

Programmation et projet encadré - L7TI005

Shell Unix

Crédits supports : Serge Fleury

Yoann Dupont yoann.dupont@sorbonne-nouvelle.fr

Pierre Magistry pierre.magistry@inalco.fr

2025-2026

Université Sorbonne-Nouvelle
INALCO
Université Paris-Nanterre

Introduction



- Unix ?
- Linux ?
- GNU/Linux ?
- Système d'exploitation
 - kernel
 - shell
 - applications
- logiciels et licences

Historique

- Unix aux Bell Labs, AT&T et lois anti-trust
- projet GNU
- Linux
- Libre et Open Source

tissu associatif et "communs"

- April
- Parinux (samedi du libre)
- Debian
- Framasoft
- ...

ESR, The Art of Unix Programming – <http://www.catb.org/~esr/writings/taoup/html/ch01s06.html>

Existence et importance du système de fichiers

Tout est fichier

- dans les années 60, ce n'était pas évident,
- même les programmes, les périphériques, le réseau, ...
- simplicité → on ne pense qu'en termes de fichiers et dossiers,
- d'où l'importance de bien les organiser !

ESR, The Art of Unix Programming – <http://www.catb.org/~esr/writings/taoup/html/ch01s06.html>

Existence et importance du système de fichiers

Tout est fichier

- dans les années 60, ce n'était pas évident,
- même les programmes, les périphériques, le réseau, ...
- simplicité → on ne pense qu'en termes de fichiers et dossiers,
- d'où l'importance de bien les organiser !

Un ensemble de commandes pour manipuler les fichiers et leur contenu.

(depuis **la ligne de commandes**)

- Chaque commande doit faire **une** chose et la faire **bien** (K.I.S.S.)
- Chaque commande est pensée pour pouvoir interagir avec les autres.
- On fait des choses complexes en combinant des commandes simples.
(*pipelines* ou *scripts*)

Le Système de fichier



à définir

- fichier
- dossier (ou répertoire)
- dossier «parent»
- arborescence
- racine
- dossier personnel
- dossier courant ou «de travail» (*working directory*)
- chemin absolu
- chemin relatif
- caractères de remplacement (jokers ou *wildcards*),

Points clefs

- les **fichiers** sont dans un **dossier**
- pouvant eux même être dans un **dossier** « parent »
- ce qui forme une **arborescence**

L'ensemble des données sur la machine sont regroupées en une seule **arborescence**, il est intéressant de pouvoir identifier les fichiers et les dossiers par un **chemin** dans l'arbre.

/ désigne la **racine** de l'arbre

~/ désigne le dossier personnel de l'utilisateur ("*HOME*")

./ désigne le dossier courant (*working directory*)

../ désigne le dossier parent

un **chemin** est formé par une suite de noms de dossiers séparés par des /, pouvant se finir par le nom d'un fichier.

un chemin absolu indique la position d'un fichier en partant de la **racine**.

ex: `/home/pierre/PPE1`

un chemin relatif indique la position d'un fichier en partant du dossier courant.

ex: `../../dev/input/mouse3`

→ habituez vous à toujours être capable de donner le chemin relatif et absolu vers les fichiers que vous manipulez.

Le système de fichier – Jokers ! (*wildcards*)

ou « caractères de remplacement »

Dans un chemin, caractères ? et * ont un comportement spécial.

- ? peut remplacer n'importe quel caractère (unique)

- * peut remplacer n'importe quelle suite de caractères

→ le chemin peut alors désigner plusieurs fichiers !

À tester avec la commande `ls`.

à retenir

- fichier
- dossier (ou répertoire)
- dossier «parent»
- arborescence
- racine
- dossier personnel
- dossier courant ou «de travail» (*working directory*)
- chemin absolu
- chemin relatif
- caractères de remplacement (jokers ou *wildcards*),

Confirmez que vous maîtrisez ces notions dans votre journal, ou exprimez y vos doutes. Et posez des questions la semaine prochaine !

Les Commandes



à découvrir

- commande
- options
- arguments
- page de man(uel)
- et moultres commandes à découvrir

Les commandes sont des fichiers comme les autres,

- qui ont la propriété d'être exécutables
- qui sont placés par convention dans un dossier où le système sait les trouver (typiquement `/usr/bin/`)

La syntaxe d'une commande

nom [-options...] [arguments...]

- les **options** peuvent avoir une forme courte (-o avec un seul tiret)
- ou une forme longue (--option avec deux tirets).
- Les **arguments** sont typiquement des chemins vers des fichiers (mais pas toujours).
- La première action de l'interpréteur de commande est de *tokeniser* la ligne pour découper le nom de la commande, les options et les arguments.

→ habituez vous à anticiper comment les lignes de commandes que vous saisissez vont être découper et à repérer nom, options et arguments, comme l'interpréteur

Quelques commandes à connaître

`cd, ls, pwd, cat, less, wc, echo, head, tail, mkdir, cp, mv, rm, rmdir, file...`

Et surtout: `man`

La plupart des commandes ont aussi une option `--help` qui permet d'obtenir une description concise du fonctionnement et des options de la commande

pwd *print working directory*

ls *list* le contenu d'un dossier

cd *change directory*

cp copier

mv *move* déplacer

rm *remove* supprimer

mkdir *make directory* créer un dossier

touch crée un fichier (effet de bord bien pratique)

zip compresser une archive zip

unzip décompresser une archive zip

tar manipuler les archives tar

file donne des informations sur le type de fichier

cat lit le contenu d'un ou plusieurs fichiers

head lit le début d'un fichier

tail lit la fin d'un fichier

cut sélectionne une ou plusieurs colonnes dans un fichier tabulé

less lecteur (interactif)

à retenir

- commande
- options
- arguments
- page de man(uel)
- et moultres commandes à découvrir

N'hésitez pas à noter les commandes et les options les plus utiles dans votre journal (les pages man sont parfois très longues !)

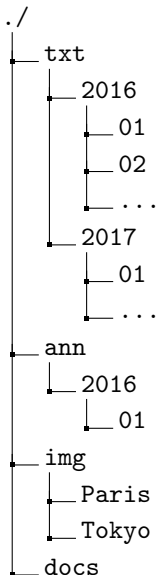
Exercice

L'adresse d'une archive zip a été ajoutée sur le tableur contenant la liste des inscrits au cours (un fichier par personne, en fin de chaque ligne). copier cette adresse qui sera à saisir sur icampus.

Puis dans un terminal:

- créer un dossier "Exercice1" en sous-dossier de votre répertoire personnel avec la **commande** `mkdir`
- rendez-vous ce dossier (par exemple avec la **commande** `cd ~/Exercice1`)
- Télécharger l'archive `fichiers.zip` avec la **commande** `wget` en lui donnant l'url du zip en **argument**.
- utiliser la commande `unzip` pour décompresser l'archive. (consultez le **manuel** si besoin).
- créer une arborescence pour classer les documents
 - par type de fichier (txt, ann, img, docs)
 - puis par date pour les txt et ann
 - puis par lieux pour les photos.

Exercice



Consignes pour le rendu

- L'arborescence à obtenir peut être schématiser comme ci-contre.
- créer une nouvelle archive contenant les fichiers triés
- obtenez l'historique des commandes avec la commande **history** et copier son contenu (à partir du DEBUT de l'exercice) dans un fichier texte
- le nouveau zip et le fichier texte d'historique sont à déposer sur iCampus avant lundi 30 à 23h59.