

Découverte de la console

I. Introduction

Qu'est ce que la console ? Pourquoi l'utiliser ?

II. Quelques généralités

Relativité des chemins

Les types de fichiers et les droits sous Unix

Décortiquons le prompt

III. Première approche

Qu'est-ce qu'une commande ? Un argument ?

Notre première commande

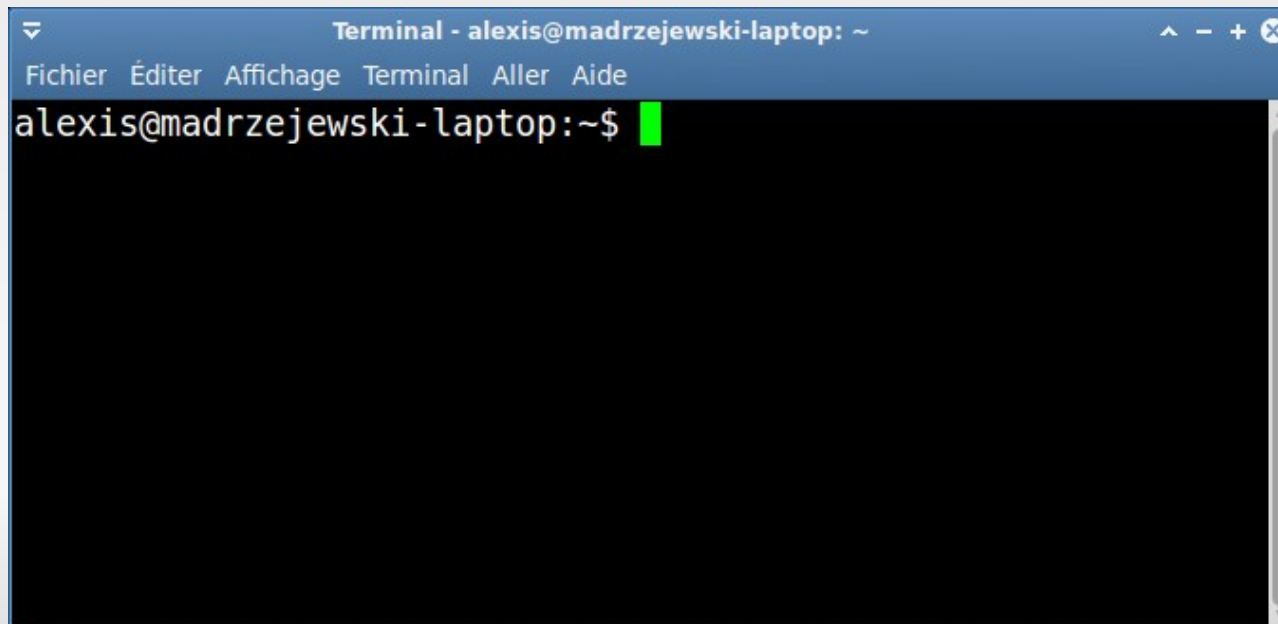
Quelques raccourcis pratiques

Introduction

- Qu'est-ce que la console ?

C'est un écran noir, qui est en attente d'une instruction. C'est une interface avec Unix. Il offre à l'utilisateur l'interface de base avec le système d'exploitation.

Sur Ubuntu, ça ressemble à cela :



```
Terminal - alexis@madrzejewski-laptop: ~
Fichier Éditer Affichage Terminal Aller Aide
alexis@madrzejewski-laptop:~$
```

Introduction

La console ci-dessus, est une console en mode graphique. Il faut savoir qu'il existe d'autres types de consoles, moins amicales accessible via le raccourci "ctrl-alt-f1" (jusqu'à f6 et f7 pour revenir en mode graphique).

Cette ligne :

```
alexis@madrzejewski-laptop:~$
```

est le prompt. Elle nous donne quelques informations que nous décortiquerons plus tard.

Introduction

- Pourquoi l'utiliser ?

La plupart des débutants se demandent pourquoi on utilise encore la console. A première vue, c'est dépassé, plus compliqué et en plus on ne peut pas utiliser la souris !

En réalité, la console est bien plus puissante qu'un environnement graphique courant. Elle permet de réaliser certaine tâche beaucoup plus rapidement qu'en utilisant la souris.

De plus, on apprend des commandes standard Unix. Il faut savoir que ces commandes existent depuis plus de 30 ans et reste identique. Donc en les apprenant une fois, on est tranquille pendant longtemps ! Ça vaut le coup de s'investir.

Introduction

- Pourquoi la ligne de commande est-elle aussi puissante ?

En 1972, Doug MacIlroy énonce les principes de la boîte à outil Unix :

1. écrire des programmes qui font une seule chose et qui le font bien,
2. écrire des programmes qui peuvent communiquer entre eux,
3. écrire des programmes qui manipulent du texte car c'est l'interface universelle.

Ce sont ces 3 règles qui rendent les lignes de commandes si puissantes même après 30 ans ...

Introduction

- Un exemple :

Imaginons que je souhaite compter le nombre d'images (jpg) contenues dans mon dossier "Images" qui se situe dans mon répertoire personnel. En mode graphique, comment je fais ? Je compte les fichiers un à un en prenant le risque de me tromper ?

En ligne de commande c'est très simple :

```
alexis@madrzejewski-laptop:~$ cd Images ; ls -l | grep jpg | wc -l  
42
```

Quelques généralités

■ Architecture des dossiers

Sous GNU/Linux, l'architecture des dossiers est très différentes de Windows. Voici un petit récapitulatif des différents dossiers important que l'on retrouve couramment.

/ racine	• bin	contient des programmes (exécutables)
	• home	répertoires personnel des utilisateurs
	• boot	fichiers permettant le démarrage de Linux
	• etc	fichiers de configuration
	• var	contient des logs
	• proc	contient des informations système
	• tmp	dossier temporaire utilisé par les programmes
	• usr	c'est ici que vont s'installer les programmes demandés par l'utilisateur
	• media	c'est ici que sont monté vos périphériques
	• opt	répertoire utilisé pour les ADD-ONS de programme
	• dev	fichiers contenant les périphériques
	• lib	contient les bibliothèques utilisées par les programmes
	• root	dossier personnel du super-utilisateur
	• ...	

Quelques généralités

- Notions de relativité des chemins

Il existe 2 manières d'écrire le chemin d'un fichier : de manière relative et absolue.

Absolute : la liste des différents répertoires traversés pour aller de la racine ('/') au fichier. Les différents répertoire sont séparés par le séparateur '/'.
Ex: /home/alexis/dossier1/fichier.txt

Ex: /home/alexis/dossier1/fichier.txt

Relative : la liste des différents répertoires traversés pour aller du répertoire courant (la ou on se situe actuellement) au fichier.

Ex: dossier1/fichier.txt (sachant que je me situe déjà dans le dossier "/home/alexis")

Quelques généralités

- Les types de fichiers Unix

Sous Unix, tout est un fichier ! Mais pour faire simple, il existe 2 grands types de fichiers : les normaux (txt, mp3, jpg) et les spéciaux (votre lecteur CD, clé usb, dossier etc..).

Voici une liste détaillée :

Type de fichier		Description
<u>Ordinaire (régulier)</u>	-	Texte, programme, son, image, vidéo etc...
<u>Répertoire</u>	d	Collection de fichiers et/ou répertoires
<u>Lien symbolique</u>	l	Pointeur vers un autre fichier (raccourcis)
<u>Tube nommé (fifo)</u>	p	Zone pour l'échange unidirectionnel d'octet entre processus. (pipe)
<u>Socket Unix</u>	s	Zone pour l'échange bidirectionnel d'octets (ou datagramme) entre processus
<u>Périphérique caractère</u>	c	Point d'accès à un périphérique en mode caractère (clavier, souris, imprimante ...)
<u>Périphérique bloc</u>	b	Point d'accès à un périphérique en mode bloc (disque, CD, disquette ...)

Quelques généralités

- Les droits sous Unix

La gestion des droits sous Unix est assez puissante sans être trop compliqué. Pour faire simple :

Un utilisateur a le droit de faire 3 choses sur un fichier : le lire, écrire (le modifier), et l'exécuter (comme un programme).

De la même manière, il existe plusieurs types d'utilisateurs : l'utilisateur propriétaire d'un fichier, le groupe propriétaire du fichier et les autres.

Le super utilisateur, c'est à dire l'utilisateur root, possède tous les droits. On n'a donc pas à s'en occuper pour attribuer des droits sur un fichier puisqu'il pourra tout faire.

Quelques généralités

- Les droits sous Unix

Français	Anglais	Console	Octal
Lecture	Read	r	4
Écriture	Write	w	2
Exécution	Execute	x	1
Propriétaire	User	u	
Groupe propriétaire	Group	g	
Les autres	Others	o	

Les droits en octal seront utiles lorsqu'on voudra changer les droits d'un fichier. Chaque droit est représenté par un chiffre. Si l'on souhaite donner le droit de lecture/écriture, on mettra un 6 car $4+2=6$. Ne vous inquiétez pas, on en reparlera plus tard.

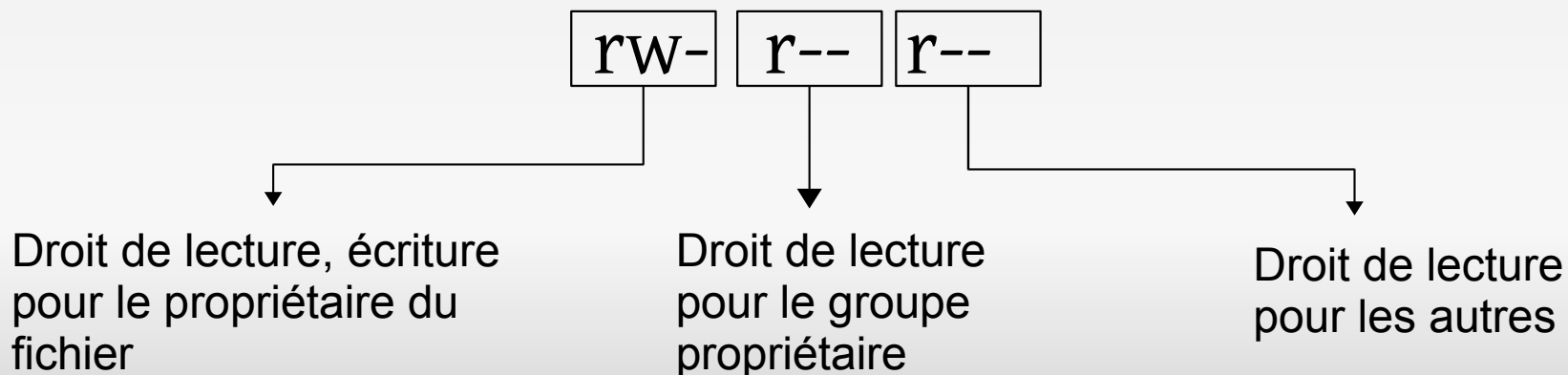
Quelques généralités

- Les droits sous Unix

La plupart du temps, les droits d'un fichier sont représentés sous forme de ligne. Par conséquent, il faut apprendre à lire ces droits qui respectent un formalisme précis.

```
-rw-r--r-- 1 alexis alexis 441 2010-12-11 17:02 fichier1.txt
```

Le bloc encadré en rouge, issue d'une ligne donnée par la commande "ls -l", donne les droits des utilisateurs pour le fichier : "fichier1.txt".



Quelques généralités

- Décortiquons le prompt

```
alexis@madrzejewski-laptop:~$
```

Indique votre niveau d'autorisation.
"\$" = utilisateur normal

Ce premier élément est votre pseudonyme.
(nom d'utilisateur)

Séparateur

Ce deuxième élément est le nom de votre machine. Ici :
"madrzejewski-laptop"

Séparateur

Indique le répertoire où vous vous situez.
"~" = répertoire personnel

```
root@madrzejewski-laptop:/home/alexis/Images#
```

Utilisateur root = administrateur

On a changé de répertoire

Autorisation de super-utilisateur

Première approche

- Commandes et arguments

Dans une console on travaille avec des "commandes". Ces instructions sont nombreuses et on ne peut pas toutes les apprendre. C'est pour cela qu'il existe un manuel qui nous permet d'avoir des informations sur ces commandes. On apprendra à utiliser ce manuel un plus tard.

Une commande peut recevoir une indication supplémentaire, c'est ce que l'on appelle un argument. Un argument est placé après une commande, généralement précédé d'un tiret (-) si il est court et 2 s'il fait plusieurs lettres. Certain argument on besoin d'une autre valeur pour être correct. Une commande peut recevoir plusieurs arguments.

Première approche

- Exemples :

>commande -D

Attention : la console est sensible à la casse, cela veut dire que "d" est différent de "D".

On exécute une commande avec comme argument "D".

>commande -d -a -e

>commande -dae

On exécute une commande avec plusieurs argument, "d" "a" et "e". Ces 2 commandes sont identique, la seconde est une simplification de la première (on peut le faire avec des paramètres courts).

>commande --parametrelong 20

On exécute une commande avec un paramètre long qui à besoin de la valeur 20.

Première approche

- Notre première commande

On possède désormais les bases nécessaire pour taper notre première vraie commande.

Une commande indispensable est "ls" (qui veut dire "list" en anglais). Elle permet de lister les différents fichiers contenu dans un répertoire. Si je fais un "ls" de mon répertoire tuto, voici ce que j'obtiens :

```
alexis@madrzejewski-laptop:~/tuto$ ls
```

```
IMAG0107.jpg output1.mkv plancmd.txt
```

On constate que l'on à une liste des fichiers contenu dans ce dossier. On remarque aussi qu'ils sont colorés en fonction de leur type (ce n'est pas toujours le cas).

Première approche

- Notre première commande

Imaginons que je souhaite avoir plus d'informations sur ces fichiers comme leur taille, ou l'auteur du fichier etc..

Il existe un argument qui nous permet d'avoir ces informations, c'est "-l" (long listing format).

```
alexis@madrzejewski-laptop:~/tuto$ ls -l
```

```
total 13716
```

```
-rwx----- 1 alexis alexis 1492343 2010-11-20 17:11 IMAG0107.jpg
```

```
-rw-r--r-- 1 alexis alexis 12517681 2010-11-25 14:32 output1.mkv
```

```
-rwx----- 1 alexis alexis 1150 2010-11-11 13:53 plancmd.txt
```

Droit et type
de fichier

Nombre
de
Lien

Utilisateur
Propriétaire

Nom du
groupe

Taille
(en octet)

Date de dernière
modification

Nom du fichier

Première approche

- Le manuel : RTFM

Sous Unix, chaque commande possède un manuel. Il est essentiel de savoir maîtriser cet outil qui répond à 90% des questions que l'on se pose sur une commande.

Pour ouvrir le manuel d'une commande, la syntaxe est la suite :

>**man** commande

S'affichera ensuite une page contenant des informations sur la commande avec une mise en page normalisé.

On va apprendre à lire ce manuel, qui n'est pas forcément évidemment pour un néophyte au premier coup d'œil !

Première approche

■ Le manuel : exemple de mkdir

MKDIR(1)	User Commands	MKDIR(1)	
NAME	_____→		Nom de la commande avec un court résumé
mkdir - make directories			
SYNOPSIS	_____→		Indique la manière dont on utilise la commande
mkdir [OPTION]... DIRECTORY...			
DESCRIPTION	_____→		Description plus importante.
Create the DIRECTORY(ies), if they do not already exist.			Liste exhaustive de tous les différents arguments avec une description. C'est la dedans que l'on passe la plupart de notre temps à la recherche d'informations.
Mandatory arguments to long options are mandatory for short options too.			
-m, --mode=MODE set file mode (as in chmod), not a=rwx - umask			
[...]			
--version output version information and exit			
AUTHOR	_____→		Nom de l'auteur
Written by David MacKenzie.			
REPORTING BUGS [...]	_____→		Email de contact en cas de bug
COPYRIGHT [...]			
SEE ALSO [...]	_____→		Voir aussi : parfois intéressant.
GNU coreutils 8.5	June 2010	MKDIR(1)	

Première approche

- Le manuel : exemple de mkdir

```
MKDIR(1)                                User Commands                                MKDIR(1)

NAME
  mkdir - make directories

SYNOPSIS
  mkdir [OPTION]... DIRECTORY...

DESCRIPTION
  Create the DIRECTORY(ies), if they do not already exist.

  Mandatory arguments to long options are mandatory for short options too.

  -m, --mode=MODE
      set file mode (as in chmod), not a=rwx - umask

  -p, --parents
      no error if existing, make parent directories as needed

  -v, --verbose
      print a message for each created directory

  -Z, --context=CTX
      set the SELinux security context of each created directory to CTX

  --help display this help and exit

Manual page mkdir(1) line 1
```

Première approche

- Quelques raccourcis 1/2

Il existe des raccourcis très pratique lorsqu'on travaille dans une console. Pour commencer, en voici quelques-uns pour travailler avec le manuel.

Touche raccourci	Action
↓ ↑	Se déplacer ligne par ligne dans le manuel.
Espace	Se déplacer d'une longueur d'écran dans le manuel (~page par page)
/mot à rechercher	Rechercher un mot dans le manuel
n	Va à la prochaine occurrence du mot recherché (next)
g	Remonter tout en haut du manuel
q	Quitter le manuel

Première approche

- Quelques raccourcis 2/2

En voici d'autres qui sont "plus général".

Touche raccourci	Action
↓ ↑	Permet de consulter l'historique des commandes afin d'éviter de les retaper (super pratique).
TAB	Permet de compléter automatiquement le nom d'une commande, ou d'un argument. Ça complète le texte à notre place. La encore, très utile !
CTRL + C	Arrêter la commande en cours (très utile)
CTRL + Z	Interrompt temporairement un processus, qui peut être relancé avec la commande fg (au premier plan) ou bg (en arrière-plan)
CTRL + L	Efface le contenu de la console (ça fait le ménage, il existe la commande "clear" qui fait la même chose)
CTRL + A	Ramène le curseur au début de la commande, pratique pour les longues commandes

Première approche

- Conclusion

Si une erreur s'est glissée dans ce document ou dans la vidéo qui l'accompagne, merci de me le signaler par email à l'adresse suivante :

bvek1.prof [at] gmail.com

Si vous avez des questions ou des remarques, vous pouvez laisser un commentaire sur le site :

<http://www.tutoriels-video.fr>

Bien évidemment, ce document est sous licence creative commons 2.0 ;)

Première approche

- Sources

Voici les différentes sources utilisées pour la création de ce document :

<http://www.ubuntu-fr.org/>

<http://www.siteduzero.com/>

Le livre « UNIX. Pour aller plus loin avec la ligne de commande » que je recommande fortement. Il est disponible gratuitement sur :

<http://www.framabook.org/>

Et bien évidemment le manuel des commandes ;)