Introduction aux scripts Bash

DIPLÔME UNIVERSITAIRE DE TECHNOLOGIE – INFORMATIQUE

M1101 :INTRODUCTION AUX SYSTÈMES INFORMATIQUES

SEMESTRE 1 / UE 11 : BASES DE L'INFORMATIQUE

CHAMP DISCIPLINAIRE : ARCHITECTURE MATÉRIELLE – SYSTÈMES D'EXPLOITATION - RÉSEAUX

Boucles for : comparaison C / Shell

```
int i, u = 0;
for (i = 0; i < 20; i++){
        printf ("%d\n", u);
        u = 2 * u + 1;
}
```

```
typeset -i u
u=0
for i in 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
do
      echo $u
      u=2*u+1
done
```

Boucles for : comparaison C / Shell

```
typeset -i u
u=0
for i in {0..19}
do
        echo $u
        u=2*u+1
done
```

```
int i, u = 0;

for (i = 0; i < 20; i++){

        printf ("%d\n", u);

        u = 2 * u + 1;

}
```

```
typeset -i u
u=0
for (( i=0 ; i<20 ; i++))
do
        echo $u
        u=2*u+1
done
```

Boucles for : tableaux

```
#! /bin/bash
Tableau=(élément-zéro élément-un élément-deux élément-trois)
echo ${Tableau[0]}              # élément-zéro (premier élément)
echo ${!Tableau[@]}             # 0 1 2 3 (tous les indices de Tableau)
for i in ${!Tableau[@]}
do
        echo ${Tableau[i]} # élément-zéro
           # élément-un
           # élément-deux
           # élément-trois … tous les éléments de Tableau.
done
```

Boucles for : traiter des fichiers

```
for FICH in *
do
        echo $FICH
done
for FICH in $(ls)
do
        echo $FICH
done
```

Comparer les deux boucles sur un répertoire vide ou non.

Paramètres / arguments de la ligne de commande

numérotés de 1 à n

Le nom de la commande porte le numéro 0

Exemple :

./fichier-script toto titi tata

$0 $1 $2 $3

Paramètres / arguments de la ligne de commande : dans un programme C

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[]) {
        int i;
        char **ptr;
        extern char **environ;
        for (i = 0; i < argc; i++)                      /* echo all command-line args */
                printf("argv[%d]: %s\n", i, argv[i]);
        for (ptr = environ; *ptr != 0; ptr++)           /* and all env strings */
                printf("%s\n", *ptr);
        exit(EXIT_SUCCESS);
}
```

Paramètres / arguments de la ligne de commande : dans un script shell Unix

Variables réservées

$0, $1, ... $9 : les paramètres positionnels

$*, $@ : tous les paramètres en même temps

$# : le nombre de paramètres

Instruction utile

shift : décalage à gauche d'un argument (suppression de $1)

shift n

Paramètres / arguments de la ligne de commande : dans un script shell Unix

```
#! /bin/bash
echo "Exécution de la commande $0"
echo "Nombre de paramètres : $#"
echo "Liste des paramètres : $*"

for i in "$*"
do
        echo "boucle1: $i"
done
```

Paramètres / arguments de la ligne de commande : dans un script shell Unix

```
#! /bin/bash
echo "Exécution de la commande $0"
echo "Nombre de paramètres : $#"
echo "Liste des paramètres : $*"

for i in "$@"
do
        echo "boucle2: $i"
done
```

Paramètres / arguments de la ligne de commande : dans un script shell Unix

```
#! /bin/bash
echo "Exécution de la commande $0"
echo "Nombre de paramètres : $#"
echo "Liste des paramètres : $*"
if [[ $# -ge 1 ]]
then
        echo "Premier paramètre : $1"
else
        echo "Pas de paramètres !"
fi
```

Paramètres / arguments de la ligne de commande : dans un script shell Unix

```
#! /bin/bash
echo "Exécution de la commande $0"
echo "Nombre de paramètres : $#"
echo "Liste des paramètres : $*"

while [[ $# -ge 1 ]]
do
   echo "paramètre : $1"
   shift
done
```

Paramètres / arguments de la ligne de commande : dans un script shell Unix

```
#! /bin/bash
echo "Exécution de la commande $0"
echo "Nombre de paramètres : $#"
typeset -i num=1
while  (( $# ))
do
        echo "Argument numero $num : $1"
        shift
        num=num+1
done
```

Choix multiples

```
case expression in
        cas1)
                liste_commandes1
                ;;
        cas2|cas3)
                liste_commandes2
                ;;
        *)
                liste_commandes3
                ;;
esac
```

```
case $# in
        0)
                variable=`pwd`
                ;;
        1)
                variable=$1
                ;;
        *)
                echo "Erreur de syntaxe"
                exit
                ;;
esac
```

Boucles While et Until

```
while condition
do
        liste_commandes
done
```

```
until condition
do
        liste_commandes
done
```

Exemple While

```
#!/bin/bash
var0=0
LIMITE=10
while [  "$var0" -lt "$LIMITE" ]
do
        echo -n "$var0 "                # -n supprime le retour chariot.
        #                ^        espace, pour séparer les numéros affichés.
        var0=`expr $var0 + 1`           # var0=$(($var0+1)) fonctionne aussi.
                                        # var0=$((var0 + 1)) fonctionne aussi.
                                        # let "var0 += 1"    fonctionne aussi.
done                                    # D'autres méthodes fonctionnent aussi.
echo
exit 0
```

Exemple While

```
#!/bin/bash
# teste_read.sh
read LIGNE
while [[ -n $LIGNE ]]
do
        echo $LIGNE
        read LIGNE
done
```

```
./teste_read.sh
ou bien
./teste_read.sh < texte.txt
```