Le Shell

Le shell est un programme qui permet à l'utilisateur de dialoguer avec le cœur du système Linux (noyau).

il permet de réaliser plusieurs tâches :

* interpréter les commandes
* Exécution des programmes
* il réalise des redirections des entrées / sorties, les pipes (plusieurs commandes), etc…

un script shell est une distribution de commandes lancées par l'interpréteur, les commandes sont écrites dans un fichier .sh pour la visibilité (pas obligatoire)

les droits d'exécution doivent être placés en -x aux moins pour le propriétaire du fichier.

1. Variables

en shell, une variables n’a pas à être déclaré explicitement, dès que qu’un lui affecter une valeur, elle commence à exister

var=15 **(!sans espace!)**

pour afficher le contenue de la variables :

echo $var $ pour afficher son contenue (ca vous retourne 15)

1. Calculs arithmétique

echo $((2\*(4+(10/2))-1))

= 17

((.)) peut aussi servir pour vérifier des conditions arithmétique

echo $(((25+2)<28))

1 (1 = true , 0 = false)

&& ||

1. Paramètres

se sont des données necessaires a l'execution du script, il existe 2 catégories :

* les paramètre positionnelle
* les paramètres spéciaux

les paramètres positionnelle : les paramètre positionnelle sont utilisés pour accéder aux information qui sont fournis sur la ligne de commande lors du lancement du script (./) ces paramètres sont aussi des arguments transmis à une fonction (function(A, B))

les arguments sont placés dans des paramètres qui peut être consultée avec la syntaxe suivante $0 $1 $2 $3 $4 ect.. $9

pour consulter le contenue d’un paramètres qui comporte plus d’un chiffres il faut l’encadrer par des accolades ${12}

par convention $0 contient le nom du programme/script ex : script.sh

Example : affiche\_arguements.sh

#!/bin/bash

echo 0 : $0

if [-n “$1”]; fi

then echo $1; fi

if [-n “$2”];

then echo $2;

arguments\_function.sh

#!/bin/bash

function affiche-ses\_arguements(){

echo “Dans les fonction : ”

echo 0 : $0

echo 1 : $1

ect..

}

la fonction interne “set” peut être utilisée pour fixer les paramètres $1, $2 ect..

set a b c

ont peut modifier les paramètres positionnelles dans un script en utilisant la commande “shift” cette commande permet de décaler l'ensemble des paramètres $1 est remplacé par $2 ect…

exemple

#!/bin/bash

while [-n “$1”]

do

echo $1

shift

done

(-n veut dire si c'est pas vide)

paramètres spéciaux :

il existe 4 paramètres spéciaux :

$\* : liste des paramètres : $1 $2 $3

$@ : la même chaise que $\*

$# : nombre de parametre fournis au script

$$ : contient le PID du shell

PID :

exemple :

while [$# -me 0]

do

echo $1

shift

done

-me = not equal

les condition :

il existe deux instruction permettant de réaliser des condition

* if - elif-else-fi
* case esac

instruction if - if

la syntaxe est la suivante

if [ condition ]

then

echo “blabla”

fi

if [ condition ]

then

echo “”

else

echo “”

fi

autre exemple

if [ condition1 ]

then

echo “”

elif [ condition2 ]

then

echo “”

elif [ condition3]

then

echo “”

else

echo “”

fi

il existe différente condition :

* pour les chaines de caractères
* pour les nombres
* pour les fichiers

ce script test si les deux paramètres sont différents :

#!/bin/bash

if [ $1 != $2 ]

then

echo “les 2 sont différents”

else

echo “les deux sont identique”

fi

if [ $1 != $2 ] && [ $3 != $4 ]

then

echo “les 2 sont différents”

else

echo “les deux sont identique”

fi

l’instruction case esac

le rôle est de tester la valeur d’une même variable de manière plus concises est lisibles

syntaxe :

case $var in

motif1)

echo “”

;;

motif2)

echo “”

;;

\*)

echo “”

;;

esac

6) les fonctions

une fonction en shell et un ensemble d'instruction permettant d'effectuer plusieurs tâches avec des paramètre d'entrée différents

son utilisation vous permettra de rendre votre programme plus lisible et structuré

1 structure d’un script SHELL :

#!/bin/bash // chemin de l'interpréteur

# définitions n des variables globales

# définitions des fonctions

# corps du script

* appel des fonctions

2 déclaration de fonctions :

function nomF() {

blabla…

}

nomF()

Exemple 1 : Paramètres dans et en dehors d’une fonctions

script : parametres.sh

#!/bin/bash

function Affiche\_ses\_Arguments(){

echo “Dans la fonction :”

echo 0 : $0

echo 1 : $1

echo 2 : $2

}

echo “Dans le script :”

echo 0 : $0

echo 1 : $1

echo “appel de la function :”

Affiche\_ses\_Arguments 20 30

echo “je reviens dans le script :”

echo 0 : $0

echo 1 : $1

echo 2 : $2

exemple 2 : test\_arguments.sh

#!/bin/bash

function trois\_arguments(){

if [$# -ne 3]

then echo “nombre d’arguments erroné dans Trois\_arguments()

return

fi

echo “traitements des arguments dans trois\_arguments”

}

trois\_arguments un deux trois quatre

trois\_arguments un deux trois

exemple 3 : additionne