Linux Shell Scripts

DevOps @ SathyaTech
mymail2sateesh@gmail.com

Shell Scripting

Program	Shell Script
*) set of Instructions	*) set of Commands
*) Variable Declaration int a=10; char b='x';	*) No Variable Declaration a=10 b='x'
*) Data types	*) No Data Types
*) Compile & Execute	*) Directly execute
*) Ctrl Statements	*) Ctrl Statements (if, else, while...)
*) Operators (arith , logi, relational)	*) Operators
*) Input and Output stmts (printf, scanf...)	*) Input and Output stmts (echo , print , read)

Shell Scripting

Shell Scrit : set of Commands which can perform a specific task

extention --> ".sh"

Input and Output

Input statements : It will read the data from std Input

Ex:

```
read  n
read  a b c
```

Output statements : It will display the data on std output

Ex:

```
echo   "Hello World"
printf  "\n Good Day"
a=10
echo  $a (echo , print , read)
```

Shell Scripting

Escape Sequences:

\n ---> new Line
\t ---> Horizantal tab
\b ---> backspace
\r ---> carriage return

Expr : It can perform Arithmetic Operators

Ex:

```
a=10
b=20
c=`expr  $a + $b`
c=`expr  $a - $b`
c=`expr  $a \* $b`
c=`expr  $a / $b`
c=`expr  $a % $b`
```

Script: script to read and display values

```
#vi  demo.sh
  printf "\n Enter Three Values : "
  read a b c
  echo Value-1: $a
  echo Value-2: $b
  echo Value-3: $c
```

Execution:

```
#sh  demo.sh
 Enter Three Values : 10 A 12.5
Value-1: 10
Value-2: A
Value-3: 12.5
```

OPERATORS

1) Arithmetic Opertors : + , - , * , / , %
2) Logical Opertors : -a , -o , !
3) Relational Opertors : -lt , -gt , -le , -ge , -eq , -ne
4) String Comparision Opertors :

str1 == str2

str1 != str2

Script: Arithmetic Operations

```
#vi  demo.sh
 printf "\n Enter Two  Values : "
 read a b
      c=`expr $a + $b`
      echo ADD: $c
      c=`expr $a - $b`
      echo SUB: $c
      c=`expr $a \* $b`
      echo MUL: $c
      c=`expr $a / $b`
      echo DIV: $c
      c=`expr $a % $b`
      echo MOD: $c
```

Execution:

```
      Enter Two  Values : 25 12
     ADD: 37
     SUB: 13
     MUL: 300
     DIV: 2
     MOD: 1
```

Script: File Copy :

```
printf "\n Enter Two diff Files : "
        read f1 f2
        cp $f1 $f2
        echo Data Copied!
```

Execution:

```
Enter Two  diff files :  file1  file2
        Data Copied!
```

Control Statements

---

* simple if
* if - else
* nested if - else
* if - elsif
* case
* while loop
* until loop
* for loop
* break
* continue
* sleep
* exit

simple if :

Syntax:

```
if [condition ]
then
    ...stmts
fi
```

Ex:

```
a=10
b=5
if [ $a -gt $b ]
then
    echo  $a is Big
fi
```

if - else :

Syntax:

```
if [condition ]
then
    ...stmts-1
else
    ...stmts-2
fi
```

Ex:

```
a=10
b=15
if [ $a -gt $b ]
then
    echo  $a is Big
else
    echo  $b is Big
fi
```

Script: Even Or Odd Number

```
        #vi  demo.sh
 printf "\n Enter a Number : "
read n
if [ `expr $n % 2` -eq 0 ]
then
  echo Even Number
else
  echo Odd Number
fi
```

Execution:

```
 Enter a Number : 5
      Odd Number
 Enter a Number : 6
      Even Number
```

Script: Script to Verify the Given IP is Valid or Not

```
        #vi  demo.sh
printf "\n Enter an IP Address / HostName : "
read ip
ping $ip -c1 > /dev/nul
if [ $? -eq 0 ]
then
  echo Valid IP / HostName
else
  echo Invalid IP / HostName
fi
```

Execution:

```
Enter an IP Address / HostName : 10.142.0.4
        Valid IP / HostName
```

Script: Script to Verify the Given User is Valid or Not

```
        #vi  demo.sh
printf "\n Enter an User Name : "
read uname
grep $uname /etc/passwd > /dev/nul
if [ $? -eq 0 ]
then
  echo Valid User
else
  echo Invalid User
fi
```

Execution:

```
 Enter an User Name : ravi
Valid User
```

nested if - else :

Syntax:

```
if [condition-1 ]
then
   if [condition-2 ]
   then
      ...stmts-1
   else
      ...stmts-2
   fi
else
    ...stmts-3
fi
```

if - elif :

Syntax:

```
if [condition-1 ]
then
    stmts-1
elif [condition-2 ]
then
    stmts-2
        ....
else
    default stmts
fi
```

Script: to Find Biggest Value :

```
        #vi  demo.sh
printf "\n Enter Three Values : "
read a b c
if [ $a -gt $b -a $a -gt $c ]
then
  echo Biggest Value : $a
elif [ $b -gt $c ]
then
  echo Biggest Value : $b
else
  echo Biggest Value : $c
fi
```

Execution:

```
Enter Three Values : 12 99 35
    Biggest Value : 99
```

File Options

returns 'TRUE'

-e <File Name> ----> a File is Exist
-f <File Name> ----> a File is a Regural Fil(txt,img,docs..)
-d <File Name> ----> a File is a Directory
-r <File Name> ----> a File contains Read permi.
-w <File Name> ----> a File contains Write permi.
-x <File Name> ----> a File contains Execute permi.
-l <File Name> ----> a File is a Link File
-s <File Name> ----> a File contains more then one byte
-O <File Name> ----> a File is Owned by User
-G <File Name> ----> a File is Owned by Group
<File1> -ef <File2> ----> a File-1 is Link with File-2
<File1> -nt <File2> ----> a File-1 is Newer than File-2
<File1> -ot <File2> ----> a File-1 is Older than File-2

Script: to Verify the file is Exist or Not

```
        #vi  demo.sh
 printf "\n Enter a File : "
 read file
if [ -e $file ]
then
   echo File Exist
else
   echo File does not Exist
fi
```

Execution:

```
 Enter a File : file1
    File Exist
```

Script: to Verify the file is a Regular file or Directory

```
        #vi  demo.sh
printf "\n Enter a File : "
read file
if [ -e $file ]
then
  if [ -f $file ]
  then
    echo Is a Regular File
  elif [ -d $file ]
  then
    echo Is a Directory
  fi
else
  echo File does not Exist
fi
```

Execution:

```
Enter a File : mydir
      Is a Directory
```

Script: to Verify the File Link

```
        #vi  demo.sh
printf "\n Enter two diff Files : "
read f1 f2

if [ $f1 -ef $f2 ]
then
  echo File Link is Exist
else
  echo File Link does not Exist
fi
```

Execution:

```
 Enter two diff Files : file SLink
     File Link is Exist
```

case statement:

Syntax:

```
case  <Var>  in
1)  stmts-1  ;;
2)  stmts-2  ;;
      .....
n)  stmts-n  ;;
*)  default stmts  ;;
esac
```

Ex:

```
echo Enter a Value [1-3]:
read  n
case n in
1)  echo ONE  ;;
2)  echo TWO  ;;
3)  echo THREE  ;;
*)  echo Values from 1-3 only! ;;
esac
```

Script: write a script for case Statement :

```
        #vi  demo.sh
printf "\n ---------- \n 1.Server Name \n 2.IP Address"
printf "\n 3.Date \n 4.User Name \n 5.Cal \n ----------"
printf "\n Enter your Option : "
read op
case $op in
1) hostname -f ;;
2) hostname -i ;;
3) date ;;
4) whoami ;;
5) printf " Enter Month and Year : "
   read m y
   cal $m $y ;;
*) echo Invalid Option!
esac
```

Execution:

```
---------------
1.Server Name
2.IP Address
3.Date
4.User Name
5.Cal
------------
Enter your Option : 2
    10.142.0.4
```

While Loop :

Syntax:

```
while [condition ]
do
   ...stmts
done
```

Ex:

```
n=1
while [ $n -le 10 ]
do
   echo $n
   n=`expr $n + 1`
done
```

Script: write a script to Find a Reverse Number:

```
        #vi  demo.sh
printf "\n Enter a Number : "
 read n
 m=0
 while [ $n -gt 0 ]
 do
   r=`expr $n % 10`
   m=`expr $m \* 10 + $r`
   n=`expr $n / 10`
 done
 echo Reverse Number : $m
```

Execution:

```
Enter a Number : 123
Reverse Number : 321
```

until Loop :

Syntax:

```
until [condition ]
do
   ...stmts
done
```

break : this ctrl stmt will terminate a Loop

Ex:

```
until false
do
  echo "Hello World"
  break
done
```

Output:

```
Hello World
```

continue : this ctrl stmt will skip the stmts from its Execution

---

Ex:

```
while true
do
  echo "SATHYA TECH"
    continue
  echo "LINUX"
done
```

Output:

```
SATHYA TECH     SATHYA TECH   .....
```

Script: write a script to display stop watch with Neste while

```
#vi  demo.sh
 h=0
  while [ $h -lt 24 ]
  do
    m=0
    while [ $m -lt 60 ]
    do
      s=0
      while [ $s -lt 60 ]
      do
        clear
        printf "\n\n\t\t\t STOP WATCH "
        printf "\n\t\t\t $h : $m : $s"
        sleep 1
        s=`expr $s + 1`
     done
      m=`expr $m + 1`
    done
    h=`expr $h + 1`
  done
```

Output:

0 : 1 : 5

for Loop:

Syntax:

```
for <Var> in <Values>
do
    ....stmts
done
```

Ex:

```
echo Enter Three Values :
read  a b c
for n in $a $b $c
do
  echo  $n
done
```

for Loop : Ex-1

```
for ip in $(cat myips)
  do
    ping -c1 $ip > /dev/null
    if [ $? -eq 0 ]
    then
     echo $ip is Valid IP / HostName
    else
     echo $ip is InValid IP / HostName
    fi
Done
```

for Loop : Ex-2

```
for usr in $(cat allusers)
  do
    grep $usr /etc/passwd > /dev/null
    if [ $? -eq 0 ]
    then
     echo $usr is Valid User
    else
     echo $usr is InValid User
    fi
done
```

Script: write a script to display of all files :

```
#vi  demo.sh
for file in *
  do
    echo  $file Data :
    echo ----------
    cat $file
  done
```

Output:

```
file1 Data :
----------
HELLO
GOOD DAY
BYE
```

Script: write a script to verify a set of IPs valid or Not

```
#cat myips
        10.142.0.4
        10.142.0.7
        sathyatech.com

# vi demo.sh
    for ip in $(cat myips)
    do
      echo ---------------
      ping $ip -c1 > /dev/nul
      if [ $? -eq 0 ]
      then
       echo Valid IP/ HostName: $ip
      else
       echo InValid IP/ HostName: $ip
      fi
    done
```

Output:

Valid IP/ HostName: 10.142.0.4

InValid IP/ HostName: 10.142.0.7

Positional Parameters

"command line arguments"

$0 ---> File Name
$# ---> No. of Arguments
$1,$2,... $9 ---> other arguments
$* ---> All arguments

Script: write a script for positional perameters :

```
# vi demo.sh
 echo File Name : $0
 echo No. of Arguments : $#
 echo All Arguments : $*

 for a in $*
 do
  echo Arg: $a
 done
```

Execution:

```
# sh demo.sh 10 20 30
    File Name : demo.sh
    No. of Arguments : 3
    All Arguments : 10 20 30
    Arg: 10
    Arg: 20
    Arg: 30
```

Script: write a script file copy :

```
# vi demo.sh
 if [ $# -eq 2 ]
  then
    if [ -e $1 ]
    then
      cp $1 $2
      echo File Copied!
    else
      echo Error: Sourcr File does Not Exist!
    fi
  else
    echo Error: Invalid No.of Arguments
  fi
```

Execution:

```
# sh demo.sh file1 file2
        File Copied!
```