Introduction à Linux

Introduction:
Fichiers de commandes (script)

variables pré-définiers

- `$# nb paramètres passés au programme shell`
- `$? code de la dernière commande ou programme shell exécuté`
- `$0 le nom du programme shell`
- `$* UNE chaîne de caractère contenant les paramètres passés`
- `$@ une liste des paramètres passés`

Paramètres de système et variables

- Set : pour voir tout ce qui est assigné
  - Set | more …page par page
- Voir un paramètre en particulier :
  - Echo $SHELL
  - Echo $username
- Assignation
  - Var=valeur
- Lire la valeur
  - Echo $Var

Aide sur les commandes

- -- help
- Man
- Info

Commande - -help	if -help ?
Info commande	info if
Man commande	man if

«quote»

- **' ' (apostrophes)**
  le contenu est traité sans interprétation
- **" " (guillemets)**
  les variables contenues sont interprétées
- **\ («backslash»)**
  le caractère suivant ne sera pas interprété comme un caractère spécial
- **` ` (accents graves «backtick»)**
  la commande insérée sera interprétée
  par exemple `var=`wc -l test.tx`
  affectera à `var` le nombre de lignes dans le fichier test.txt

wait

- **waitlock**
- `while test -r lockfile ; do sleep 5 ; done`
- `#` ^ *caractère "new line"*

- Ce programme attend qu'un fichier soit détruit.

Création d'un fichier

CAT : Concaténation

Concatène ce qui est entré au clavier dans un fichier :

Cat > Test.sh

Concatène le fichier à la console :

Cat Test.sh

Arrière plan

```
until test -r stopfile
do sleep 2
    echo Hello
done &
# & pour exécution en arrière-plan
```

- Que fait ce programme ?
- Créer un fichier nommé `stopfile` ; que se passe-t-il ?

for

fruits

```
for fruits in apples pears oranges mangos
do    #do exécuté pour chaque élément dans liste
      echo $fruits are fruits
done
```

case

breeds

```
for breed       # exécute le do pour chaque paramètre
do
        case $breed in # ;; fin des commandes exécutées pour ce cas
                  arabian|palomino|clydesdale) echo $breed is a horse ;;
                  jersey|guernsey|holstein) echo $breed is a cow ;;
                  husky|shepherd|setter|labrador) echo $breed is a dog ;;
                  siamese|persian|angora|) echo $breed is a cat ;;
                  *) echo $breed is not in our catalog ;;
        esac
done
```

Exécuter :

```
breeds husky holstein terrier
```

Traitement paramètres

lsdir6

```
# liste les sous-répertoires des répertoires donnés en paramètres
# usage : lsdir6 [rép1/ rép2/ ... répn/]
#
if test $# = 0          # $# variable contenant le nombre de paramètres
    then
       param=.
    else                # $@ variable contenant les paramètres sous
       param="$@"       # forme d'une liste, équivalent $1 $2 ... $n
fi
for i in $param
do
    for j in $i/*       # $i/* la liste des éléments dans le iᵉ répertoire
    do
        if test -d $j
            then
                echo $j
        fi
    done
done
```

Utilisation de fonctions (1/2)

```
fna
# utilisation de fonctions
TMP=/tmp
PIDLOG=${TMP}/pidlog$$
DATEFILE==${TMP}/date$$
ERRLOG=${TMP}/demolog
#
# fonction shell pour nettoyer et quitter
errexit() {
     echo $1
     date >> $ERRLOG
     echo $1 >> $ERRLOG
     rm -f $PIDLOG $DATEFILE
     exit
}
#
```

Utilisation de fonctions (2/2)

```
# fonction shell pour lire une réponse y/n et fixer code retour 0 ou 1

ok() {
        while true
        do
                read ans
                case $ans in
                        [yY]*)    return 0 ;;
                        [nN]*)    return 1 ;;
                        *)    echo Please answer y or n ;;
                esac
        done
}
# début traitement de la commande
echo $$ > $PIDLOG          # $$ variable contenant le PID du shell
date > $DATEFILE
#
# test error exit
echo Test errexit function [y/n]?
ok && errexit "Testing the errexit function"
#  ^^ pipeline exécuté si vrai (|| utilsé si faux)
echo Normal Termination
echo Please remove $PIDLOG and $DATEFILE
```

- Tester en exécutant fna et en répondant successivement ***y***, ***maybe*** et ***n***.

Permissions

-rwxrwxrwx Owner Group Fichier

- r : Read – Lecture
- w : Write – Écriture
- x : eXecute – exécuter

- Pour changer les permissions
  - CHMOD

    Chmod 700 fich.sh 700 = Masque binaire : 7= 4+2+1

    Chmod u +x fich.sh