hestim
Engineering & Business School
Reconnue par l'État

Niveau & Filière : 1ACP

Matière

Introduction pratique aux commandes linux et aux scripts Shell

Animée par : M. Arsène COULIBALY

OBJECTIF DU COURS

Objectif : Maitriser les commandes linux de bases ainsi que les scripts Shell

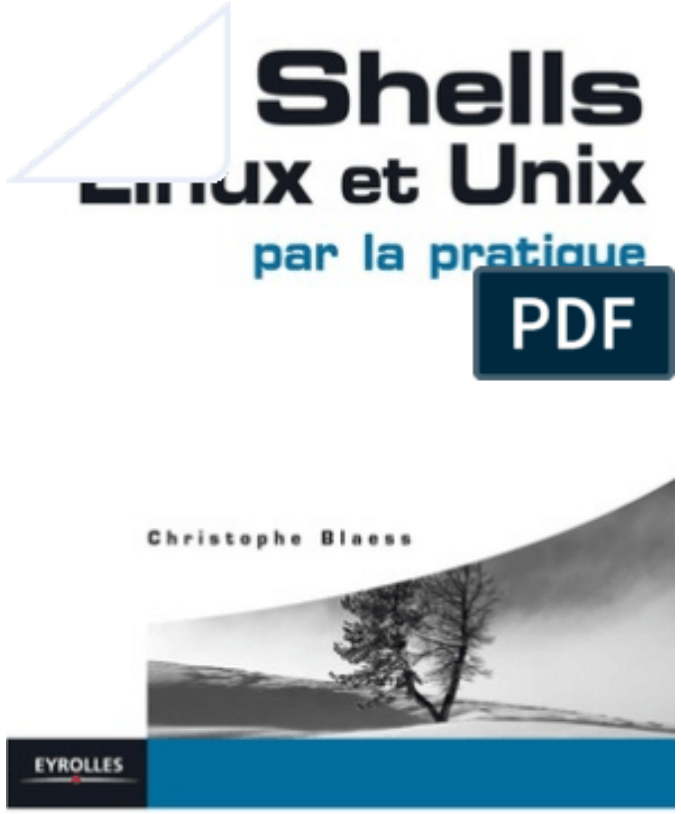
Répartition du volume horaire : 14 h coursera

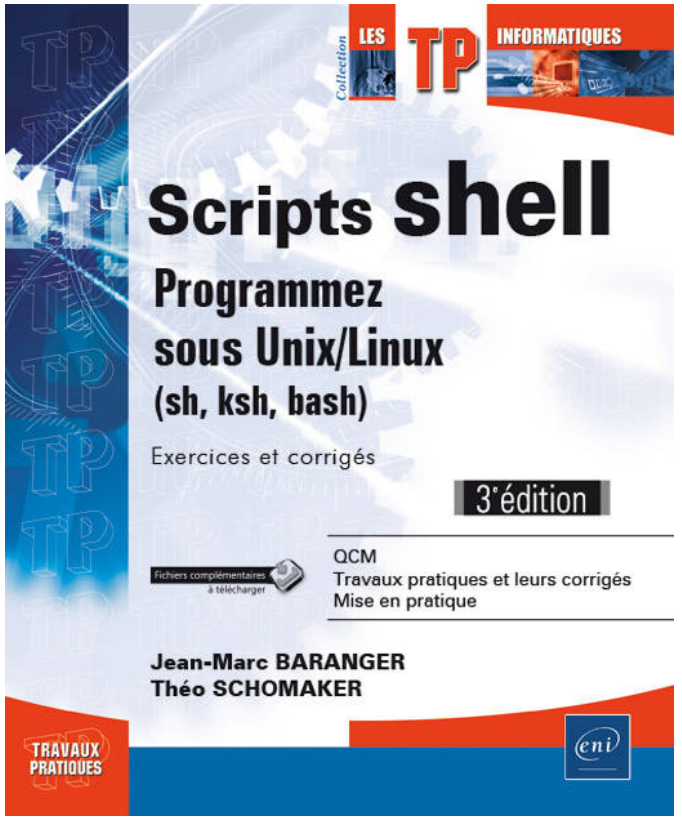
Mode d'évaluation :

Examen Ecrit 70%

Certification 30%

Références





https://openclassrooms.com/fr/courses/43538-reprenez-le-controle-a-laide-de-linux

PLAN DETAILLE DU COURS

Initiation à Coursera

I. Introduction à linux

1. Généralités sur les systèmes d'exploitations
2. Histoire de GNU linux
3. Architecture du système Linux et notion de distribution

II. Introduction aux commandes linux

III. Introduction aux scripts Shell

IV. Projet final et examen final

Initiation à coursera

coursera

Free Online Courses From Top Universities

- Coursera est une plateforme numérique proposant des courts en ligne.
- Le contenu est alimenté en grande partie par des universités à travers le monde mais aussi par des entreprises comme IBM ou google
- On y trouve des cours gratuits et des cours payants
- Pour suivre les cours sur cette plateforme il vous suffit de créer un compte coursera sur la plateforme.

Cours sponsorisé par

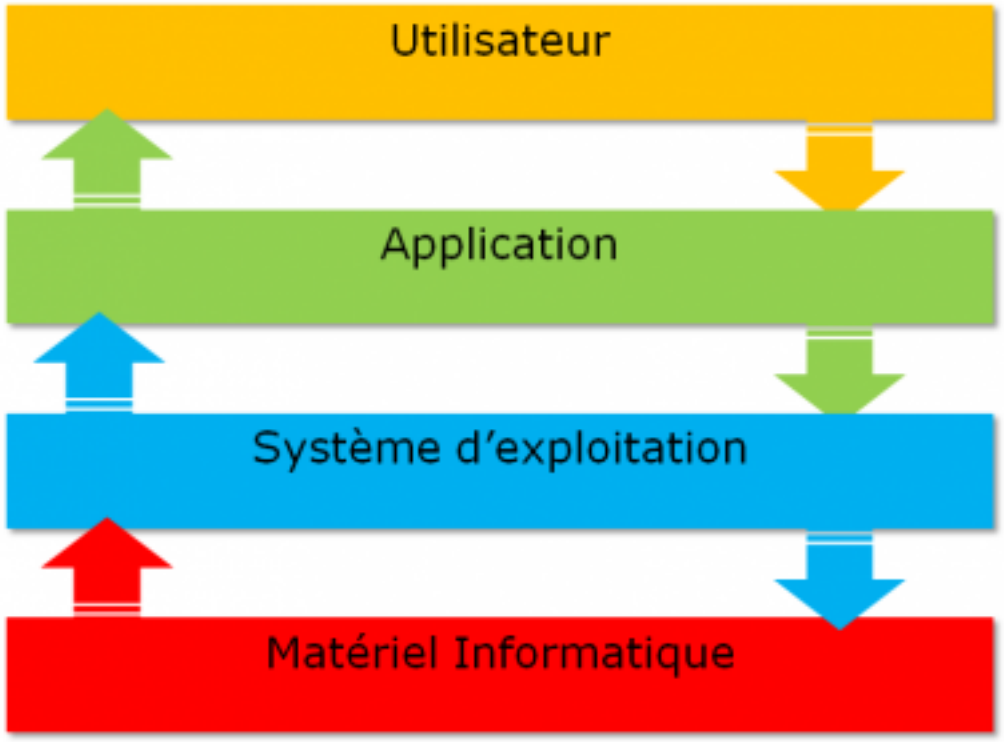
Introduction à linux

1. Généralités sur les OS

Le système d'exploitation (noté SE ou OS, abréviation du terme anglais Operating System), est chargé d'assurer la liaison entre les ressources matérielles, l'utilisateur et les applications.

Deux taches :

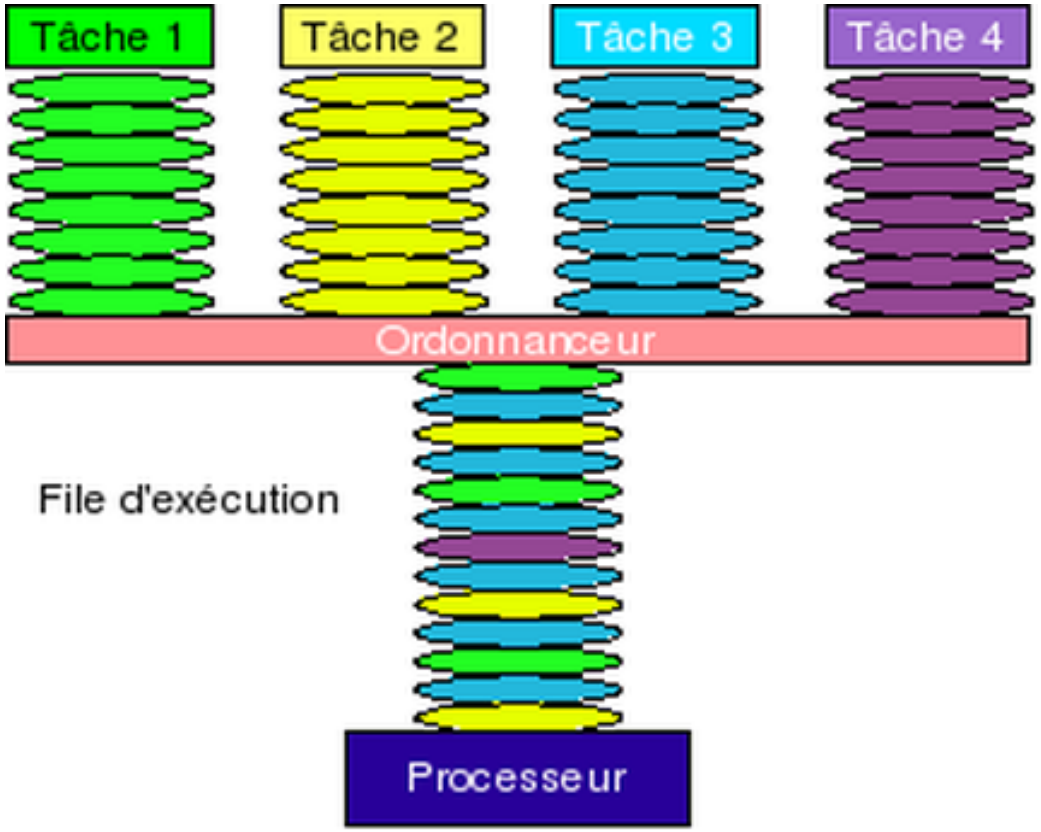
1. Fournir à l'utilisateur une machine étendue ou virtuelle, plus simple à programmer
2. Gestion des ressources. Deux dimensions du partage(multiplexage) :
   - Temps
   - Espace



Introduction à linux

1. Généralités sur les OS

L'os est un grand ordonnanceur



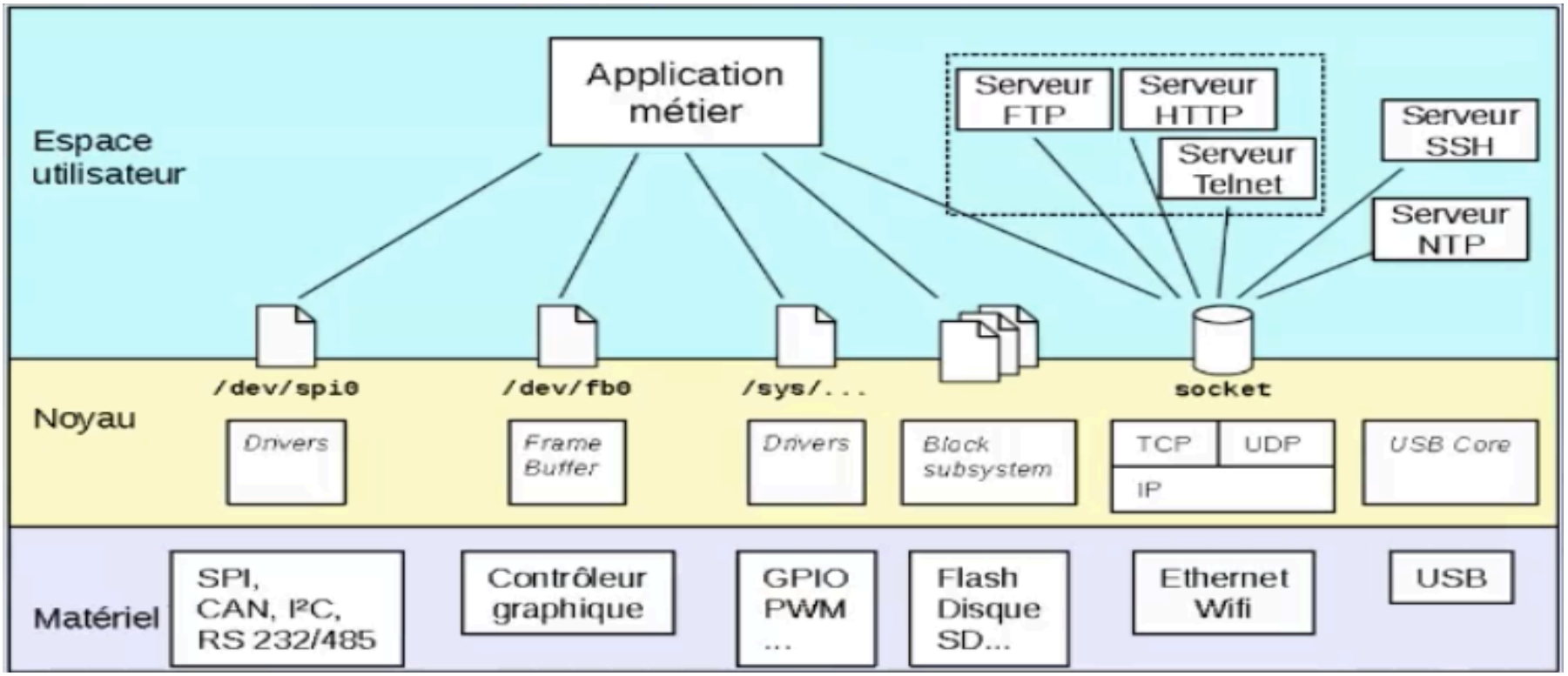
L'OS simule le parallélisme sur une machine séquentielle

Introduction à linux

1. Généralités sur les OS

Abstraction des périphériques

Objectif : Masquer la complexité matériel/ Facilité l'accès au matériel



Linux est un OS à accès fichier

2. Histoire de GNU linux

Logiciel Libre, concepts et licences

La licence libre est document autorisant l'utilisation d'un logiciel sous certaines conditions : Elle constitue un contrat entre l'éditeur et l'utilisateur. Une licence libre ajoute trois libertés fondamentales :

- Utiliser le logiciel(même commercialement)
- Etudier et modifier le code source
- Distribuer la version modifiée.

Un logiciel libre ne doit pas être confondu avec le **freeware**, dont le code source n'est pas disponible et la licence pas forcément compatible avec le modèle libre

Logiciel Libre , concepts et licences

2. Histoire de GNU linux

Définition d'un Logiciel Libre

- Un programme est considéré comme libre lorsque sa licence offre à tous les utilisateurs les quatre points de liberté suivantes :
  - **La liberté d'exécuter le logiciel pour n'importe quel but**
  - **La liberté d'étudier le logiciel et de le modifier**
  - **La liberté de redistribuer des copies**
  - **La liberté de distribuer des copies de versions modifiées**
- Ces libertés sont accordées pour une utilisation commerciale et non commerciale
- Elles impliquent la disponibilité du code source, le logiciel peut être modifié et distribué aux clients

2. Histoire de GNU linux

Qu'est-ce que l'Open Source ?

L'Open Source repose sur les principes du logiciel libre, mais est né d'une scission avec la FSF (Free Software Foundation) vers 1998 et la création de l'OSI (Open Source Initiative) par **Eric Raymond.**

2. Histoire de GNU linux

Naissance du Logiciel Libre

- **1983**, Richard Stallman, Projet GNU et le concept de logiciel . Début du développement de gcc, gdb, Glibc et d'autres outils importants.

- **1991**, Linux Torvalds, Linux kernel Project. De plus avec les logiciels de GNU et de nombreux autres composants open-source.
  -> Un système d'exploitation entièrement libre, GNU/Linux

2. Histoire de GNU linux

Naissance du Logiciel Libre

- **1991**, Linux Torvalds, Linux kernel Project. De plus avec les logiciels de GNU et de nombreux autres composants open-source.
  -> Un système d'exploitation entièrement libre, GNU/Linux

```
Salut à tous les utilisateurs de Minix,
Je suis en train de réaliser un système d'exploitation (gratuit), (c'est juste un
passe-temps, il ne sera pas important et professionnel comme le Gnu) pour les
clones d'AT 386 (ou 486). Il mijote depuis le mois d'avril, et commence à être au
point. J'aimerais des remarques sur ce que les gens aiment ou non dans Minix, car
mon système lui ressemble un peu (même organisation du système de fichiers, pour
des raisons pratiques, entre autres).
J'y ai porté Bash (1.08) et Gcc (1.40), et tout semble fonctionner. Je devrais
donc disposer de quelque chose d'utilisable dans les mois à venir, et j'aimerais
connaître les fonctionnalités que la plupart d'entre vous aimeraient. Toutes les
suggestions sont les bienvenues, mais je ne promets pas de toutes les
implémenter ;-)
Linus (torvalds@kruuna.helsinki.fi)
PS : Oui, il est indépendant de tout code provenant de Minix, et il dispose d'un
système de fichiers multi-thread. Il n'est PAS portable (utilisation de la
commutation de tâches 386, etc.) et ne supportera probablement jamais rien
d'autres que les disques durs AT car ce sont les seuls dont je dispose :-(.
```

2. Histoire de GNU linux

Naissance du Logiciel Libre

- **1995**, Linux devient de plus en plus populaire sur les systèmes serveur
- **2000**, Linux devient de plus en plus populaire sur les systèmes embarqués
- **2008**, Linux devient de plus en plus populaire sur les appareils mobiles
- **2010**, Linux devient de plus en plus populaire sur les téléphones

Introduction à linux

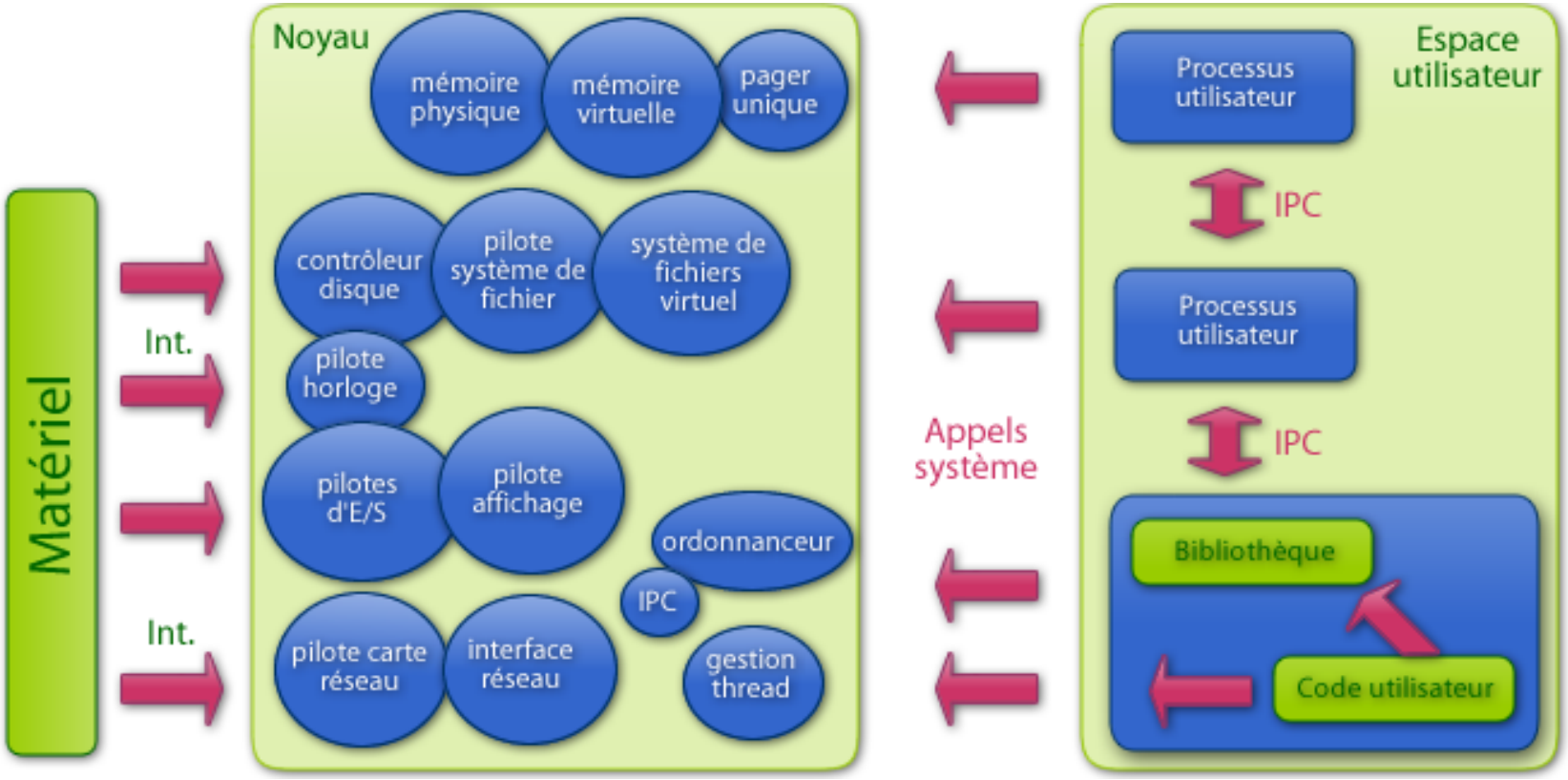
2. Histoire de GNU linux

Qu'est-ce que Linux

- **Linux ne désigne que le noyau**
- Linux est associé aux outils GNU d'où le nom GNU/Linux
- Systèmes avec les outils GNU mais un noyau différent : GNU/Hurd, Solaris etc
- Systèmes Linux sans GNU : Android

Introduction à linux

3. Architecture de linux et notion de distribution

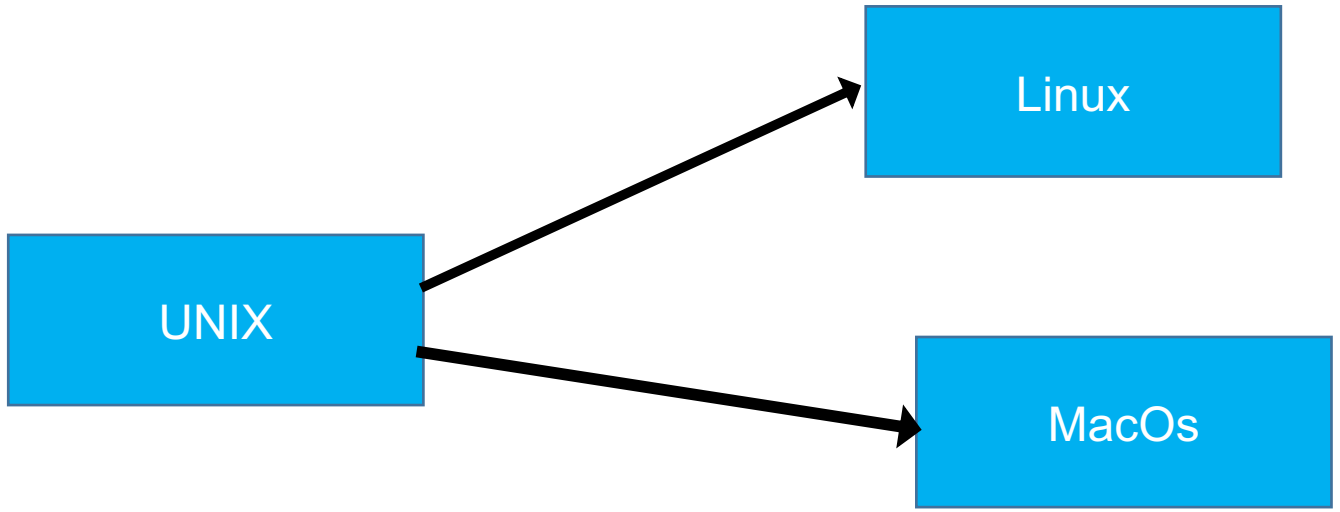


Linux est un système d'exploitation:

- **Monolithique**
- **Modulaire**

Introduction à linux

3. Architecture de linux et notion de distribution



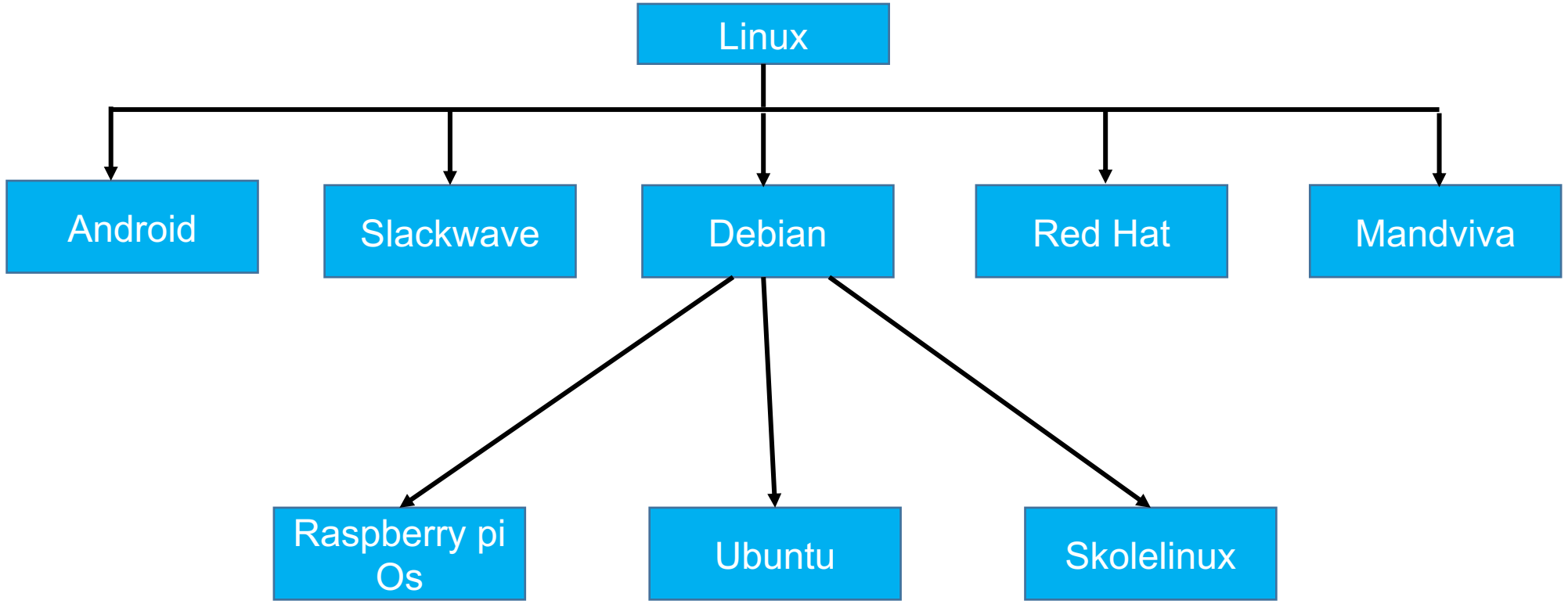
Linux et MacOs sont tous les deux Basés sur Unix.

Par Windows qui est un Os propriétaire est basé sur MS-DOS

Introduction à linux

3. Architecture de linux et notion de distribution

Une distribution de Linux est un Os qui est basé sur le noyau Linux mais dont l'espace utilisateur a été personnalisé. Toutes les distributions de Linux ont donc le même noyau Linux.



Introduction aux commandes Linux

1. Entrer une commande

Le terminal ou encore la console est l'interface nous permettant de saisir une commande. Il existe sur linux au total 6 terminaux nommés ttyi avec i allant de 1 à 6.

```
Last login: Wed Sep 27 16:56:08 on ttys000
You have new mail.
[3] 36595
(base) mac@korota ~ %
```

La console affiche une invite de commandes au début de la ligne

Introduction aux commandes Linux

1. Entrer une commande

Une commande est un programme(un exécutable) qui demande au noyau linux d'exécuter une tache spécifique.
Exemple : La commande ls(List directory) permet de lister le contenu d'un répertoire

```
(base) mac@korota Cours_linux_2AIIIA % ls
Cours    Examens TD      TP
```

Introduction aux commandes Linux

2. Structure des dossiers sur linux

Sur linux tout est fichiers. Toute la partie matériel est virtualisé sous forme de fichiers. On distingue deux grands types de fichiers :

- Les fichiers classiques : fichiers texte, les sons, les programmes
- Les fichiers spéciaux : Ce sont des fichiers qui virtualise un matériel spécifique de l'ordinateur. Par exemple votre clé usb , votre lecteur CD sont des fichiers pour linux.

Introduction aux commandes Linux

2. Structure des dossiers sur linux

2.1. La racine

Dans un système de fichiers, il y a toujours ce qu'on appelle une racine, c'est-à-dire un « **gros dossier de base qui contient tous les autres dossiers et fichiers.**

Sur Windows on a deux racines :

- **C:\ : pour le disque dur**
- **D:\ : Pour le lecteur CD**

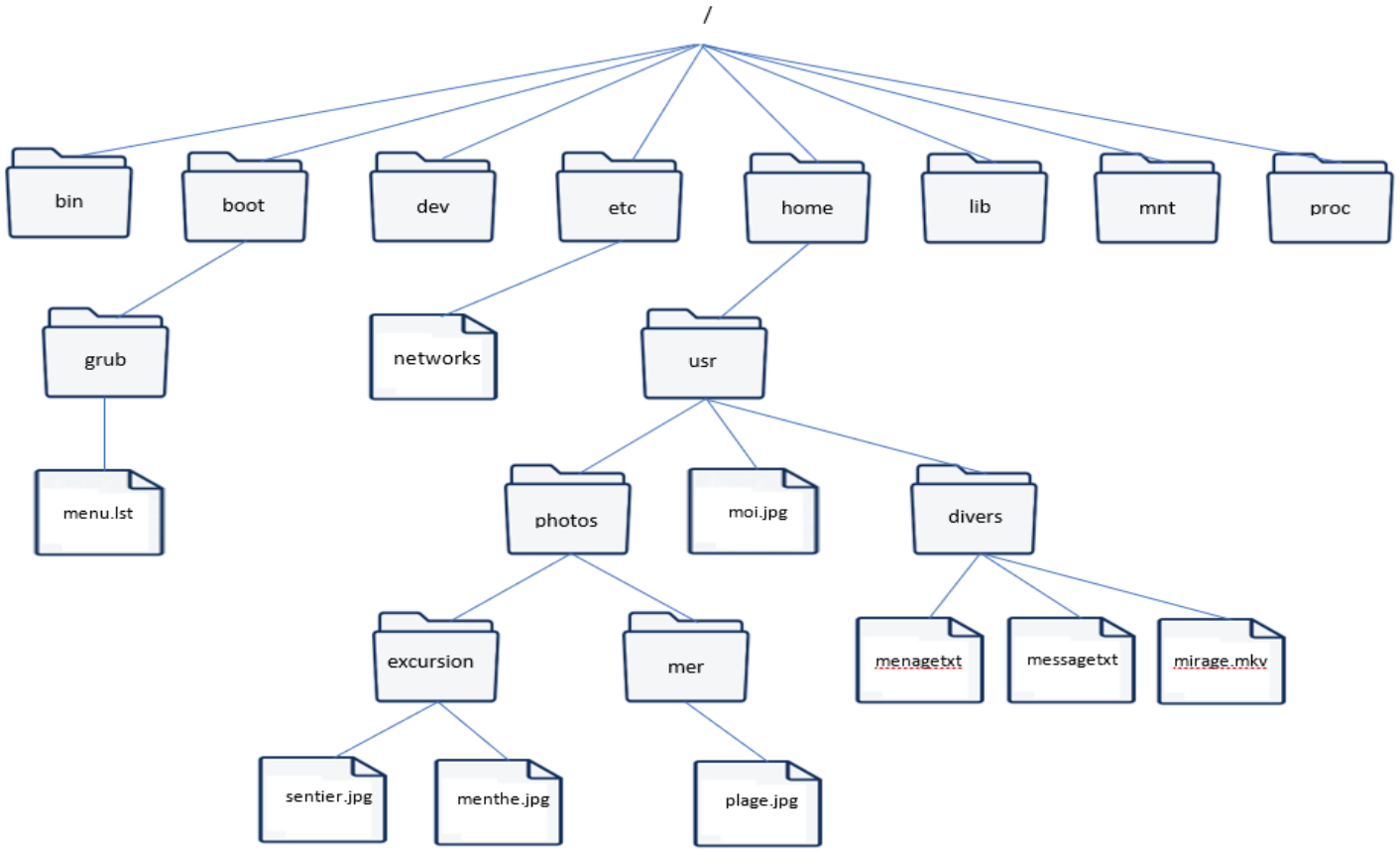
Sur Linux on a par contre une seule racine « / ». Il n'y a pas de dossier de plus haut niveau que la racine. Voici le contenu de la racine :

```
(base) mac@korota ~ % ls /
Applications    Volumes         etc             sbin
Library         bin             home            tmp
System          cores           opt             usr
Users           dev             private         var
(base) mac@korota ~ %
```

Introduction aux commandes Linux

2. Structure des dossiers sur linux

2.2. Arborescence



Introduction aux commandes Linux

2. Structure des dossiers sur linux

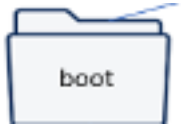
2.2. Arborescence

contient des programmes (exécutables) susceptibles d'être utilisés par tous les utilisateurs de la machine.

répertoires personnels des utilisateurs. c'est dans ce dossier que vous placerez vos fichiers personnels. Chaque utilisateur possède son dossier personnel

fichiers de configuration

contient des informations système



fichiers permettant le démarrage de Linux.

fichiers contenant les périphériques

dossier contenant les bibliothèques partagées (généralement des fichiers .so) utilisées par les programmes.

Introduction aux commandes Linux

2. Quelques commandes usuelles

2.3. Arborescence

pwd(Print Working Directory) & which : Où suis-je ?

pwd : Permet de savoir dans quel répertoire nous nous trouvons

```
(base) mac@korota /home % pwd
/home
(base) mac@korota /home %
```

which : Permet de trouver l'emplacement d'une commande

```
(base) mac@korota /home % which ls
/bin/ls
(base) mac@korota /home %
```

Introduction aux commandes Linux

2. Quelques commandes usuelles

2.3. Arborescence

ls(list) : Permet de lister les fichiers et les répertoires contenus dans un dossier

cd(Change Directory) : changer de dossier

```
(base) mac@korota ~ % pwd
/Users/mac
(base) mac@korota ~ % cd Desktop
(base) mac@korota Desktop % pwd
/Users/mac/Desktop
(base) mac@korota Desktop %
```

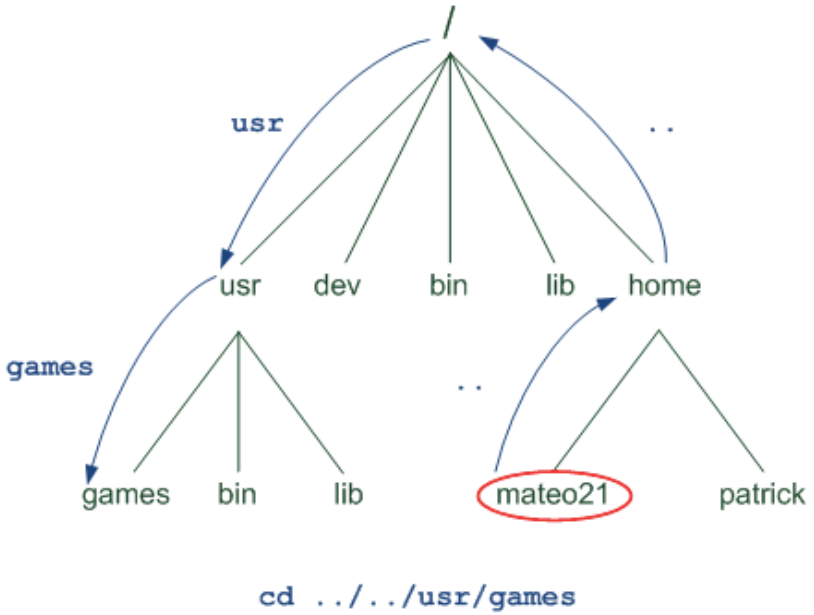
Introduction aux commandes Linux

2. Quelques commandes usuelles

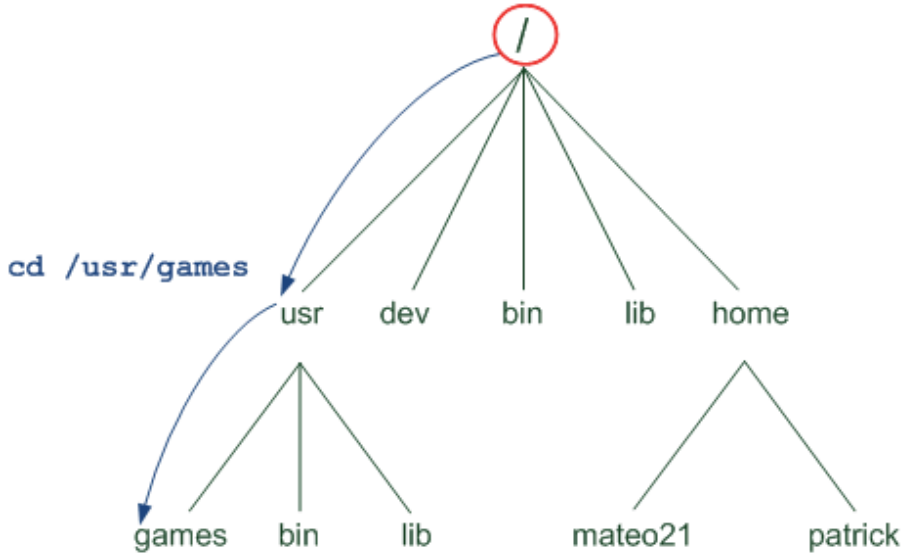
2.3. Arborescence

Chemin relatif vs chemin absolu

Un chemin relatif est un chemin qui dépend du dossier dans lequel vous vous trouvez.



les chemins absolus fonctionnent quel que soit le dossier dans lequel on se trouve. Un chemin absolu est facile à reconnaître : il commence toujours par la racine (/).



Introduction aux commandes Linux

2. Quelques commandes usuelles

2.4. Manipulation des fichiers

touch & mkdir : créer des fichiers et dossiers

cp & mv : copier et déplacer un fichier

rm : supprimer des fichiers et dossiers

head & tail : afficher le début et la fin d'un fichier

cat & less : afficher un fichier

Introduction aux commandes Linux

2. Quelques commandes usuelles

2.5. Les utilisateurs et les droits

Introduction aux commandes Linux

2. Quelques commandes usuelles

2.6. Recherche de fichiers

Introduction aux commandes Linux

2. Quelques commandes usuelles

2.7. Les flux et les rédirections

Introduction aux commandes Linux

2. Quelques commandes usuelles

2.8. Exécuter un programme à une heure différée

Introduction aux scripts Shell

1. Introduction sur le shell

Le shell est le programme qui interprète les différentes commandes(cp, ls …) tapés en ligne de commande . Il existe plusieurs types deShell :

- sh : Bourne Shell. L'ancêtre de tous les shells.
- bash : Bourne Again Shell. Une amélioration du Bourne Shell, disponible par défaut sous Linux et Mac OS X.
- ksh : Korn Shell. Un shell puissant assez présent sur les Unix propriétaires, mais aussi disponible en version libre, compatible avec bash.
- csh : C Shell. Un shell utilisant une syntaxe proche du langage C.
- tcsh : Tenex C Shell. Amélioration du C Shell.
- zsh : Z Shell. Shell assez récent reprenant les meilleures idées de bash, ksh et tcsh.

Introduction aux scripts Shell

2. Premier script Shell

Un script Shell est tout simplement un fichier sur lequel on retrouve un ensemble de commande linux permettant d'exécuter une tache donnée.

Nous allons créer un script bash qui affiche le nom de l'utilisateur, puis le dossier dans lequel nous nous trouvons et ensuite liste le contenu de ce dossier.

```
 GNU nano 2.0.6                File: script1.sh                      Modified

#!/bin/bash

#Nom d'utilisateur
whoami

#Le dossier courant
pwd

#Contenu du dossier
ls
```

Introduction aux scripts Shell

2. Premier script Shell

- Un script Shell porte l'extension .sh.
- Toutes les lignes débutant par un # sont des commentaires sauf la première ligne
- La première ligne appelée « **Schebang** » permet de spécier le type de shell que l'on souhaite utiliser. Dans notre cas nous utilisons le bash.

```
#!/bin/bash
```

- Les autres lignes sont une suite de commandes

Introduction aux scripts Shell

2. Premier script Shell

Exécution d'un script

1. Commencer par donner le droit d'exécution au script avec chmod.

```
(base) mac@korota Desktop % ls -l script1.sh
-rw-r--r--  1 mac  staff  91 Oct  2 03:51 script1.sh
(base) mac@korota Desktop % chmod +x script1.sh
(base) mac@korota Desktop % ls -l script1.sh
-rwxr-xr-x  1 mac  staff  91 Oct  2 03:51 script1.sh
(base) mac@korota Desktop %
```

2. Exécution du script :

```
(base) mac@korota Desktop % ./script1.sh
mac
/Users/mac/Desktop
-factorielle
01-Set_Variables.jpg
02-Display_Values.png
03-CurrentTS.png
04-Set_Value.png
```

Introduction aux scripts Shell

3. Afficher et manipuler des variables

Une variable possède un nom et une valeur :

```
#Déclarations de variables
nom="Nabaloum"
prenom="Abari"
age=23
```

echo : permet d'afficher le contenu d'une variable
Syntaxe : > echo $nomVariable

read : demander une saisie
syntaxe : > read nomVariable

Introduction aux scripts Shell

3. Afficher et manipuler des variables

Effectuer des opérations mathématiques

Pour effectuer des opérations mathématiques on utilise en général deux méthodes:

1. le keyword « **let** »
2. **$((opérations à effectuer))**

```
#opérations mathémathiques
let "a=4"
let "b = 5"
let "c = a + b"

$a=4
$b=5
$c=$((a+b+2*4+1))
```

Introduction aux scripts Shell

3. Afficher et manipuler des variables

Variables d'environnement

Les variables d'environnement sont des variables que l'on peut utiliser dans n'importe quel programme. On parle aussi parfois de **variables globales**. La commande « **env** » permet d'afficher toutes les variables d'environnement.

```
(base) mac@korota Desktop % env
TMPDIR=/var/folders/5_/vrqk_jsj0197tfl3glh3rljc0000gn/T/
XPC_FLAGS=0x0
LaunchInstanceID=28B0FA98-4BE9-47C2-9347-8EDD27B02D1C
TERM=xterm-256color
SSH_AUTH_SOCK=/private/tmp/com.apple.launchd.AswAPPwjW2/Listeners
SECURITYSESSIONID=186a7
XPC_SERVICE_NAME=0
TERM_PROGRAM=Apple_Terminal
TERM_PROGRAM_VERSION=433
TERM_SESSION_ID=F1516297-5729-475A-A388-1C1779E9AAF8
SHELL=/bin/zsh
HOME=/Users/mac
LOGNAME=mac
USER=mac
```

Introduction aux scripts Shell

3. Afficher et manipuler des variables

Variables d'environnement

Les variables d'environnement sont des variables que l'on peut utiliser dans n'importe quel programme. On parle aussi parfois de **variables globales**. La commande « **env** » permet d'afficher toutes les variables d'environnement.

```
(base) mac@korota Desktop % env
TMPDIR=/var/folders/5_/vrqk_jsj0197tfl3glh3rljc0000gn/T/
XPC_FLAGS=0x0
LaunchInstanceID=28B0FA98-4BE9-47C2-9347-8EDD27B02D1C
TERM=xterm-256color
SSH_AUTH_SOCK=/private/tmp/com.apple.launchd.AswAPPwjW2/Listeners
SECURITYSESSIONID=186a7
XPC_SERVICE_NAME=0
TERM_PROGRAM=Apple_Terminal
TERM_PROGRAM_VERSION=433
TERM_SESSION_ID=F1516297-5729-475A-A388-1C1779E9AAF8
SHELL=/bin/zsh
HOME=/Users/mac
LOGNAME=mac
USER=mac
```

Introduction aux scripts Shell

3. Afficher et manipuler des variables

Variables paramètres

Ce sont les paramètres passés au script lors de son exécution :

- `$#` : contient le nombre de paramètres ;
- `$0` : contient le nom du script exécuté (ici `./variables.sh`) ;
- `$1` : contient le premier paramètre ;
- `$2` : contient le second paramètre ;
- … ;
- **$9** : contient le 9e paramètre.

Introduction aux scripts Shell

3. Les conditions

Condition simple

```
if [ test ]
then
        echo "C'est vrai"
fi
```

```
if [ test ]
then
        echo "C'est vrai"
else
        echo "C'est faux"
fi
```

```
if [ test ]
then
        echo "Le premier test a été vérifié"
elif [ autre_test ]
then
        echo "Le second test a été vérifié"
elif [ encore_autre_test ]
then
        echo "Le troisième test a été vérifié"
else
        echo "Aucun des tests précédents n'a été vérifié"
fi
```

Introduction aux scripts Shell

3. Les conditions

Condition simple

`$num1 -eq $num2`	Vérifie si les nombres sont égaux (**eq***ual*). À ne pas confondre avec le « = » qui, lui, compare deux chaînes de caractères.
`$num1 -ne $num2`	Vérifie si les nombres sont différents (**n***on***e***qual*). Encore une fois, ne confondez pas avec « != » qui est censé être utilisé sur des chaînes de caractères.
`$num1 -lt $num2`	Vérifie si `num1` est inférieur (<) à `num2` (**l***ower***t***han*).
`$num1 -le $num2`	Vérifie si `num1` est inférieur ou égal (<=) à `num2` (**l***owero***e***qual*).
`$num1 -gt $num2`	Vérifie si `num1` est supérieur (>) à `num2` (**g***reater***t***han*).
`$num1 -ge $num2`	Vérifie si `num1` est supérieur ou égal (>=) à `num2` (**g***reatero***e***qual*).