

# Le système Linux

Temps d'étude conseillé : 3h.

Durant ce premier TD, nous vous proposons de découvrir le système Linux au travers de la distribution Ubuntu. L'objectif final de cet atelier est de vous rendre autonome vis-à-vis du terminal, c'est à dire savoir exécuter des tâches basiques permettant de se déplacer ou manipuler des fichiers.

## 1 Avant c'était tout noir !

Parce qu'il y a quelques années encore, l'informatique n'était pas ce que l'on connaît aujourd'hui, piqûre de rappel pour tout le monde. Nous sommes en 1985, l'informatique est ouvert depuis peu au grand public. Pour avoir un aperçu de l'histoire, nous allons réaliser les exercices suivants en utilisant la *ligne de commande*. Cela permet d'utiliser l'ordinateur sans la souris et donc sans se fatiguer (c'est que vous diront les geeks).

### 1.1 Introduction au terminal

Une fois que votre système Linux est démarré, vous avez accès au terminal. Pour cela, saisissez **terminal** dans la barre de recherche de votre OS. Vous voyez apparaître une fenêtre comportant du texte sur fond noir : votre terminal est ouvert et vous donne quelques informations. Observons ce que votre enseignant obtient sur son poste.

```
potoo@TeamIbijau:~$
```

Le *prompt* vous donne les trois informations suivantes :

- **potoo** rappelle votre login ;<sup>1</sup>
- **TeamIbijau** indique le nom de votre machine (information particulièrement utile si vous avez un problème de réseau) ;
- le symbole **:** sert de séparateur ;
- le symbole **~** indique que vous êtes dans le répertoire d'accueil (nous verrons cette notion plus tard) ;
- le symbole **\$** indique que le prompt attend la saisie d'une commande.

Dans tout ce chapitre, nous supposons que nous manipulons un terminal dont l'utilisateur est **potoo**. Vous devez donc adapter les informations de ce document à votre cas en remplaçant le login **potoo** par votre login.

Imaginez ce terminal comme un explorateur de documents. Vous vous situez dans un dossier particulier et vous pouvez réaliser des actions sur ce dossier telles que : vous déplacer dans un autre dossier, ajouter un nouveau document, en supprimer un, lister l'ensemble des fichiers de ce dossier etc. Mais avant de réaliser ces actions, il est nécessaire de comprendre l'organisation d'un système Linux.

1. La recherche de ce mot dans Google est à vos risques et périls.

## 1.2 Tous les chemins mènent à Rome

L'organisation d'un système Linux diffère quelque peu de celle d'un Windows<sup>2</sup> dans le sens où l'ensemble des informations accessibles par votre ordinateur le sont à partir d'un unique dossier : la *racine*. Cette propriété s'appelle *l'arborescence* du système de fichier.

**Définition 1 (Racine).** La *racine* est le dossier parent de tous les dossiers. Il est noté `/`.

À partir de ce dossier parent, vous avez accès à un ensemble de dossiers, parmi lesquels on trouve entre autres :

- le dossier **bin**, contenant un ensemble d'exécutables nécessaires à une utilisation basique de l'OS ;
- le dossier **boot**, contenant les informations nécessaires au lancement de la machine ;
- le dossier **media**, permettant d'accéder aux média amovibles (clés USB, CD-rom ...) ;
- de nombreux autres dossiers dont vous découvrirez l'utilisation au cours du temps.

L'objectif de cette section est de savoir se repérer et évoluer dans un système Linux. On utilise pour cela une série de *commandes* qui ne sont rien de plus que des instructions à réaliser dans le dossier dans lequel le terminal se situe.

### Exercice 1. Se repérer.

Saisissez la commande `pwd` afin de savoir dans quel dossier vous vous situez actuellement sur la machine.

**Définition 2.** Ce dossier est traditionnellement nommé *répertoire d'accueil* et se note aussi `~`. Vous avez un accès illimité à ce dossier, aussi bien en lecture qu'écriture.

En partant de la racine du système, la commande `pwd` vous donne la liste des dossiers à atteindre avant d'arriver sur le dossier sur lequel vous êtes.

**Définition 3.** Le *chemin absolu* d'un élément indique l'ensemble des dossiers contenant cet élément depuis la racine. Il commence donc toujours par `/`.

Le reste de l'arborescence (sauf *media*) vous est globalement soit inutile, soit interdit d'écriture / suppression. Le déplacement depuis un dossier vers un autre s'effectue à l'aide de la commande `cd` suivie immédiatement du chemin vers le dossier à atteindre. Afin de remonter facilement vers le dossier parent, il existe un raccourci magique noté « `..` ».

**Définition 4.** Le dossier `..` est un raccourci indiquant le dossier parent à tout dossier. La racine (`/`) est le seul dossier à ne pas posséder de parent.

### TD Exercice 2.

**Définition 5.** Un chemin *relatif* permet d'indiquer l'emplacement d'un objet à partir du dossier courant.

2. Un système Windows est organisé sous la forme de disques : le disque **C:** contenant souvent les programmes et le système d'exploitation, un disque pour les CDs, un disque pour les clés USB...

Retenez la différence entre chemin relatif et chemin absolu. Vous apprendrez très rapidement à utiliser le dossier `..` !

## 1.3 Le terminal et ses commandes

| commande <i>argument</i> | Utilité   |
|--------------------------|---|
| <code>pwd</code>         | Affiche le chemin absolu du répertoire courant              |
| <code>mkdir Rep</code>   | Crée le dossier <i>Rep</i> dans le répertoire courant       |
| <code>rmdir Rep</code>   | Supprime le répertoire <i>Rep</i>                           |
| <code>cd Rep</code>      | Se déplacer dans le répertoire <i>Rep</i>                   |
| <code>ls</code>          | Liste le contenu du répertoire actuel                       |
| <code>touch File</code>  | Crée le fichier <i>File</i> dans le répertoire courant      |
| <code>rm File</code>     | Supprime le fichier <i>File</i>                             |
| <code>cp src dst</code>  | Copie le fichier <i>src</i> à l'emplacement <i>dst</i>      |
| <code>mv src dst</code>  | Déplace le fichier <i>src</i> vers l'emplacement <i>dst</i> |

TABLE 1 – Les principales commandes unix

Afin de manipuler plus facilement chemins relatifs et absolus, voyons les commandes disponibles dans votre terminal. Le tableau 1 liste les principales commandes à connaître<sup>3</sup>.

**Astuce.** Il est possible de relancer facilement une commande précédemment exécutée en appuyant sur la flèche du haut du clavier. Vous pouvez également saisir les commandes et les noms de dossier plus rapidement en commençant à saisir un mot puis en appuyant sur la touche tabulation pour compléter automatiquement le mot en cours de saisie.

### TD Exercices 3 et 4.

La liste des commandes et leur fonctionnement sont à connaître par cœur !

## 1.4 Le manuel

La commande ultime sous Linux reste l'appel au manuel qui se réalise en tapant `man` suivi du nom d'une commande. Qu'il soit en français ou en anglais, le formatage est identique. Une *page de man* se compose de plusieurs champs dont les trois premiers sont le nom de la commande (contenant un résumé de la commande), le synopsis permettant d'un seul coup d'œil de voir l'ensemble des options disponibles, et la description contenant l'explication de chaque option.

**Exemple 1.** Étudions l'exemple de la page de man de la commande `mkdir`. Lancez donc la commande `man mkdir` dans le terminal et étudions le résultat petit à petit.

|           |                               |           |
|-----------|-------------------------------|-----------|
| MKDIR(1L) | Manuel de l'utilisateur Linux | MKDIR(1L) |
|-----------|-------------------------------|-----------|

Cette première partie indique la section de manuel utilisée (1L) et le nom de la commande.

3. Une liste bien plus complète peut se trouver, entre autres, sur Wikipédia : [https://fr.wikipedia.org/wiki/Commandes\\_Unix](https://fr.wikipedia.org/wiki/Commandes_Unix)

NOM  
mkdir - Créer des répertoires.

SYNOPSIS  
mkdir [-p] [-m mode] [--parents] [--mode=mode] [--help] [--version] rep...

On trouve ensuite le **NOM** de la commande avec un très court résumé ainsi que le **SYNOPSIS**. Il s'agit du moyen d'utiliser la commande. Les éléments entre crochets sont la liste des options décrites plus bas. Le dernier mot (**rep**) est un *argument* obligatoire de la commande. Les trois points indiquent qu'il est possible de saisir autant d'arguments que souhaité.

DESCRIPTION  
Cette page de manuel documente la version GNU mkdir.

mkdir crée un répertoire correspondant à chacun des noms mentionnés.

Par défaut, les répertoires sont créés avec les permissions d'accès 0777 moins les bits positionnés dans le umask.

La section **DESCRIPTION** indique ensuite des informations plus complètes sur la commande.


OPTIONS

-m, --mode mode  
Créer les répertoires avec le mode d'accès indiqué. Celui-ci est donné sous forme symbolique, comme dans `chmod(1)`, en utilisant le mode par défaut comme valeur de départ.

-p, --parents  
S'assurer que chaque répertoire indiqué existe. Créer les répertoires parents manquants. Ces derniers sont créés avec l'autorisation d'accès umask modifiée par ``u+wx'`. Ne pas considérer les répertoires déjà existants comme des erreurs.

--help Afficher un message d'aide sur la sortie standard, et terminer normalement.

--version  
Afficher un numéro de version sur la sortie standard, et terminer normalement.

On trouve enfin la liste des **OPTIONS** avec, si existants, leur raccourci. Par exemple, l'option `-m` est le raccourci de l'option `--mode` et doit être suivie d'une valeur remplaçant l'argument `mode`. 

### TD Exercice 5.

Le man est votre ami ! Il est toujours présent, même dans les moments les plus difficiles pendant lesquels vous n'aurez pas accès à Internet.

## 2 Un peu de couleur

Le terminal n'est qu'une interface entre vous et le contenu de votre ordinateur. Il permet de réaliser les mêmes actions qu'avec l'explorateur de documents comme voir

le contenu d'un dossier, ajouter / déplacer / supprimer des fichiers etc. Comparons les actions disponibles dans le terminal et celles présentes dans l'explorateur de documents.

## 2.1 Équivalence du mode graphique et du mode console

Tapez à nouveau dans la console la commande permettant de vous situer. Dans l'explorateur de documents, trouvez le dossier indiqué par la console. Mettez la fenêtre de navigation à coté de la console pour avoir les deux en vue simultanée.

### TD Exercice 6.

## 2.2 Les éditeurs de texte

Maintenant que vous savez vous déplacer, vous repérer et supprimer des fichiers et documents, il est temps d'apprendre à créer des fichiers! Plusieurs moyens s'offrent à vous selon les usages que vous souhaitez faire du fichier.

**L'éditeur bureautique (type traitement de texte)** : il est principalement utilisé pour du traitement de texte bien sûr, donc non adapté à la programmation. La raison est que le texte écrit est formaté dans un type précis, tout comme la mise en page etc. Ces informations n'apparaissent pas lorsqu'on ouvre le document mais sont présentes dans le fichier. Pour preuve regardez la taille d'un document bureautique contenant uniquement quelques mots...

**L'éditeur de texte (type emacs, gedit)** : le fichier contenant le texte entré est directement codé suivant la table ASCII (1 caractère = 1 code sur 1 octet). N'importe quel éditeur de texte brut sur des OS différents peut donc afficher le texte contenu dans le fichier.

**L'éditeur en ligne de commande (type vim, nano)** : il possède les mêmes fonctions que l'éditeur de texte mais n'est pas en mode graphique. Tout se passe dans la console et les actions telles que la sauvegarde et la modification sont réalisées par le biais d'une combinaison de touches (`esc` : `w` pour sauvegarder sous vim par exemple). L'avantage principal est sa légèreté et peut donc être implanté dans des environnements disposant de peu de ressources.

**L'Environnement de Développement Intégré (type Code::Blocks)** : l'IDE combine l'éditeur de texte, le compilateur, le débbugger etc. Il est donc ultra-complet et permet de gérer facilement des gros projets composés de plusieurs dizaines de fichiers.

Pour le moment, nous allons coder nos programmes dans un, voir deux fichiers. Utiliser un IDE est donc inutile et nécessite tout de même une prise en main pas nécessairement intuitive. Reste donc `vim`, `gedit` et `emacs`. Libre à vous d'utiliser celui que vous préférez, nous préconisons tout de même l'emploi de `gedit` ou équivalent, pour plus de simplicité.

Ouvrez l'éditeur de texte pour créer un fichier nommé *bonjour.c* puis tapez le code suivant en respectant scrupuleusement la syntaxe et la mise en page :

```
#include <stdio.h>

int main()
{
    printf("Les bijoux ne sont pas des hiboux.\n");
    return 0;
}
```

**TD** Exercice 7.

L'extension d'un fichier permet de sélectionner le programme qui ouvre par défaut le fichier. La changer ne modifie en rien le fichier, et donc ne le convertit pas.

## 2.3 Les droits

Linux est fondé sur un système de droits : chaque utilisateur de la machine est référencé dans un groupe et possède certains droits sur les fichiers concernant leur lecture, écriture ou exécution. L'option `-l` de la commande `ls` permet de visualiser en début de ligne les droits associés à ce fichier. Pour chaque fichier et dossier, une ligne telle que ci-dessous apparaît :

```
drwxr-xr-x 16 util groupe 4096 2012-09-19 11:18 Documents
```

Voici les détails des champs :

**drwxr-xr-x** : 10 caractères présentant le type et les droits de l'élément. Le premier caractère indique le type de l'élément : nous avons donc ici un dossier (**d**), - dans le cas d'un fichier. Les trois caractères suivants indiquent les droits en lecture (**r**), écriture (**w**) et exécution (**x**) de l'utilisateur (**u**), un - indiquant que les droits sont absents, les trois suivants du groupe (**g**) et les trois derniers des autres personnes (**o**).


**16 util groupe** : **16** indique le nombre de liens physiques associés à cet élément (il y a 16 endroits différents dans l'ordinateur permettant d'accéder à **Documents**). **util** indique le nom de l'utilisateur possédant cet élément, **groupe** indique le nom du groupe de l'utilisateur.

**4096** : indique la taille de l'élément en blocs mémoire. Ce ne sont pas des octets !

**2012-09-19 11:18** : date de dernier accès.

**Documents** : nom de l'élément.

La commande `chmod` permet de changer les droits affectés à l'élément en utilisant les options `[ugoa] [+ -=] [rwx]`.

**Exemple 2.** La commande `chmod u+x toto.txt` permet d'ajouter (+) les droits en exécution (**x**) à l'utilisateur (**u**) sur le fichier *toto.txt*. 

**TD** Exercice 8.

On peut également attribuer un nombre à un élément indiquant ses droits : Le nombre possède alors trois chiffres, le premier pour **u**, le deuxième pour **g** et le dernier pour **o**. Chaque chiffre est une somme de puissance de 2. Le 1 représente les droits en exécution, 2 pour l'écriture et 4 pour la lecture.

**Exemple 3.** On souhaite accorder tous les droits à l'utilisateur, de lecture et d'exécution au groupe et de lecture seule aux autres. Le code pour tous les droits est `rwX` donc  $4+2+1 = 7$ . Les droits de lecture et d'exécution (`r-X`) valent respectivement 4 et 1 donc il faut affecter la valeur 5 au groupe. Enfin, la valeur 4 est affectée aux autres. On écrit alors `chmod 754 <fichier>`. ▲

**TD** Exercice 9.

**Astuce.** Les droits sont particulièrement utiles lorsque vous récupérez un exécutable provenant de l'extérieur de la machine (Internet, clé USB...). Ce dernier est alors privé des droits d'exécution. Un `chmod +x` sur votre fichier sera alors bien utile.

## 3 Pour aller plus loin

Ces questions vont nécessiter une recherche de votre part sur Internet. Le but est de vous faire découvrir des astuces pour vous faciliter la vie avec la console et les programmes disponibles sur Linux.

**TD** Exercice 10.

## 4 Synthèse

Cette dernière section sera présente dans l'ensemble des TD. Elle vous permettra de résumer les compétences que vous venez d'acquérir durant le TD. De plus il est fortement recommandé de créer une fiche révision à partir des informations de cette section. Et parce qu'on est gentil, la première fiche, c'est cadeau, c'est gratuit, c'est pour nous ! Vous trouverez donc en dernière page un exemple de fiche révision.

À la fin de ce chapitre, vous devez être capable de ...

- faire la différence entre chemin relatif et absolu et les utiliser à bon escient ;
- utiliser correctement les commandes `mkdir`, `cd`, `ls`, `chmod`, `rm`, `cp` et `mv` ;
- utiliser correctement le `man`.

**Fiche révision Chapitre 1 : Linux**

**Chemin relatif** : chemin vers un fichier à partir d'un dossier particulier. Exemple :  
`../Images/Linux.jpg`

**Chemin absolu** : chemin vers un fichier depuis la racine (/) de l'OS. Exemple :  
`/home/user/Images/Linux.jpg`

**Les commandes du terminal :**

- `pwd` : Affiche le chemin absolu du répertoire courant
- `mkdir Toto` : Crée le dossier *Toto* dans le répertoire courant
- `rmdir Toto` : Supprime le répertoire *Toto*
- `cd ../Images` : Remonter vers le dossier parent (..) puis se déplacer dans le répertoire *Images*
- `ls` : Liste le contenu du répertoire actuel
- `touch exp.txt` : Crée le fichier vide *exp.txt* dans le répertoire courant
- `rm exp.txt` : Supprime le fichier *exp.txt*
- `cp exp.txt Toto` : Copie le fichier *exp.txt* dans le dossier *Toto*
- `mv exp.txt exemple.txt` : Déplace le fichier *exp.txt* vers le fichier *exemple.txt* (donc renomme). ATTENTION : risque d'écrasement !

**La commande `chmod`** : change les droits d'un fichier/dossier. Droits en lecture (r ou 4), écriture (w ou 2), exécution (x ou 1). Droits attribués à l'utilisateur (u), au groupe (g), aux autres (o) ou à tout le monde (a).

Voir les droits : `ls -l`. Modifier les droits :

- `chmod +x toto.txt` ajoute le droit en exécution à *toto.txt* ;
- `chmod a-w toto.txt` supprime l'écriture à tout le monde ;
- `chmod 400 toto.txt` uniquement lecture pour l'utilisateur ;
- `chmod 777 toto.txt` tous les droits pour tout le monde ;