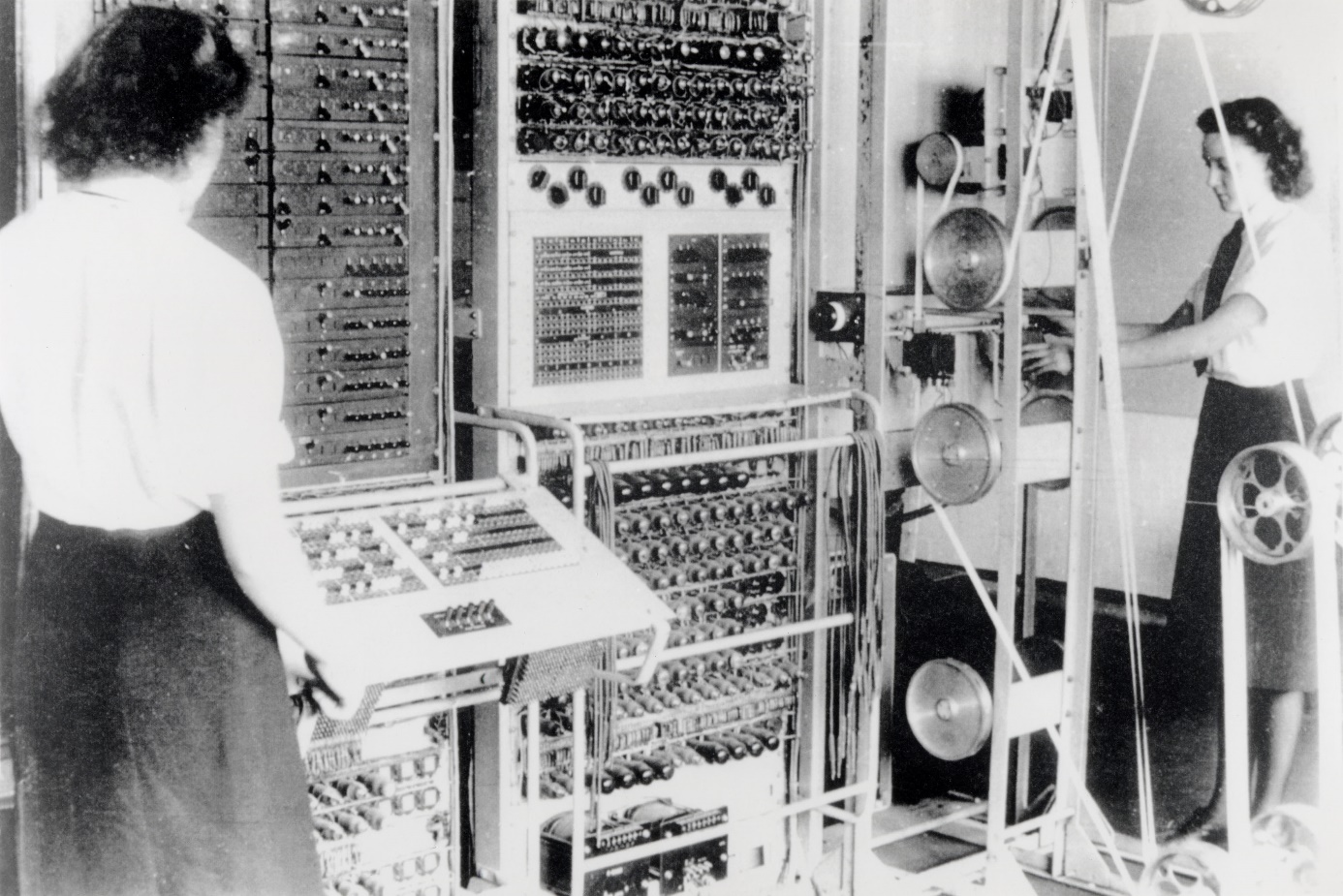
**Qu’est-ce qu’un terminal ?**

Commençons par découvrir ce qu’est un **terminal**.

Un terminal est une **application** permettant de **dialoguer** avec son ordinateur via l’écriture de **lignes de commande**.

Une ligne de commande est une instruction écrite, qui une fois envoyée à l’ordinateur via le terminal, va effectuer une action, comme par exemple créer un répertoire, lancer un programme ou afficher le contenu d’un fichier.

Historiquement, lorsque les ordinateurs faisaient la taille d’un immeuble, un terminal était un **point d’accès physique** avec un écran et un clavier, pour envoyer des instructions à la machine et recevoir ses réponses.



Les débuts de l'informatique

De nos jours, il s’agit généralement d’un programme, qui a gardé le côté un peu austère de ses premières heures, mais qui a surtout le même usage : dialoguer avec votre ordinateur.

D'une certaine façon, le terminal est l'**ancêtre** de programmes comme l'**explorateur Windows** ou **Finder** (sous Mac).

Mais que permet un terminal ?

Eh bien, entre autres choses, le terminal sert à :

* naviguer dans une arborescence de fichiers ;
* créer, supprimer et déplacer des dossiers et des fichiers ;
* lancer des programmes.

« Console », « terminal », « ligne de commande », « term », tous ces mots sont des synonymes pour désigner la même chose. Dans la suite du cours, l'un ou l'autre de ces termes seront utilisés indifféremment.

**Comment tirer au mieux parti du cours ?**

Dans ce cours, vous découvrirez des commandes simples mais essentielles pour **naviguer** dans vos dossiers et fichiers, **rechercher** du contenu, et même **créer** de nouveaux fichiers.

La **première partie** se concentrera sur l’**exploration** : qu’est-ce qu’une ligne de commande ? Comment lister le contenu d’un répertoire ? Comment se déplacer ?

La **seconde partie**, elle, vous permettra véritablement d’**agir** sur votre ordinateur, en créant des dossiers et des fichiers. Vous apprendrez aussi à les déplacer, à les copier et même à les supprimer. Cette seconde partie s’achèvera sur la découverte de quelques autres possibilités du terminal, afin de vous montrer toute la puissance de cet outil.

Une**vidéo d'introduction** sera présente**au début de chaque chapitre** pour expliquer les concepts généraux. Ces concepts seront détaillés davantage dans le texte sous la vidéo ainsi que dans une **seconde vidéo** dans laquelle vous verrez **mon écran**. À la fin de chaque partie, des quiz seront présents pour vous aider à tester vos connaissances.

En terminant ce cours, vous serez capable d'utiliser les commandes de base du terminal, et prêt à aller plus loin si vous le désirez.

**De quoi aurez-vous besoin pour commencer ?**

Si vous êtes sous **Linux**, vous pouvez directement ouvrir un terminal en suivant les instructions suivantes :

Cliquez su*r Applications > Accessoires > Terminal*.

Vous pouvez également utiliser le raccourci : *Ctrl + Alt + T*

Si vous êtes sous **Mac**, vous pouvez accéder à un terminal via :

*Applications > Utilitaires > Terminal.app*

Si vous êtes sous **Windows**, accéder à un terminal sera un peu plus complexe, car celui-ci n’est pas disponible de base et doit être installé préalablement au cours.

Pour ceci, je vous propose deux options :

* "**Cygwin**", que vous pouvez télécharger et installer en suivant les instructions du [site officiel (en anglais)](https://www.cygwin.com/install.html) ;
* **le shell Bash pour Windows,** que vous pouvez installer en suivant [les instructions officielles de Microsoft (en français)](https://docs.microsoft.com/fr-fr/windows/wsl/install-win10).

Windows possède deux utilitaires directement inspirés du terminal, mais qui **ne sont pas des terminaux**même si visuellement, ils sont très proches :

* L'invite de commande (appelée aussi fenêtre de commande DOS).
* Le powershell (présent sur les Windows les plus récents et qui accepte des commandes plus proches de ce que propose le terminal)

Ces deux programmes ont le même but que le terminal et la même philosophie, mais les commandes acceptées par le terminal ne fonctionneront pas nécessairement avec ces deux programmes, ce pourquoi il est **vivement conseillé** pour ce cours d'installer un vrai terminal avec **Cygwin** ou le **shell Bash pour Windows**.

Alors, avez-vous réussi à lancer votre terminal ? 🤓

**En résumé**

* Un terminal est une **application** permettant de **dialoguer** avec son ordinateur via l’écriture de **lignes de commande**.
* Il sert à naviguer dans une arborescence de fichiers, créer, supprimer et déplacer des dossiers et des fichiers et lancer des programmes.
* Vous avez pu lancer votre terminal ! Bravo !

*Vous avez envie de lancer votre première ligne de commande ? Rendez-vous dans le prochain chapitre !*

**Utilisez votre première commande**

Bravo ! Votre terminal doit maintenant être correctement installé ! Il est désormais temps de lancer votre première commande et de comprendre les principes derrière la ligne de commande.

**Découvrez la ligne de commande**

Avant toute chose, commencez par lancer votre terminal. Si vous ne savez pas comment faire, il vous suffira de retourner à la fin du premier chapitre et de suivre les instructions.

Il est très probable que l'affichage de votre terminal ne soit pas tout à fait identique à celui des vidéos et des captures d'écran : c’est tout à fait normal. L’affichage dépend de plusieurs paramètres, comme votre système d'exploitation ou diverses options. Par exemple, par défaut, certains n’indiquent ni le nom de l’utilisateur, ni le nom de la machine, simplement un $ et rien d’autre.  Cela ne change pas son fonctionnement.

N'ayez pas peur et lancez-vous !

Mon terminal

Voici une capture d’écran de mon terminal. J’utilise Cygwin.

Dans cet exemple, “David” est mon nom d’utilisateur. “Scalde” est le nom de ma machine (mon ordinateur).

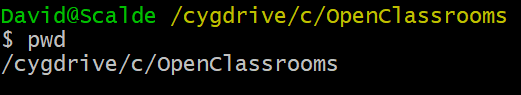
Le “~” signifie que je n’ai pas encore changé de répertoire : je suis dans le répertoire de l’utilisateur “David”, à savoir /home/David. Je peux taper mes instructions derrière le $.

**Découvrez où vous êtes sur votre ordinateur : la commande « *pwd* »**

Votre terminal attend maintenant vos instructions. Elles sont transmises par le biais de **commandes**. La plupart des commandes dans un terminal sont des **raccourcis** ou des **acronymes**.

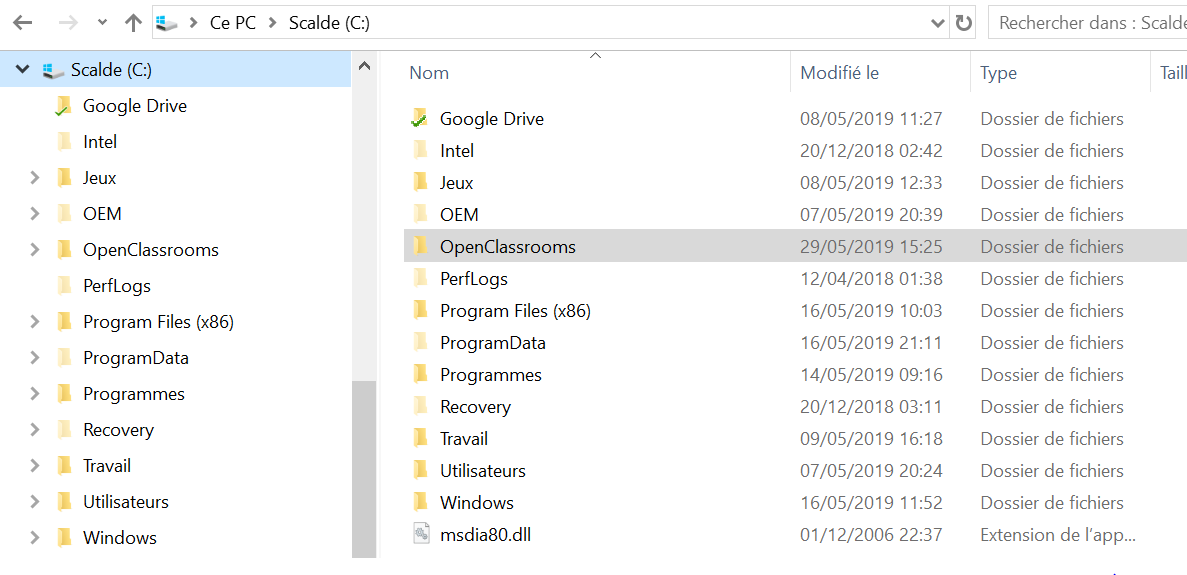
Commençons avec la commande « ***pwd***», qui signifie en réalité « **Print Working Directory** » (ou, en français, « afficher le répertoire de travail »). Avouez que « *pwd* » est tout de même plus rapide à écrire ! 🙂

Dans votre terminal, tapez la commande « *pwd* », puis appuyez sur « Entrée ».



Résultat de la commande pwd

La commande pwd demande au terminal d’indiquer **l’endroit où vous vous trouvez** dans l’**arborescence de fichiers**. Dans mon cas, , « */cygdrive/c/OpenClassrooms* » indique que je me trouve dans le dossier « OpenClassrooms», situé dans à la racine de mon disque C.



L'explorateur Windows

Mais pourquoi le terminal indique “cygdrive" ? 🧐

Je travaille actuellement sous Windows, donc pour avoir accès à un terminal, j’ai installé **Cygwin**. Avec Cygwin, si je veux accéder à mon disque C, je dois aller dans un dossier particulier. Ce dossier spécial s’appelle “**cygdrive**”. Donc lorsque mon terminal affiche “/cygdrive/c/OpenClassrooms”, cela signifie tout simplement que je suis dans le dossier “OpenClassrooms” du disque “C”.

Sous Linux et Mac, ce dossier “cygdrive” n’est pas présent, car ces systèmes d’exploitations possèdent déjà un terminal.

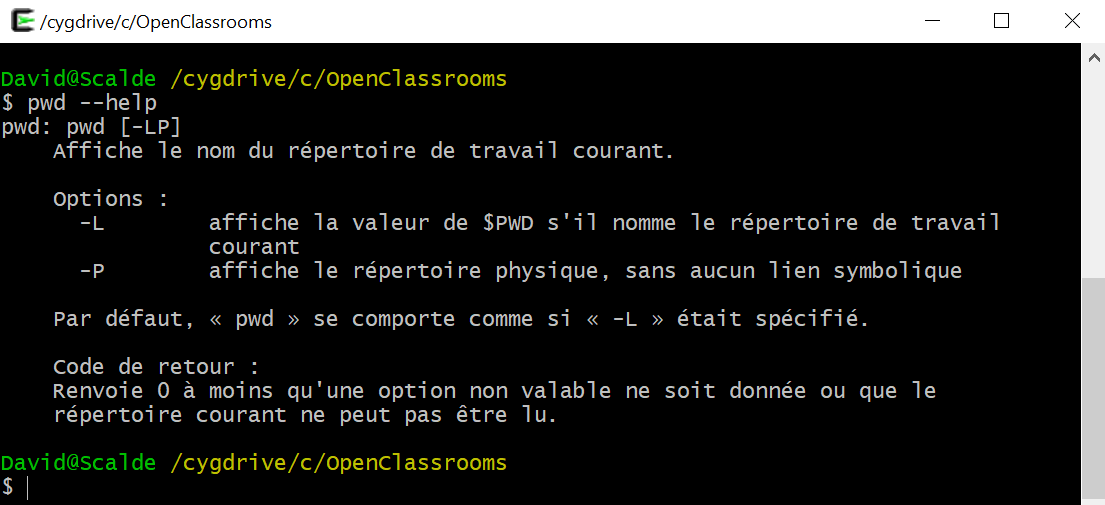
Chez vous, la réponse à pwd sera probablement légèrement différente par rapport à mon terminal, ce qui est tout à fait normal.

**Félicitations**, vous venez de lancer votre **première commande** dans un terminal !

**Comprenez les arguments de la ligne de commande**

La plupart des instructions de la ligne de commande prennent en compte des “paramètres” (ou “arguments”) qui vont venir “compléter” l’instruction et modifier son comportement.

Par exemple, avec pwd. Au lieu de taper simplement l’instruction “pwd", vous allez rajouter le paramètre “--help”. Cette commande avec l’argument --help n’affiche plus le répertoire dans lequel vous vous trouvez, mais l’aide associée à la commande.



pwd –help

 L’argument *--help*a bien modifié ce que faisait la commande.  😎😎

*Avez-vous réussi à afficher l'aide de la commande pwd sur votre machine ?*

**En résumé**

* Vous avez lancé votre terminal et découvert à quoi il ressemble.
* Vous avez lancé votre première commande : pwd.
* Vous avez ajouté un paramètre à la commande pwd : --help.

*Ces notions vous seront particulièrement utiles dans les chapitres à venir, et notamment le suivant qui va vous apprendre à****naviguer dans votre système****via le terminal !*

## Naviguez dans le système

Maintenant que vous avez lancé votre première ligne de commande, voyons comment **s’organisent les fichiers** sur votre ordinateur et comment vous pouvez **naviguer** dans cette **arborescence** grâce au terminal.

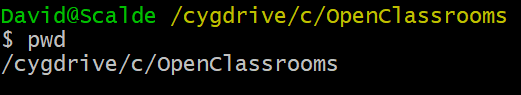
### 

### ****Déplacez-vous dans l’arborescence de fichiers****

Les fichiers sur votre ordinateur sont rangés dans des dossiers qui peuvent eux-mêmes contenir des sous-dossiers, des fichiers, etc.

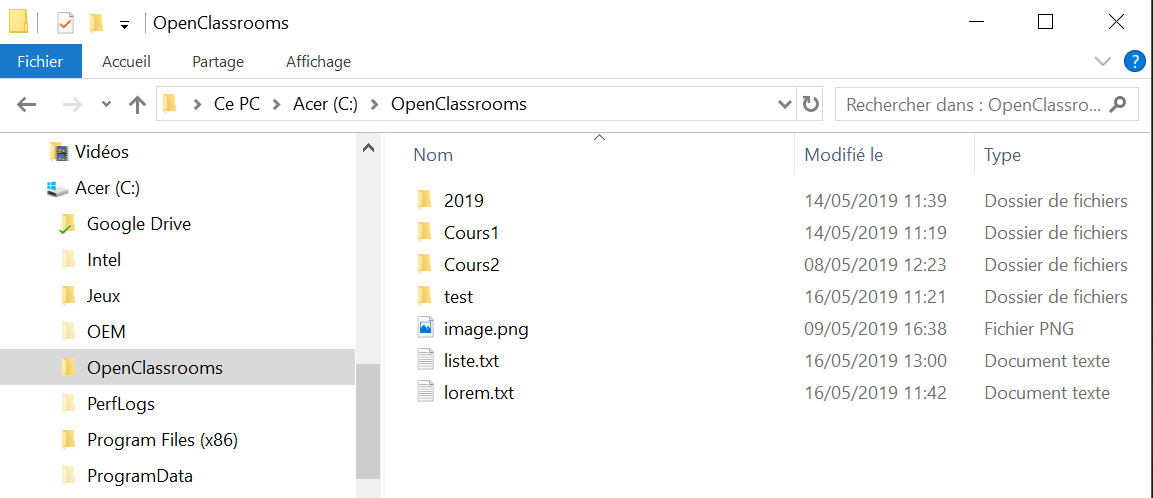
Cette organisation est appelée “l’arborescence de fichiers”. C’est ainsi que l’intégralité de vos documents sont “rangés” dans votre machine.

Dans le chapitre précédent, lorsque j'ai tapé la commande pwd, j'ai obtenu ceci :



La commande pwd avec le terminal

Concrètement, cela correspond à ceci sur mon ordinateur :



Mon explorateur Windows, au même endroit

Le dossier “OpenClassrooms” contient **plusieurs sous-dossiers**, comme “2019”, “Cours1”, “Cours2”, “test”, ainsi que **quelques fichiers**, “image.png”, “liste.txt” et “lorem.txt”.

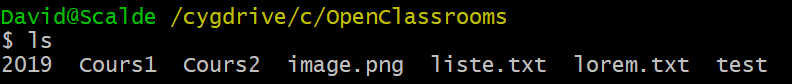
Et vous ? Qu'avez vous obtenu ? 😃

 Mais comment faire avec un terminal pour savoir ce que contient un dossier ?

### 

### ****Explorez le contenu d’un dossier avec «****ls****»****

 La commande **ls** (qui est le raccourci pour "list" en anglais) permet de **lister le contenu** d'un **répertoire**. Essayons-la.



La commande ls dans le terminal

Faites le test à partir d'un dossier sur votre ordinateur. En tapant ls, vous devriez **retrouver le contenu** de votre répertoire, avec la **liste des sous-dossiers** et des **fichiers** dans votre terminal.

Cependant, dans le terminal, vous ne voyez que les noms des éléments présents dans le dossier. Il est difficile de savoir si ce sont des dossiers ou des fichiers (les extensions comme .txt ou .png sont des indices, certes, mais il est tout à fait possible de créer des fichiers sans extensions ou des répertoires qui en possèdent une, ce n’est donc pas fiable).

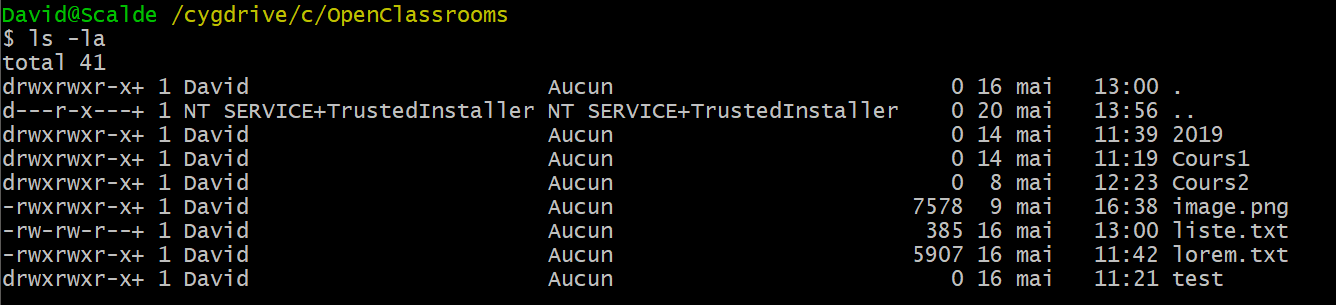
De plus, il existe des fichiers et des dossiers “**cachés**” que vous ne voyez pas avec la commande “ls” de base.

Pour remédier à ces problèmes, vous pouvez ajouter deux paramètres à la commande ls :

* -l : ce paramètre précise à la commande ls d’afficher plus d’informations sur chacun des éléments présents ;
* -a : ce paramètre indique que tous les dossiers et fichiers devront apparaître, y compris les dossiers et fichiers cachés.

Il est également possible de **combiner** ces deux commandes. Cela s’écrira ls -l -a ou plus simplement : **ls -la**

Vous pouvez constater que l'affichage est nettement plus détaillé.



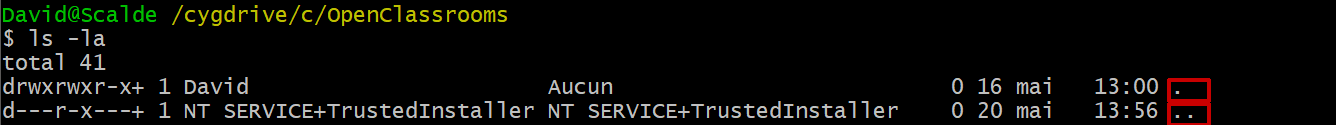
ls -la

Chaque ligne possède beaucoup d’informations. Vous pouvez par exemple repérer que chaque ligne **commence** par un “**d**” ou un “**-**”. Lorsque la ligne commence par un “d”, pour “directory”, cela signifie que l’élément correspondant est un **dossier**. Sinon, c’est un **fichier**.

Vous pouvez également voir la date de dernière modification (par exemple : 16 mai à 11 h 42 pour lorem.txt), ou encore la taille du fichier (par exemple 5 907 octets pour lorem.txt).

Alors, qu’est-ce qui apparaît chez vous ?

Vous pouvez également constater l'apparition de deux éléments supplémentaires, “**.**” et “**..**”.



Les dossiers cachés "." et ".."

Ces deux éléments sont en réalité des dossiers cachés, présents dans les répertoires, et aux propriétés un peu particulières :

* “.” : ce dossier désigne toujours le répertoire dans lequel on se trouve. Autrement dit, si “pwd” vous dit que vous êtes dans /cygdrive/c/OpenClassrooms, alors le terminal remplacera automatiquement “.” par “/cygdrive/c/OpenClassrooms” ;
* “..” : ce dossier désigne toujours le répertoire parent (c’est-à-dire le dossier qui contient le dossier dans lequel vous êtes actuellement).

Par exemple, si vous êtes dans “/cygdrive/c/OpenClassrooms”, le répertoire parent est : “/cygdrive/c”.

Si les dossiers cachés **.** et**..** ne vous semblent pas clair, **pas de panique**, nous y reviendrons plus en détail dans la suite du cours !

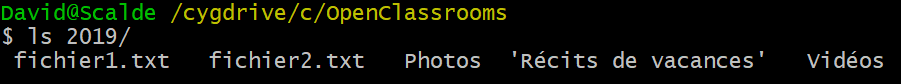
### ****Utilisez différentes options de la commande ls****

Vous allez découvrir différentes options de la commande ls.

#### **ls nomDuRepertoire**

Commençons par ls nom du répertoire. Vous savez désormais afficher le **contenu** du **répertoire courant**, mais comment faire pour afficher le contenu d’un **autre répertoire** ?  Eh bien, vous pouvez utiliser la commande **ls nomdurepertoire**.

Par exemple ici, si je tape ls 2019, le contenu du répertoire 2019 va s’afficher.



ls 2019

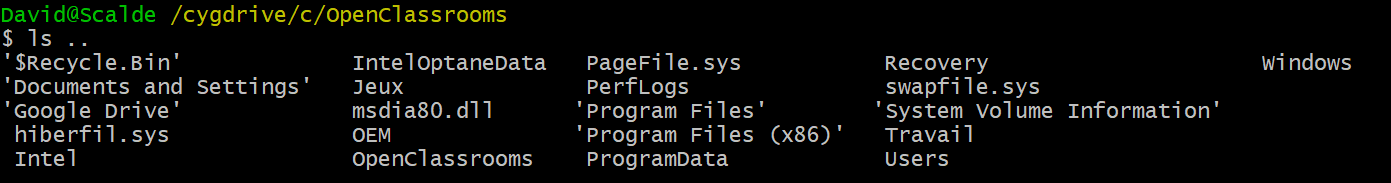
Comme vous le voyez, simplement en spécifiant le nom du répertoire, il est possible de consulter son contenu !

#### **ls ..**

Découvrons maintenant ls ..

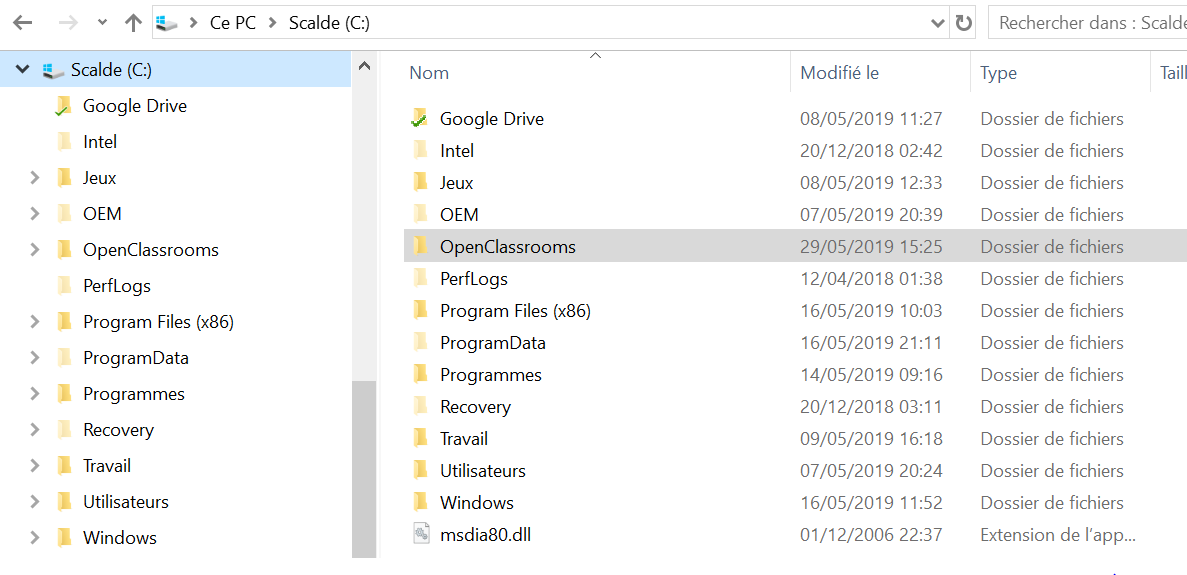
‘..’ quant à lui est un **dossier caché** qui correspond toujours au **répertoire parent**. Donc ls .. va afficher le contenu du répertoire parent.

Dans notre exemple, nous sommes dans le dossier OpenClassrooms du disque C. Le dossier dans lequel se trouve "OpenClassrooms” est directement le disque C. Donc ls .. va afficher le contenu de mon disque C.



ls ..

Pour mieux comprendre, voici l’équivalent depuis mon explorateur Windows :

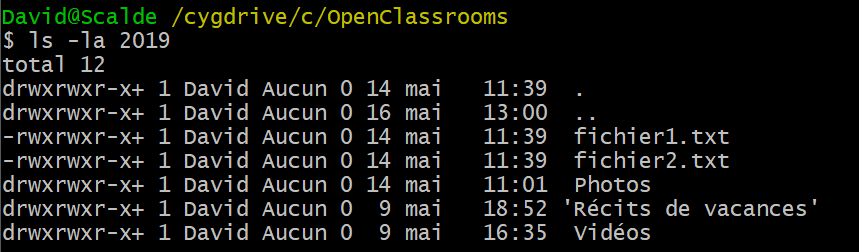


Le contenu du répertoire ciblé par « ls .. » lancé depuis le dossier OpenClassrooms

#### **ls - la répertoire**

Et bien sûr, tous les paramètres peuvent être combinés. 😱

**ls -la 2019** va afficher le contenu du **répertoire “2019”**, avec les **dossiers et fichiers cachés** (option “a”) et sous forme de **liste détaillée** (option “l”).

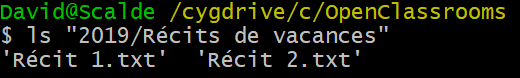


ls -la 2019

Alors, vous vous en sortez ? 😲

Si vous souhaitez lister le contenu d'un dossier dont le nom contient un espace comme « mon dossier » cela peut poser des problèmes, car la commande « ls » croira que vous cherchez à lister le contenu deux dossiers distincts : "mon" et "dossier".

Pour afficher le contenu d’un dossier dont le nom contient un espace, vous pouvez utiliser des guillemets :



ls « Récit de vacances »

### 

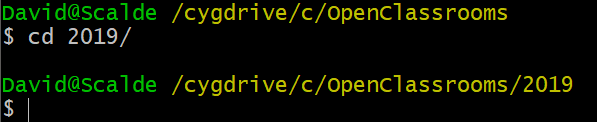
### ****Déplacez-vous dans votre répertoire avec la commande «****cd****»****

Maintenant que vous savez afficher ce qui se trouve dans un répertoire donné, voyons comment vous rendre dans ce même répertoire.

Pour cela, la commande « **cd**», pour « **Change Directory** » (« changer de répertoire », en français) vous permettra de **naviguer** d'un répertoire à l'autre.

Le fonctionnement est très simple, il suffit de taper **“cd” suivi du répertoire de destination**.

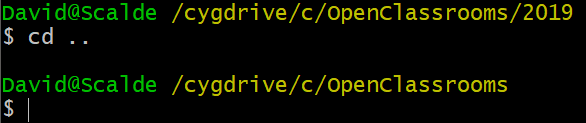
Par exemple en tapant “cd 2019”, je vais me rendre dans mon répertoire appelé “2019”.



cd 2019

Pour vérifier que la commande cd s’est bien exécutée, vous pouvez utiliser pwd.

Pour retourner dans le dossier parent, nous utilisons le .. que nous avons vu précédemment :



cd ..

Entraînez-vous à naviguer d’un dossier à l’autre ! 💪💪

### 

### ****Découvrez quelques raccourcis utiles****

Il peut parfois être un peu fastidieux de retaper à la main toutes ces commandes. Mais heureusement pour vous, il existe un certain nombre de « raccourcis » pour éviter de le faire :

* la flèche haut et la flèche bas permettent de **naviguer dans l'historique** du terminal et de relancer des commandes déjà tapées précédemment ;
* tabpermet de faire de **l'autocomplétion**, c’est-à-dire compléter automatiquement une commande ou un chemin si on a commencé à en taper le début. Si plusieurs options sont disponibles, taper **une deuxième fois** sur cette touche **affichera la liste des options** possibles ;
* ctrl + r: cette combinaison permet de faire une **recherche** dans **l'historique des commandes**. Faites d’abord ctrl+r pour passer en mode “recherche”, puis tapez une partie de la commande que vous voulez rejouer. Lorsque votre recherche a porté ses fruits, vous n’avez plus qu’à valider en appuyant sur Entrée;
* ctrl + aet ctrl + e : ces deux combinaison permettent respectivement d'**aller** automatiquement au tout **début** ou à la toute **fin** de la ligne de commande que vous êtes **en train de taper**, ce qui peut être pratique lorsque vous êtes en train d'écrire une commande particulièrement longue.

### ****En résumé****

* Vous avez découvert que les dossiers sont généralement organisés en sous-dossiers et en fichiers.
* Vous avez appris à vous **déplacer** à l’intérieur d'un dossier grâce à la commande “**cd**”.
* Vous avez utilisé différentes variantes de la commande ls.

Dans le prochain chapitre, vous irez plus loin encore et vous apprendrez à ***manipuler*** des dossiers et des fichiers !

# Testez vos connaissances sur le terminal

Bravo ! Vous avez réussi cet exercice !

### Compétences évaluées

* Découvrir les bases du terminal

### Description

Ce petit quiz vous permettra de vérifier que vous avez bien assimilé cette première partie, lancez-vous !

### Question 1

**Le terminal est ... entre vous et votre ordinateur ?**

* + 

une machine

* + 

un serveur

* + 

une interface

* + 

un bug

*Le terminal est une*interface*entre vous et votre ordinateur. Son but premier est de vous permettre de communiquer avec lui grâce à des commandes.*

### Question 2

**Pour afficher l'endroit où vous vous trouvez sur votre ordinateur, quelle est la commande à utiliser ?**

* + 

ls

* + 

pwd

* + 

mv

* + 

cd

*La commande à utiliser pour afficher l'endroit où l'on se trouve dans sa machine (appelé aussi le "répertoire de travail") est*pwd*.*

*Pwd est un sigle qui signifie "print working directory", mais la commande à taper est simplement*pwd*.*

### Question 3

**Qu'est-ce que le "répertoire de travail" ?**

* + 

L'endroit où l'on se trouve dans l'arborescence des fichiers.

* + 

Le seul endroit d'où l'on peut exécuter des commandes du terminal.

* + 

L'endroit où se trouve l'application "terminal".

* + 

Le bureau Windows.

*Le répertoire de travail correspond à l'endroit où l'on se trouve dans l'arborescence des fichiers. C'est ce répertoire qui est retourné lorsqu'on tape "pwd" dans un terminal par exemple.*

### Question 4

**À propos des paramètres ou arguments d'une commande, quelles affirmations sont vraies ?**

*Attention, plusieurs réponses sont possibles.*

* + 

Les paramètres modifient le comportement d'une commande, en lui apportant des précisions, comme des noms de dossiers ou des options.

* + 

Il est impossible de lancer une commande sans préciser au moins un argument.

* + 

Toutes les commandes peuvent interpréter exactement les mêmes arguments.

* + 

Chaque commande possède ses propres arguments.

*Les commandes peuvent prendre un nombre variable d'arguments. Certaines commandes ont un comportement par défaut qui est appliqué lorsqu'aucun argument n'est spécifié (par exemple "ls" se comporte comme si on lui avait précisé d'afficher le contenu du répertoire de travail).*

*Les arguments sont spécifiques à chaque commande et vont directement influer sur ce que va faire la commande.*

### Question 5

**Je veux afficher le contenu de mon dossier avec deux conditions :**

**- je veux pouvoir différencier leur nature : si ce sont des fichiers ou des dossiers ;**

**- je veux voir tous les fichiers et dossiers (cachés ou non).**

**Quelles commandes puis-je utiliser ?**

*Attention, plusieurs réponses sont possibles.*

* + 

ls -a

* + 

ls -l

* + 

ls -l -a

* + 

ls -la

*Pour afficher le contenu d'un dossier, la commande à utiliser est "ls" (pour "list").  
L'option -l permet d'afficher le contenu avec beaucoup plus d'options, et permet notamment de savoir, pour chaque élément, si c'est un dossier ou pas, ainsi que d'autres informations, comme sa taille ou sa date de modification.  
L'option -a permet d'afficher les fichiers et dossiers cachés. En particulier, le dossier . (qui représente le répertoire de travail) et le dossier .. (qui représente le répertoire parent), qui sont systématiquement présents.*

*La commande ls permet de combiner les arguments. De fait, écrire "ls -l -a" est strictement équivalent à écrire "ls -la".*

### Question 6

**Que signifie la commande cd ?**

* + 

Command Documentation : cette commande sert à avoir l'aide d'une autre commande.

* + 

Change Directory : cette commande sert à changer de répertoire de travail.

* + 

Change Dynamics : cette commande améliore les performances de votre ordinateur.

* + 

Command Destroy : cette commande détruit des fichiers aléatoirement sur votre ordinateur.

*La commande "cd" signifie "Change Directory". Elle sert à changer de répertoire de travail.*

*Pour avoir l'aide d'une autre commande, il faut utiliser "man" (mais nous verrons cela plus tard, dans la suite du cours !)*

### Question 7

**Si je suis dans le répertoire "/home/David/dossier/sousDossier" et que je tape "cd .." à quel endroit vais-je me retrouver ?**

* + 

Au même endroit : /home/David/dossier/sousDossier

* + 

Dans le répertoire parent : /home/David/dossier

* + 

À la racine : /

* + 

Dans mon dossier "home" : /home/David

*La commande cd sert à changer de répertoire de travail. Ici ".." est un argument qui signifie "le répertoire parent". Donc la commande "cd .." signifie : change le répertoire de travail pour aller dans le répertoire parent.*

*Comme nous sommes partis d'ici : "/home/David/dossier/sousDossier", le répertoire parent est : "/home/David/dossier/".*

### Question 8

**Suite de la question précédente : si je tape à nouveau "cd ..", à quel endroit vais-je maintenant me retrouver ?**

* + 

/home/David/dossier/sousDossier

* + 

/home/David/dossier

* + 

/

* + 

/home/David

*À la fin de la question précédente, le répertoire de travail est devenu "/home/David/dossier". Si nous retapons la commande "cd ..", alors nous demandons à nouveau au terminal d'aller dans le répertoire parent, donc la réponse est "/home/David".*

### Question 9

**Dans le terminal, que permettent les flèches vers le haut et vers le bas ?**

* + 

D'aller au début ou à la fin de la ligne.

* + 

De faire de l'autocomplétion (compléter automatiquement une commande ou un paramètre, comme un nom de répertoire).

* + 

D'afficher la liste des dossiers et fichiers du répertoire courant.

* + 

De naviguer dans l'historique des commandes déjà lancées.

*Les flèches vers le haut et vers le bas permettent de naviguer dans l'historique des commandes, très pratique pour éviter d'avoir à retaper à la main une commande que vous avez entrée précédemment.*

*Pour aller au début ou à la fin de la ligne, vous pouvez utiliser les flèches vers la gauche ou la droite, ainsi que les raccourcis ctrl+a et ctrl+e.*

### Question 10

**Quelle touche permet de faire de l'autocomplétion dans votre terminal ?**

* + 

ctrl +a

* + 

ctrl

* + 

Enter

* + 

tab

*tabpermet de faire de****l'autocomplétion****, c’est-à-dire compléter automatiquement une commande ou un chemin si on a commencé à en taper le début. Si plusieurs options sont disponibles, taper****une deuxième fois****sur cette touche****affichera la liste des options****possibles.*

## Créez vos fichiers et vos dossiers

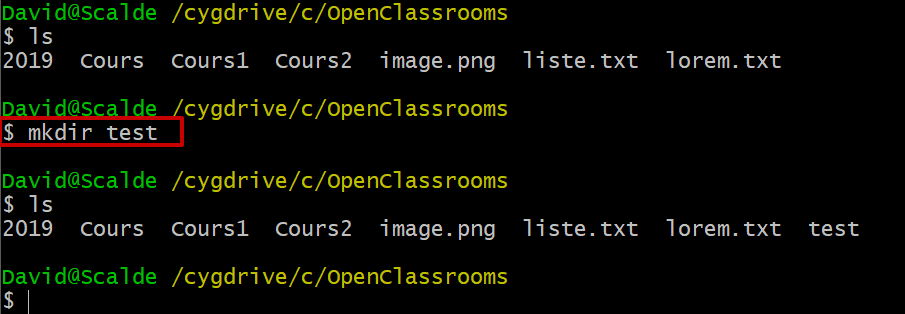
Pour le moment, vous avez navigué dans des répertoires et avez affiché leur contenu. Essayons désormais de créer de nouveaux éléments.

### 

### ****Utilisez mkdir****

**mkdir** est la contraction de “make directory”, c’est-à-dire “Créer un répertoire”, en français.

La syntaxe de mkdir est simple : il suffit de taper "mkdir" suivi du nom du répertoire que l'on désire créer. Par exemple, si je tape “mkdir test”, cela créera un répertoire appelé “test” à l’endroit où je me trouve :



mkdir test

Dans l’exemple ci-dessus, j’ai commencé par un **ls** pour voir l’ensemble des dossiers et fichiers présents, puis j’ai créé un **nouveau répertoire** appelé “test” et j’ai enfin exécuté un second “ls” pour **vérifier** que mon nouveau répertoire a bien été créé.

En tapant **cd test**, vous pouvez même vous rendre à l’intérieur de ce nouveau répertoire.

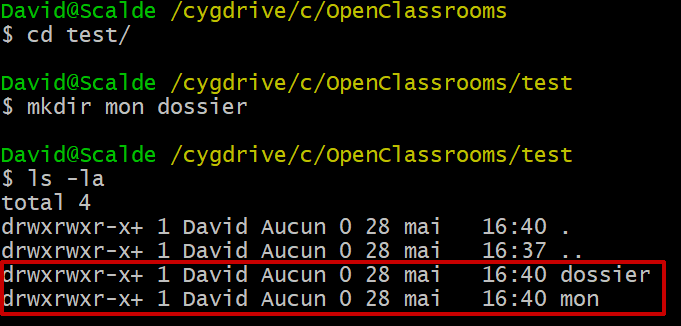
Alors, avez-vous réussi à créer un dossier ? 😀

### 

### ****Que faire si le nom possède un espace ?****

Si vous voulez créer un dossier dont le nom possède un espace, par exemple “mon dossier”, la première idée est simplement d’écrire “mkdir mon dossier”.

Essayons :



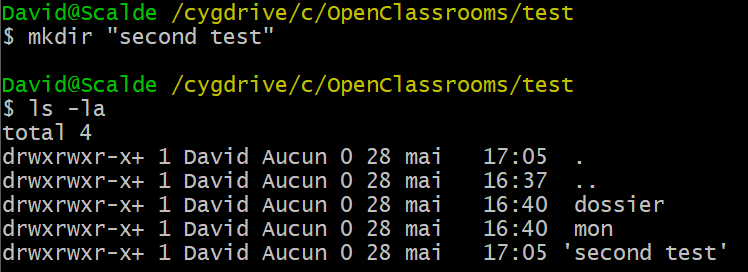
mkdir mon dossier

Vous pouvez constater que **deux** dossiers ont été créés. Le premier s'appelle “**dossier**” et le second “**mon**”.

En effet, le terminal considère l’espace comme un **séparateur** entre deux dossiers, alors que “**mon dossier**”, est en réalité le nom complet.

Il faut donc indiquer au terminal que cet espace est bien un caractère et pas un séparateur.

La façon la plus simple de faire est d’entourer le nom du dossier avec des guillemets.

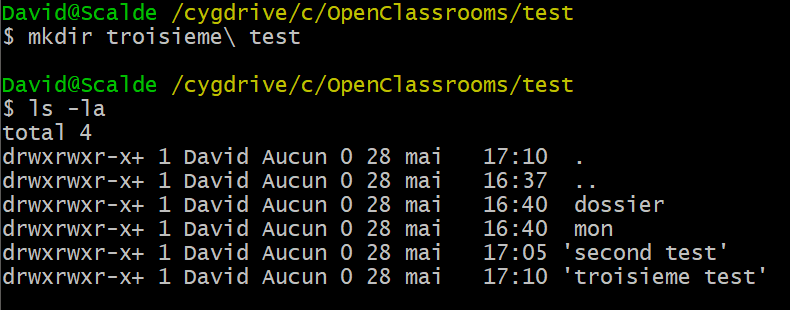


mkdir « second test »

Ici, vous pouvez constater que “second test” a bien été créé.

L’autre manière de faire est “d’**échapper**” le caractère espace, c’est-à-dire d’indiquer au terminal que l’espace en question n’est **pas un séparateur** mais doit se comporter comme une “lettre” normale. Le **caractère d'échappement** s’appelle un “**antislash**” (ou **backslash**) :**\**.

Il suffit de le mettre devant le caractère à “échapper”.



mkdir troisieme\ test

Et vous pouvez constater la présence du nouveau répertoire.

Essayez de refaire l'opération. Alors, vous trouvez cela plus facile maintenant ?

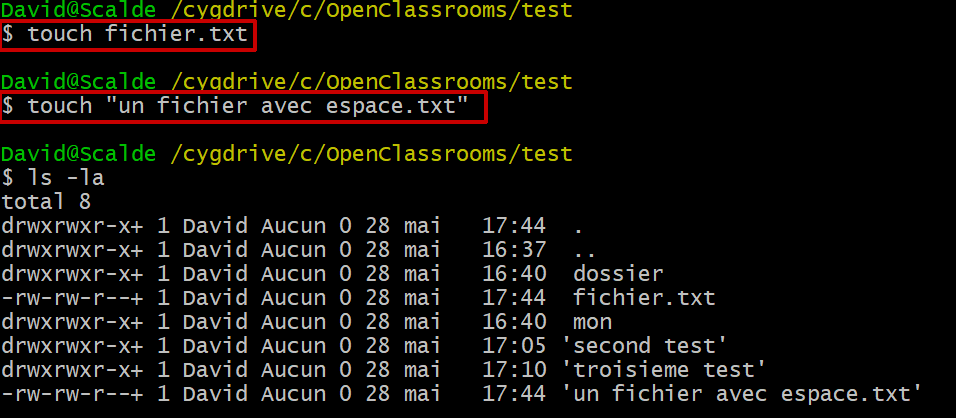
### 

### ****Apprenez à utiliser touch****

Vous savez désormais créer des dossiers, mais comment créer des fichiers ? Eh bien, exactement de la même manière que pour créer un dossier, mais avec une autre commande.

Cette fois-ci, vous allez utiliser touch, avec la même syntaxe, c'est-à-dire : touch nomDuNouveauFichier.

Par exemple, pour créer un fichier appelé fichier.txt, il suffira de taper : touch fichier.txt



touch fichier.txt et touch "un fichier avec espace.txt"

Et vous voici avec deux fichiers :

* fichier.txt ;
* un fichier avec espace.txt.

Pour créer un **nom** de fichier avec **espace**, il faudra prendre les mêmes précaution que pour la création d’un dossier, c’est-à-dire soit :

* **entourer** le nom du fichier avec des guillemets : **“nom fichier"**;
* **échapper** l’espace avec un antislash : **nom\ fichier**.

### 

### ****En résumé****

Dans ce chapitre, vous avez appris :

* à **créer** des **dossiers**avec la commande mkdir ;
* à **créer** des **fichiers** avec la commande touch ;
* à **manipuler** des **noms** de dossiers et de fichiers **comportant des espaces**.

L’étape suivante va être de manipuler des fichiers existants, en les ***déplaçant***, en les ***copiant*** et même en les ***supprimant*** !

## Manipulez des éléments sur votre ordinateur

Maintenant que vous savez vous repérer avec le terminal et même créer des dossiers et fichiers, il est temps d’apprendre à les **déplacer**, à les **copier** et à les **supprimer**.

Soyez particulièrement vigilant lorsque vous ferez vos tests : manipuler des éléments implique qu’il va être possible de les supprimer, et donc potentiellement d’effacer des fichiers que vous voudriez garder.

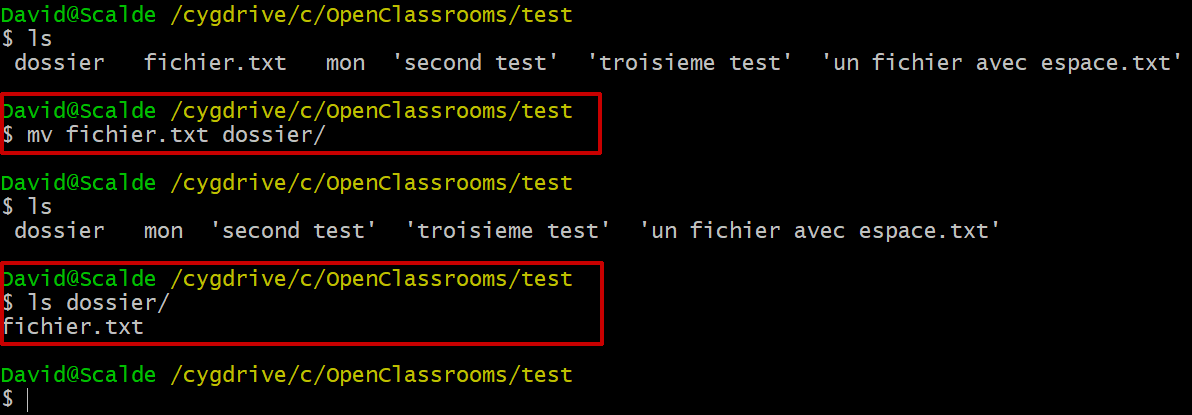
N’hésitez pas à vous créer un répertoire spécial pour faire vos tests.

### 

### ****Déplacez du contenu avec mv****

Pour déplacer un fichier, la commande à utiliser est **mv** pour “move”, déplacer, en français. La syntaxe sera donc : mv elementADeplacer destination.

Par exemple, pour déplacer le fichier “fichier.txt” à l’intérieur du répertoire “dossier”, nous écrirons :



mv fichier.txt dossier

Pensez à **vérifier** que vos commandes se sont bien passées à l’aide de **ls** !

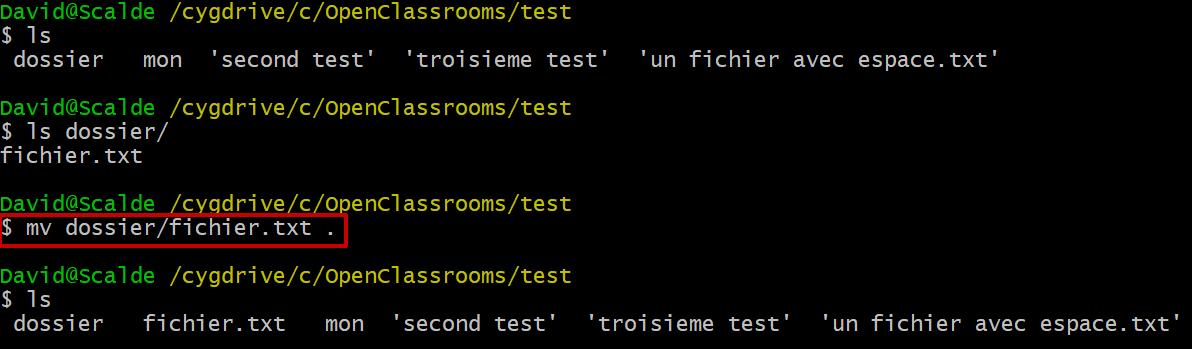
Dans l’image ci-dessus, nous pouvons voir que le fichier “fichier.txt” qui était initialement présent dans le répertoire "test" a bien disparu, mais se retrouve à l’intérieur du dossier “dossier”.

Alors, vous avez réussi à déplacer des fichiers ?

Et si je veux faire l’opération inverse et remettre le fichier “fichier.txt” là où il était ?

Souvenez-vous du **dossier caché** “**.**”, celui qui indiquait quel est le **dossier courant**. C’est ici qu’il va prendre toute son utilité.

En effet, la commande à utiliser est  mv dossier/fichier.txt .



mv dossier/fichier.txt .

“dossier/fichier.txt” sert ici à désigner le fichier “**fichier.txt**” qui se trouve**à l’intérieur** du répertoire “**dossier**”, et “**.**” signifie “**ici**”.

En d’autres termes, avec cette commande je demande au terminal de me **déplacer le fichier “fichier.txt”**.

Et comme à chaque fois, un petit **ls** pour **vérifier** que tout a fonctionné comme nous nous y attendions.

### Découvrez les wildcards

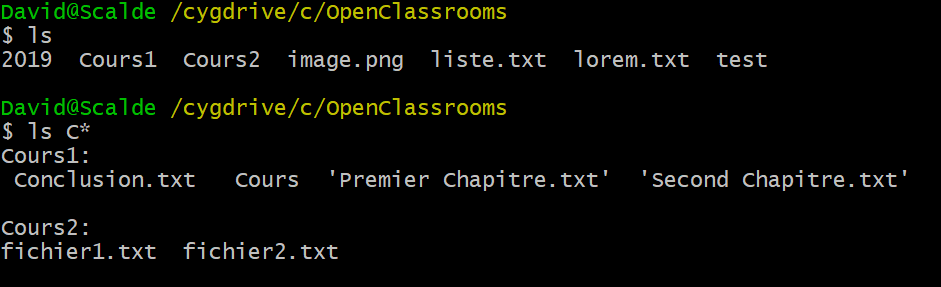
Vous savez désormais déplacer du contenu, mais comment s’y prendre si on veut déplacer **beaucoup de fichiers d’un coup** ?

Il est possible de retaper “mv” pour chaque fichier, mais si vous avez une centaine de fichiers à déplacer, vous risquez de vous ennuyer assez vite :-).

C’est pour cette raison que l’on a créé des “**wildcards**” ou “**jokers**”. Ce sont des caractères qui “**remplacent**” une partie du nom d’un fichier.

Il en existe plusieurs, mais le plus connu d’entre eux est probablement le caractère **\***. Ce caractère peut être utilisé comme substitut pour n’importe quelle autre chaîne de caractères dans une recherche.

Plutôt qu’un long discours, voici un exemple :



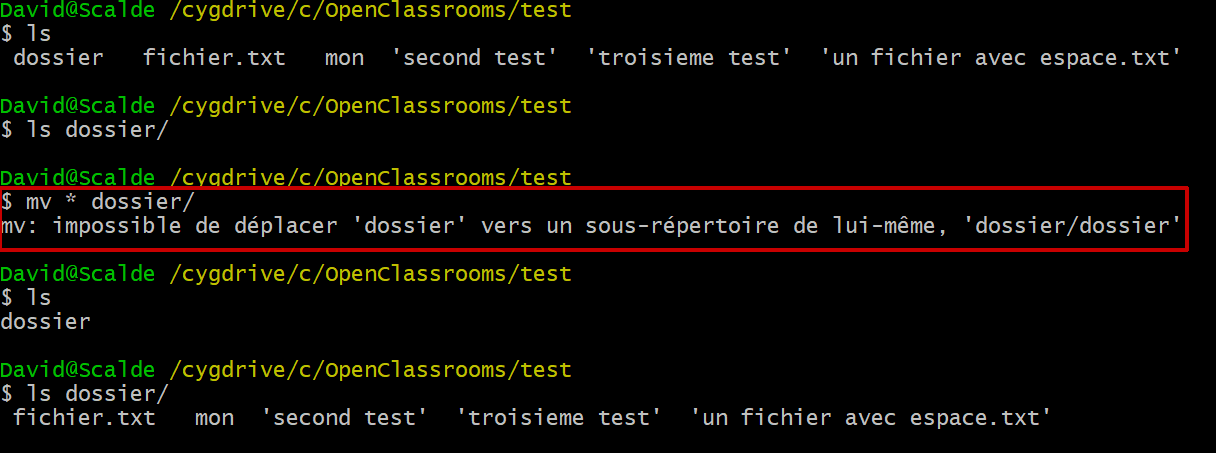
ls c\*

Avec ls, vous pouvez constater que j’ai deux dossiers qui commencent par “C” dans mon répertoire courant : **Cours1** et **Cours2**.

Si je tape **ls C\***, alors **C\*** va être remplacé, au moment de l'exécution de la commande, par tous les noms qui “correspondent” à C\*, ici Cours1 et Cours2.

Taper **ls C\*** revient donc en réalité à taper **ls Cours1 Cours2**, mais en plus rapide (et si vous avez 100 cours, en beaucoup plus rapide ! ):D

Et cela fonctionne avec toutes les commandes :



mv \* dossier/

Dans ce nouvel exemple, je suis dans mon **répertoire test**, qui contient plusieurs sous-répertoires et fichiers (dossier, fichier.txt, mon, ‘second  test’, ‘troisième test’, ‘un fichier avec espace.txt').

Parmi eux, le répertoire appelé “**dossier**” ne contient rien du tout.

Lorsque je fais **mv \* dossier**, je demande au terminal de déplacer (mv) tout ce qui correspond à \* (c’est-à-dire, absolument tout !) dans le répertoire “dossier”.

Notez le message d’erreur qui apparaît, car il est impossible de déplacer un répertoire à l’intérieur de lui-même. 😃

Cependant, tous les autres éléments ont bien été copiés à l’intérieur du répertoire “dossier”. Si vous tapez la commande ls, vous pouvez vérifier qu’il n’y a plus que le répertoire “dossier” dans le répertoire courant, et que les autres éléments ont bien été déplacés.

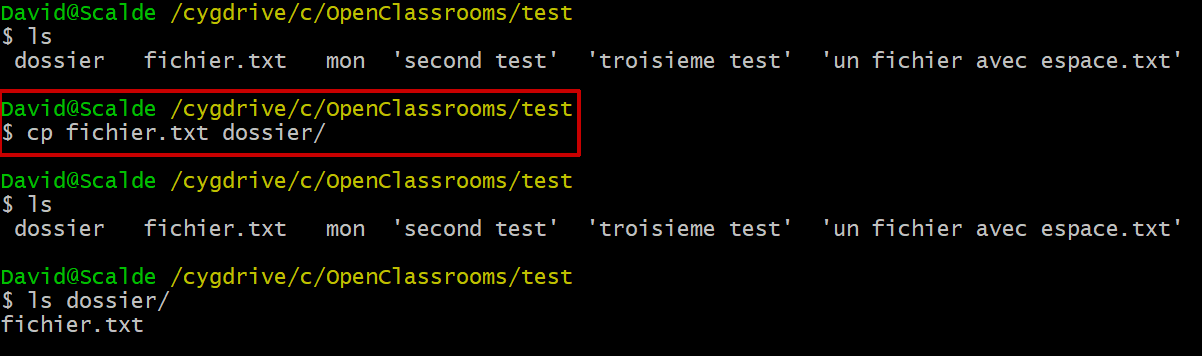
### 

### ****Copiez du contenu avec cp****

Pour copier du contenu, nous allons procéder exactement de la **même manière** que ce que nous avons fait pour la commande mv, mais avec la commande “**cp**”.

La syntaxe pour copier un élément est cp elementACopier destination.

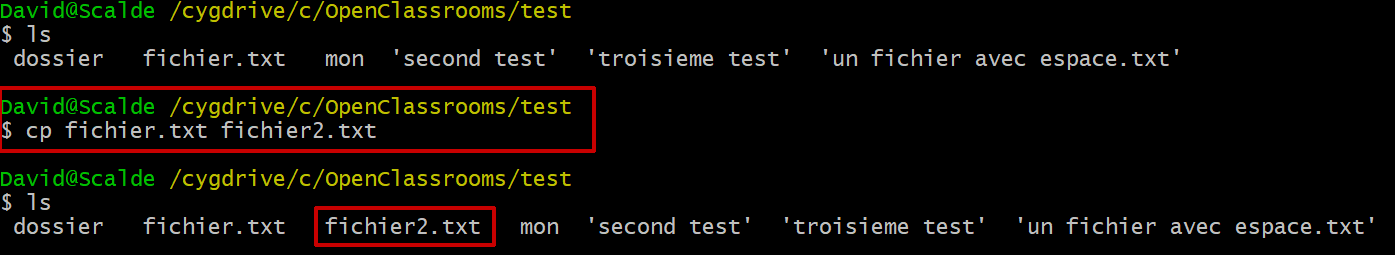
Par exemple pour **copier le fichier** “fichier.txt” **dans le répertoire** “dossier” :



cp fichier.txt dossier/

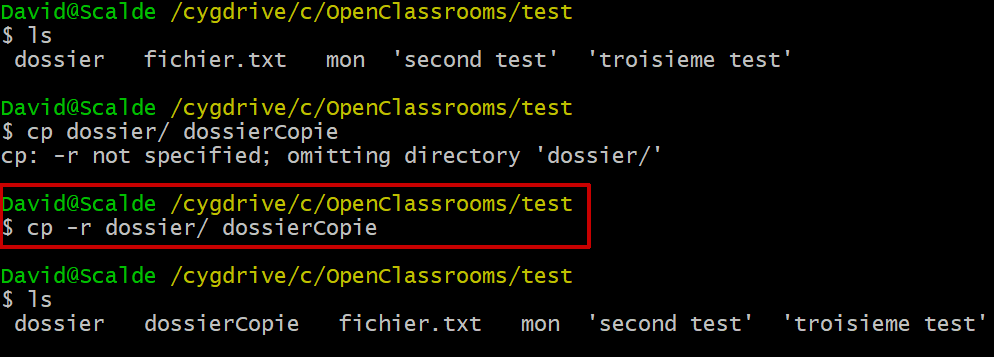
Après la copie, nous pouvons voir que le fichier est**présent à deux endroits**, là où je suis et dans le répertoire “dossier”.

Second exemple : pour**copier le fichier** “fichier.txt”**dans le même répertoire**, mais avec un **nouveau nom**, ici “fichier2.txt”.



cp fichier.txt fichier2.txt

Dernier exemple, **la copie d’un répertoire** :



cp -r dossier/ dossierCopie

Notez le message d'erreur :

cp -r not specified

Celui-ci est apparu parce que j'ai tenté de faire la copie directement, comme je l'aurais fait pour un fichier. Cependant, le terminal a ici indiqué qu’il manquait l’option “-r”.

En effet, copier un répertoire signifie copier également les éventuels sous-répertoires et fichiers qu’il contient, ce qui peut concerner **potentiellement beaucoup de fichiers** (imaginez que vous copiez le répertoire "Windows", par exemple...).

Le terminal demande donc une **confirmation** avec l’**option -r** (r pour “récursif” qui signifie au terminal qu’il va devoir aller à l’intérieur du dossier, et des sous-dossiers, et ainsi de suite).

Une fois la commande lancée avec cette option, la copie s’effectue alors sans problème.

### 

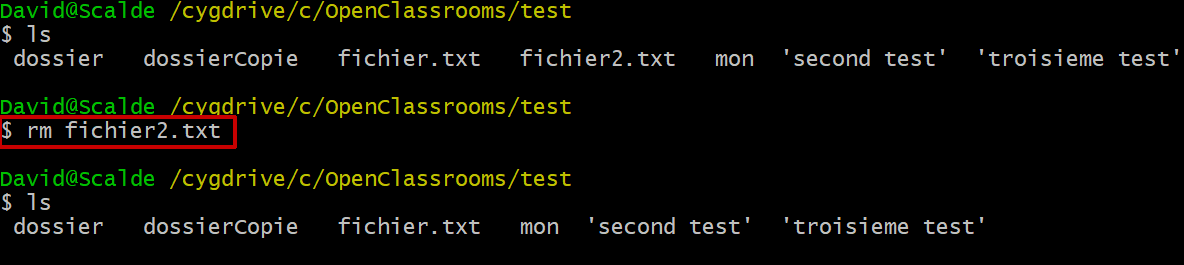
### ****Supprimez des fichiers et des répertoires avec rm****

Dernier point et pas des moindres, la **suppression**.

La suppression se fait avec **rm**, remove, qui signifie tout simplement “supprimer”, en français.

Soyez vigilant avec cette commande car elle supprime du contenu ! N'hésitez pas à faire vos tests dans un répertoire à part pour plus de sécurité.

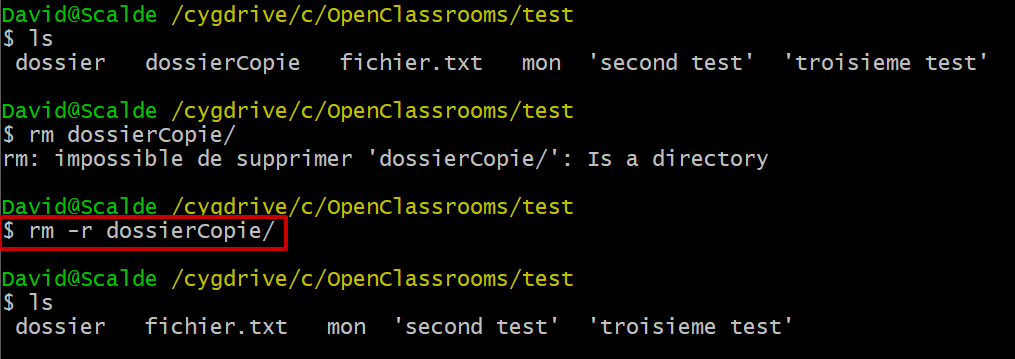
La syntaxe de rm est très simple : ***rm elementASupprimer***. Par exemple, pour supprimer le fichier “fichier2.txt” je n’ai qu’à écrire rm fichier2.txt.



rm fichier2.txt

Et avec ls nous avons vérifié que ce fichier a bien été supprimé.

Essayons maintenant de **supprimer un répertoire**. Ici, je tente de supprimer le répertoire “dossierCopie”.



rm -r dossierCopie/

Comme pour ce que nous avons vu avec cp, supprimer un répertoire signifie également **supprimer ses éventuels sous-répertoires** et fichiers. Par**mesure de sécurité**, la commande rm exige que l’on précise que l’on veut réellement supprimer le dossier et ***tout ce qu’il contient***.

Pour cela, c’est la même option à utiliser, **-r**.

Ce qui donne au final : rm -r dossierCopie.

Dans ce chapitre, vous avez appris à :

* **créer** des **dossiers** et des **fichiers**;
* les **déplacer** ;
* les **supprimer**.

Nous sommes presque à la fin de ce cours d’introduction à la ligne de commande.

Le prochain chapitre explorera d’autres commandes pour aller **plus loin** encore et vous donner un aperçu de toute la puissance du terminal, au-delà de la manipulation de fichiers !

## Allez plus loin avec de nouvelles commandes

Vous devriez maintenant être plus à l’aise avec l’utilisation du terminal.

Dans ce dernier chapitre, nous allons nous éloigner de la simple manipulation de dossiers et fichiers via le terminal pour **explorer d’autres possibilités**.

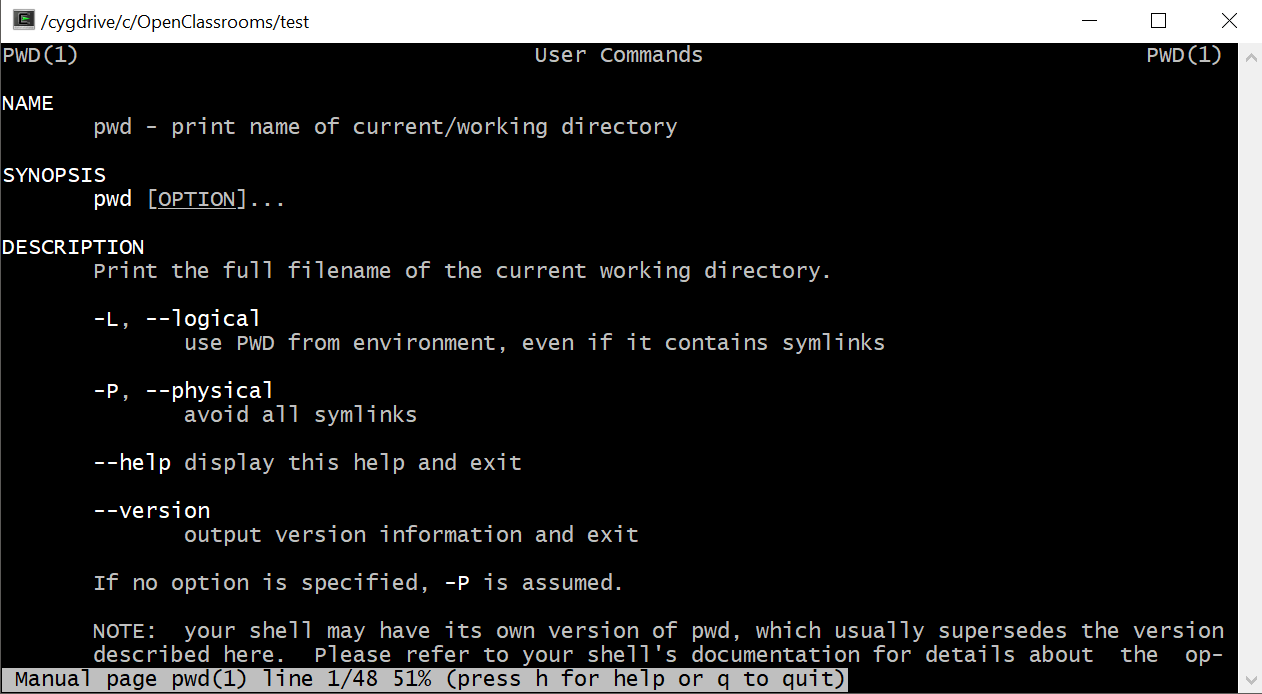
### 

### ****Affichez le manuel d’une commande avec man****

 La première de ces commandes à explorer est le “**man**”.

La commande “man”**affiche l’aide** associée à une commande. Pour cela, il suffit de taper **man commande**.

Par exemple pour afficher l’aide de la commande pwd, on tapera man pwd :



man pwd

Comme vous pouvez le constater,**l’écran entier** a été **remplacé par l’aide** de la commande pwd.

Pour naviguer dans cette aide, vous pouvez utiliser flèche haut et flèche bas. Et pour sortir de cette aide, vous pouvez taper sur la touche q (q pour pour "quitter").

En cas de **doute** sur le **fonctionnement** d’une commande, ou pour connaître l’**ensemble des options**disponibles, **man** sera votre meilleur allié.

Si vous tapez “**man ls**” vous verrez qu’il existe de **nombreuses autres options** que nous n’avons **pas explorées**. Et vous pouvez même taper “man man” pour avoir l’aide de man :-).

### 

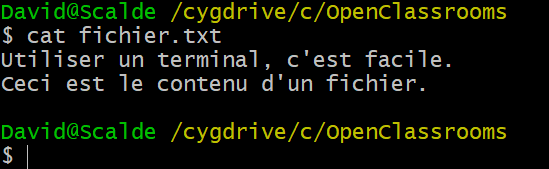
### ****Utilisez cat, less et more pour connaître le contenu d’un fichier****

Nous savons comment connaître le contenu d’un répertoire avec ls, mais comment connaître le contenu d’un fichier ?

Pour cela il existe cat, less et more.

Ces trois commandes ont le même but, à savoir**afficher le contenu d’un fichier**.

Par exemple, si je veux afficher le contenu du fichier “fichier.txt”, je vais taper cat fichier.txt.



cat fichier.txt

Le contenu du fichier s’affiche tel quel dans le terminal.

Pour “less”, la différence, c’est que cet affichage va être “paginé”. C’est-à-dire qu’au lieu d’afficher le contenu directement sous la commande, le terminal va utiliser le même mode de visualisation qu’avec la commande “man”.

Ainsi, si le fichier est très long, vous pourrez naviguer dedans à l’aide de flèche haut et flèche bas, et pour quitter, là encore, il suffira de taper q.

More est une ancienne version de “less”, avec des options navigation en moins. Je précise cependant son existence ici, car il est encore très répandu.

### Découvrez les redirections

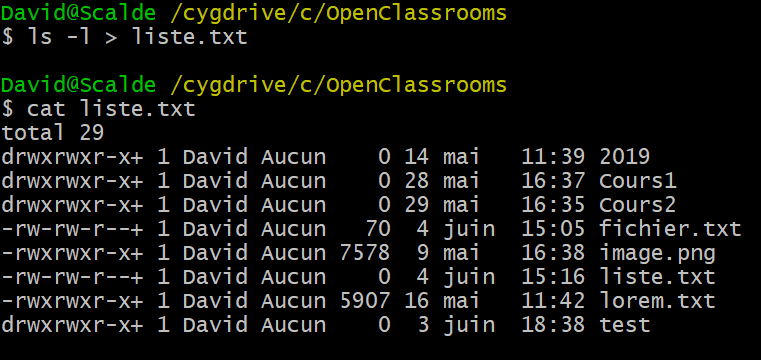
Ce qui suit n’est pas réellement une commande mais une “**redirection**”.

Il en existe plusieurs types, mais nous ne verrons ici que la plus commune : “**>**”.

Une redirection consiste à **prendre la “sortie” d’une commande** (par exemple pour “ls”, la sortie est tout simplement : afficher la liste des répertoires à l’écran) **et** à **la rediriger** vers autre chose, ici vers un fichier.

L’intérêt est notamment de pouvoir garder une trace des commandes que l’on a pu faire, en enregistrant leur résultat dans un fichier pour s’en resservir plus tard.

Voici un exemple simple pour mieux comprendre :



ls -l > liste.txt

ls -l est censé afficher la contenu de mon répertoire courant. Cependant, vous pouvez constater que ls -l > liste.txt **n’affiche rien**.

En revanche, si j’affiche le contenu du fichier “liste.txt” (grâce à la commande “cat” que nous venons juste de voir), alors nous retrouvons bien la liste des fichiers. La sortie de ls (c'est-à-dire l'affichage sur l'écran) a été **redirigée à l’intérieur du fichier**.

### 

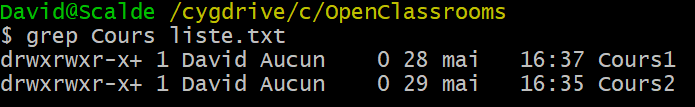
### ****Appréhendez grep****

La dernière commande que nous allons voir dans ce chapitre est **grep**.

Grep sert à **chercher des éléments à l’intérieur d’un fichier** sans même avoir besoin de les ouvrir.

La syntaxe est : grep ceQueVousVoulezChercher làOùVousVoulezChercher

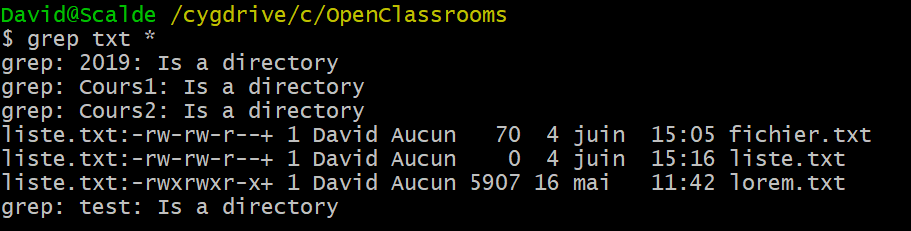
Par exemple, si vous voulez afficher toutes les lignes qui contiennent le mot “Cours” dans le fichier liste.txt, il faudra taper grep Cours liste.txt



grep cours liste.txt

Et vous pouvez constater que les deux lignes contenant le mot “**Cours**” se sont affichées à l’écran.

Si nous voulons chercher par exemple “txt” dans l’intégralité des fichiers du répertoire courant, alors la commande sera grep txt \*.



grep txt \*

Ce qui s’est passé, c’est que grep a cherché la chaîne “txt” dans tous les éléments du répertoire courant un par un (grâce à **\*** que nous avons vu précédemment et qui permet de cibler l’ensemble des fichiers).

Comme grep fait ses **recherches dans des fichiers**, lorsqu’il rencontre un élément qui est un répertoire, il affiche un message pour préciser que cet élément est un répertoire et qu'il ne peut donc pas vérifier si la chaîne se trouve bien à l'intérieur.

Par exemple, ici, “2019”, "Cours1", "Cours2" et "test" sont un répertoire.

Lorsque grep rencontre des fichiers, il les **vérifie et soit il reste silencieux** quand il n'y a **rien à l'intérieur qui corresponde à la recherche** (par exemple, le fichier "lorem.txt" ne contient pas, à l'intérieur, la chaîne "txt") **soit il affiche le contenu qui correspond** (par exemple “liste.txt” contient trois lignes avec “txt” dedans, et ces lignes sont bien affichées).

### ****En résumé****

Dans ce chapitre, nous avons appris :

* **man** pour **afficher le manuel** ;
* **cat/less/more** pour**afficher un contenu**;
* la **redirection** “**>**” pour envoyer le **résultat** d’une commande **à l’intérieur d’un fichier**;
* **grep** pour **faire des recherches**.

N’hésitez pas à faire quelques tests afin de bien prendre en main ces différentes commandes. En particulier, “man” peut vous donner de précieuses informations sur les options disponibles pour chacune d’entre elles.

## Faites le bilan

 Vous devriez désormais être un peu **plus à l’aise** avec l’utilisation de la ligne de commande.

Comme il était dit en introduction, ce cours n’avait pas pour vocation à vous apprendre tout ce qu’il est possible de réaliser avec le terminal, mais simplement de vous faire **prendre conscience** que cet **outil**, même s’il peut sembler un peu aride au premier abord, est **très simple** d’utilisation.

Si vous voulez aller plus loin, je vous conseille le [**cours de Mathieu Nebra**](https://openclassrooms.com/fr/courses/43538-reprenez-le-controle-a-laide-de-linux/37813-la-console-ca-se-mange). Ce cours concerne initialement Linux, mais **reprend en détail l’utilisation du terminal** en étant bien **plus exhaustif**.

Je pense que vous êtes prêt ! 😀

Pour rappel, voici toutes les commandes que nous avons vues ensemble, tout au long de ce cours :

* pwd : permet d’afficher le répertoire courant ;
* ls : permet d’afficher le contenu d’un répertoire ;
  + ls -a : l’option -a affiche également les fichiers et dossiers cachés,
  + ls -l : l’option -l modifie l’affichage pour rajouter de nombreuses informations ;
* cd dossier : permet de se déplacer à l’intérieur d’un répertoire ;
* mkdir dossier : permet de créer un dossier ;
* touch nomFichier : permet de créer un fichier ;
* mv source destination : permet de déplacer des éléments ;
* \* : est un caractère qui peut être utilisé comme substitut pour n’importe quel caractère dans une recherche ;
* cp source destination : permet de copier des éléments ;
  + cp -r : l’option -r permet de copier un répertoire ;
* rm fichiers : permet de supprimer des fichiers ;
  + rm -r dossiers : l’option -r permet de supprimer des répertoires ;
* man commande : permet d’afficher le manuel d’une commande ;
* cat/less/more nomFichier : permet d’afficher le contenu d’un fichier ;
* > : permet de rediriger le résultat d’une commande vers un fichier ;
* grep motif chemin : permet de faire des recherches dans des fichiers.

À bientôt !

# Testez-vous en situation réelle !

Bravo ! Vous avez réussi cet exercice !

### Compétences évaluées

* Manipuler des fichiers avec un terminal

### Description

Pour réaliser ce quiz, vous aurez besoin de télécharger le fichier [**zip**](https://s3-eu-west-1.amazonaws.com/course.oc-static.com/courses/6349650/quiz.zip).

Ouvrez votre terminal sur votre ordinateur. C'est depuis ce terminal que vous allez chercher votre dossier "quiz" que vous venez de télécharger.

Le but de ce quiz est de vous tester sur l'ensemble des commandes que vous avez pu aborder durant ce cours d'introduction.

### Question 1

**Placez-vous dans le dossier "quiz" à l'aide de la commande "cd".**

**Depuis ce dossier, si vous tapez la commande "ls", qu'obtenez-vous ?**

* + 

. .. 2019 fichier.txt image.png lorem.txt test test2

* + 

Photos Vidéos

* + 

2019 fichier.txt image.png lorem.txt test test2

* + 

Le contenu du fichier "lorem.txt"

*Le dossier "quiz" contient deux répertoires : "2019" et "test", ainsi que 4 fichiers ("fichier.txt", "image.png", "lorem.txt" et "test2").*

*Comme la commande ls n'a pas été lancée avec l'option "-a", les dossiers cachés . et .. ne devraient pas apparaître.*

*"Photos" et "Vidéos" sont des dossiers contenus dans le dossier 2019..*

### Question 2

**Si vous tapez la commande "cat fichier.txt", que voyez-vous ?**

* + 

"Lorem impsum dolor sit amet...".

* + 

Le terminal affiche une erreur.

* + 

Le contenu du répertoire "quiz".

* + 

"Ceci est un fichier. Le terminal, c'est facile."

*"cat fichier.txt" permet d'afficher le contenu du fichier appelé "fichier.txt". Ce fichier contient "Ceci est un fichier. Le terminal, c'est facile. "*

### Question 3

**Tapez la commande "ls -la > liste.txt". Que se passe-t-il ?**

* + 

Le terminal n'affiche pas le résultat de ls, mais un fichier liste.txt avec ce résultat a été créé.

* + 

Le terminal affiche le résultat de ls et un fichier liste.txt a été créé.

* + 

Le terminal affiche le contenu du fichier "liste.txt".

* + 

Le terminal a créé un dossier appelé "liste.txt".

*Le caractère ">" est ici une redirection. Donc ce qu'aurait dû afficher la commande "ls -la" est redirigé à l'intérieur du fichier "liste.txt". Donc rien n'est affiché, mais un fichier avec le résultat de la commande est bien créé.*

### Question 4

**Si je me positionne dans le dossier quiz et que je tape "ls 2" puis tab et enfin entrée, que vais-je voir ?**

* + 

Un message d'erreur, le répertoire "2" n'existe pas.

* + 

Le contenu du répertoire "2".

* + 

Photos Vidéos.

* + 

Le contenu du fichier 2019.

*Le répertoire quiz contient un dossier appelé 2019. Si je tape ls 2 puis tab, alors le terminal va compléter automatiquement la ligne, et la commande va devenir "ls 2019". En l'exécutant avec entrée, nous verrons donc le contenu du dossier 2019, c'est-à-dire "Photos"  et "Vidéos".*

### Question 5

**Dans le dossier quiz, il y a deux éléments appelés "test" et "test2". Quelle est la nature de ces deux éléments ?**

* + 

"test" est un répertoire et "test2" un répertoire.

* + 

"test" est un fichier et "test2" un fichier.

* + 

"test" est un fichier et "test2" un fichier.

* + 

"test" est un répertoire et "test2" un fichier.

*Avec la commande ls -l nous pouvons constater que test est un répertoire (la ligne commence par "d" pour "directory") et test2 est un fichier (la ligne commence par un -).*

*Il est tout à fait possible pour un fichier de ne pas avoir d'extention (comme .txt ou .png). De la même manière, il est tout à fait possible d'avoir un répertoire qui s'appellerait "dossier.txt". Le fait d'avoir seulement les noms de fichier qui finissent par une extension n'est qu'une convention et n'est en rien obligatoire.*

### Question 6

**Si depuis le dossier "quiz", je tape dans l'ordre :**

* + cp fichier.txt fichier2.txt
  + cp lorem.txt fichier2.txt
  + less fichier2.txt

**que va afficher cette dernière commande ?**

* + 

Rien, fichier2.txt est vide.

* + 

Le même contenu que fichier.txt.

* + 

Le même contenu que lorem.txt.

* + 

Cet enchaînement n'est pas valide et le terminal va retourner un message d'erreur.

*La première commande fait une copie de fichier.txt vers "fichier2.txt". À ce stade, fichier2.txt possède le même contenu que "fichier.txt".*

*La seconde commande fait une copie de "lorem.txt" vers "fichier2.txt". Même si "fichier2.txt" existe déjà, il va être remplacé par la copie de "lorem.txt". Donc à la fin de la seconde commande, "fichier2.txt" est une copie de "lorem.txt".*

*La dernière commande affiche le contenu de fichier2.txt. Comme il a été copié à partir de "lorem.txt", l'affichage sera le même que celui que l'on aurait eu en tapant "less lorem.txt".*

### Question 7

**Depuis le répertoire quiz, si je tape la commande "rm test", que va-t-il se passer ?**

* + 

Le dossier "test" sera supprimé.

* + 

Tous les éléments commençant par "test" seront supprimés.

* + 

Un message d'erreur va apparaître.

* + 

Un fichier "remarque" (rm) va être ajouté à l'intérieur du dossier test.

*Test est un dossier, donc lorsqu'on tape rm test, le terminal refuse de faire la suppression, car il exige que l'on spécifie l'option "-r", pour bien préciser que l'on est d'accord pour supprimer le dossier test ainsi que tout ce que, potentiellement, il pourrait contenir.*

*Pour supprimer tous les éléments commençant par test, il aurait fallu utiliser un joker et taper "rm test\*" (là encore, pour la suppression des dossiers, le -r est requis).*

### Question 8

**Pour créer un dossier appelé "photo", quelle commande devez-vous taper ?**

* + 

mkdir photo

* + 

makedir photo

* + 

createdir photo

* + 

cd photos

*Pour créer un nouveau dossier, la commande à utiliser est "mkdir" (qui signifie "make dir", "créer un répertoire", en français).*

*Makedir et createdir ne sont pas des commandes qui existent et cd (change directory) permet de changer de répertoire de travail.*

### Question 9

**Combien de fichiers seront créés si vous tapez la commande : "touch mon fichier second\ fichier" ?**

* + 

Aucun

* + 

1

* + 

2

* + 

3

*Par défaut, le terminal considère l'espace comme un séparateur. Cependant, l'antislash est un inhibiteur qui permet de dire au terminal que l'espace qui suit doit être considéré comme une "lettre normale". Donc ici, nous avons créé trois fichiers :*

* + *mon ;*
  + *fichier ;*
  + *second fichier.*

### Question 10

**À propos de  la commande "man man", quelles affirmations sont vraies ?**

*Attention, plusieurs réponses sont possibles.*

* + 

Elle affiche l'aide de la commande "man".

* + 

Elle écrit "man" à l'écran.

* + 

Elle affiche le contenu du fichier "man".

* + 

Pour sortir du mode manuel, il faut appuyer sur q.

*La commande "man" affiche l'aide d'une autre commande. "man man" affiche donc l'aide de la commande passée en paramètre, ici "man".*

*Une fois lancée, le terminal est entièrement remplacé par l'aide, et une fois que l'on a fini de la consulter, il suffit d'appuyer sur q(pour quit) pour sortir du mode "manuel".*