**[Introduction shell Linux](https://moodle.epita.fr/course/view.php?id=2908" \o "Edit section name)**

**1. Introduction**

* **Shell** : Interpréteur de commandes pour interagir avec le système.
* **Script Shell** : Fichier contenant des commandes exécutables.
* **Objectif** : Automatiser des tâches, gagner du temps, personnaliser le système.

**2. Structure d'un script Shell**

* **Shebang** : #!/bin/bash (indique l'interpréteur à utiliser).
* **Exemple simple** :

#!/bin/bash

echo "Hello World !"

**3. Variables**

* **Déclaration** : NOM\_VARIABLE="valeur"
* **Utilisation** : $NOM\_VARIABLE
* **Exemple** :

NOM="Alice"

echo "Bonjour, $NOM"

**4. Boucles**

**Boucle for**

* Répète une action pour chaque élément d'une liste.
* **Syntaxe** :

for i in 1 2 3; do

  echo "Numéro : $i"

done

**Boucle while**

* Répète une action tant qu'une condition est vraie.
* **Syntaxe** :

COUNT=0

while [ $COUNT -lt 3 ]; do

  echo "Compteur : $COUNT"

  COUNT=$((COUNT + 1))

done

Shape

**5. Conditions**

* Utilisez **if** pour exécuter des commandes conditionnelles.
* **Syntaxe** :

if [ CONDITION ]; then

  COMMANDE1

else

  COMMANDE2

fi

**Opérateurs courants**

* **Numériques** :
* -eq : égal à.
* -ne : différent de.
* -gt : supérieur à.
* -lt : inférieur à.

**Exemple** :

if [ "$AGE" -gt 18 ]; then

  echo "Vous êtes majeur."

fi

* **Chaînes** :
* = : égal à.
* != : différent de.
* -z : chaîne vide.
* -n : chaîne non vide.

**Exemple** :

if [ -z "$NOM" ]; then

  echo "Le nom est vide."

fi

* **Fichiers** :
* -e : existe.
* -d : est un dossier.
* -f : est un fichier.

**Exemple** :

if [ -f "/chemin/fichier.txt" ]; then

  echo "Le fichier existe."

fi

Shape

**6. Commandes utiles**

* **grep** : Recherche de motifs dans du texte.

**Exemple** :

grep -i "error" /var/log/syslog

grep est un outil puissant pour filtrer des lignes de texte en fonction de motifs.

* **^** : Ce symbole permet de rechercher des lignes qui **commencent** par un motif spécifique.
* Exemple : grep "^CPU" fichier.txt affiche uniquement les lignes qui commencent par "CPU".
* **$** : Ce symbole permet de rechercher des lignes qui **finissent** par un motif spécifique.
* Exemple : grep "error$" fichier.txt affiche uniquement les lignes qui finissent par "error".
* **-v** : Inverse la recherche pour afficher les lignes qui **ne correspondent pas** au motif.
* Exemple : grep -v "^#" fichier.txt ignore les lignes commençant par #.
* **awk** : Traitement de texte avancé.
* **sed** : Édition de flux de texte.

**7. Partie pratique**

**Exercice 1 : Créer un script simple**

1. Créez un fichier bonjour.sh :

#!/bin/bash

echo "Quel est votre nom ?"

read NOM

echo "Bonjour, $NOM !"

1. Rendez-le exécutable :

chmod +x bonjour.sh

1. Exécutez-le :

./bonjour.sh

**Exercice 2 : Utiliser une boucle for**

1. Créez un fichier liste.sh :

#!/bin/bash

for FICHIER in $(ls); do

  echo "Fichier : $FICHIER"

done

1. Exécutez-le pour lister les fichiers du répertoire courant.

**Exercice 3 : Condition avec if**

1. Créez un fichier verifier.sh :

#!/bin/bash

for FICHIER in $(ls); do

  echo "Fichier : $FICHIER"

done

1. Exécutez-le pour vérifier si test.txt existe.

**Exercice 4 : Recherche avec grep**

1. Créez un fichier rechercher.sh :

#!/bin/bash

echo "Entrez un mot à rechercher :"

read MOT

grep -i "$MOT" /var/log/syslog

1. Exécutez-le pour rechercher un mot dans les logs système.

**Exercice 5 : Gestion des services avec systemctl**

1. Créez un fichier service-chrony.sh :

#!/bin/bash

if systemctl is-active chronyd > /dev/null; then

  echo "Le service chronyd est déjà actif."

else

  sudo systemctl start chronyd

  echo "Le service chronyd a été démarré."

fi

1. Rendez-le exécutable :

chmod +x service-chrony.sh

1. Exécutez-le pour vérifier et démarrer le service chronyd si nécessaire.

**Exercice 6 : Filtrage de sorties avec grep**

1. Créez un fichier filtrage.sh :

#!/bin/bash

# Afficher le nom d'hôte complet

hostname -f

echo "#####"

# Afficher uniquement les lignes de lscpu qui commencent par "CPU"

lscpu | grep "^CPU"

echo "#####"

# Afficher le contenu de /etc/selinux/config en ignorant les lignes vides et les commentaires

grep -v -E '^$|^#' /etc/selinux/config

echo "#####"

# Afficher toutes les entrées "Mounted " dans /var/log/boot.log

grep "Mounted " /var/log/boot.log

echo "#####"

# Vérifier si un utilisateur spécifique existe

UTILISATEUR="root"

if id "$UTILISATEUR" > /dev/null 2>&1; then

  echo "L'utilisateur $UTILISATEUR existe."

  echo "Informations sur l'utilisateur :"

  id "$UTILISATEUR"

else

  echo "L'utilisateur $UTILISATEUR n'existe pas."

fi

echo "#####"

# Afficher la mémoire disponible avec free (en Mo)

echo "Mémoire disponible :"

free -m | grep "Mem:"

1. Rendez-le exécutable :

chmod +x filtrage.sh

1. Exécutez-le pour voir les résultats filtrés.

**8. Conclusion**

* Les scripts Shell permettent d'automatiser des tâches sous Linux.
* En maîtrisant les bases (variables, boucles, conditions, opérateurs), vous pouvez créer des scripts utiles et efficaces.