Programmation BASH

M1 – Architecture Système Réseau

Plan

- Structure du fichier
- Les arguments
- Les variables
- Les conditions
- Les boucles
- Le switch case
- Les intéractions
- Les commandes

Structure du fichier

Un script est un programme rapide et pratique.

Organisation d’un fichier script

```
#!/bin/bash

# ceci est un commentaire

Instruction
Boucle
        Instruction
        Instruction
        Condition
                Instruction
                Instruction
Condition
            Instruction
            Instruction
Instruction
Condition
            Boucle
                Instruction
```

Structure d’un fichier

Pour nommer le fichier

- *nom_script.sh*
- *nom_script*

Pour l ‘exécuter

Sans affecter les droits : **bash**

nom_script

Après avoir affecté les droits d’exécution : *./nom_script*

Un script de test n’a pas besoin d’avoir les droits d’exécution en phase de conception. Ceci évite les problèmes de sécurité liés aux fichiers laissés à l’abandon avec des droits élevés.

Les arguments

Lorsque l’on crée un script, il est possible de récupérer les arguments.

Le nombre de paramètres est donné par ***$#***
les paramètres sont appelés individuellement par leur numéro ***$1, $2 …***
sachant que ***$0*** est le nom de la commande
il est possible de récupérer la chaine complète avec ***$****

Les variables

Syntaxe des variables

Déclaration d'une variable

test= « test »

Affichage d'une variable

echo $test

Les variables

Les tableaux

Déclaration d'un tableau

```
tab_donnee=(« un » « deux » « trois »)
```

Manipuler le tableau

affichage d'une donnée **${tab_donnee[1]}**

afficher tous les éléments du tableau **${tab_donnee[*]}**

afficher le nombre d'éléments **${#tab_donnee[*]}**

- *Ref : http://debian-facile.org/doc:programmation:shells:tableaux*

Les conditions

Syntaxe

```
if [ $variable1 -eq $variable2 ] ;
then
        Instruction 1
         Instruction 2
          ...
elif                                                    # sinon si
         Instruction 1
        Instruction 2
         ...
else                                                    # sinon
      Instruction 1
      Instruction 2
       ...
fi                                                      # fin du if
obligatoire
```

Les conditions

Il peut également utiliser des opérateurs logiques tel que **&&** pour ET et **||** pour OU :

if [*condition 1*] **&&** [*condition 2*]

if [*condition 1*] **||** [*condition 2*]

Les conditions

les conditions possibles :

- -eq = égal (equal)
- -ne = non-égal (not equal)
- -lt = plus petit que (lesser than)
- -le = plus petit ou égal (lesser or equal)
- -gt = plus grand que (greater than)
- -ge = plus grand ou égal (greater or equal)

Les boucles

Boucle for

```
for variable in `seq 0 10`
do
    echo $variable
    instruction
done
```

Les boucles

Boucle While

while [condition]

do

    Instruction

    Instruction

done

Le switch case

```
case $choix in
« valeur1 »)
                instruction
                ;;
« valeur2 »)
                instruction
                ;;
*)
                instruction
                ;;
esac
```

Les intéractions

Pour réaliser les interactions avec un utilisateur

read -p « **message destiné à l'utilisateur** » message

echo $message

Les commandes

Le code de retour

Le code de retour d'une commande est un mécanisme fourni par le shell qui signale à l'utilisateur si l'exécution d'une commande s'est bien déroulée ou s'il y a eu un problème quelconque.

$? : retourne le code de retour de la dernière commande exécutée.

Sortir d'un script

la commande ***exit*** permet de sortir du script et de ce fait arrêter son exécution.

Les commandes

Il est bien sûr possible d’utiliser toutes les commandes fournies dans le système pour réaliser les différentes actions comme *ls, rm ...* .

Avec de la maîtrise, il est possible d’utiliser le concept de fonction pour automatiser des actions répétitives dans le programme.

Voici la syntaxe : function name { }

Optimisation

shellcheck