Linux Kernel 🡺

interface between hardware and software

Shell 🡺

* its like a container
* interface between users and kernel/os
* CLI is a shell

Find your shell 🡺

* echo $0
* cat /etc/shells

Types of Shell 🡺

* Gnome : graphical environment in linux
* KDE : another graphical environment in linux
* sh bourne shell
* bash bourne again shell
* csh
* tcsh
* ksh

Shell Scripting 🡺

* a shell script is an executable file containing multiple shell commands that are executed sequentially. The file can contain:

#!/bin/bash 🡺 shell ,

# 🡺 comments

echo cp grep …etc 🡺 commands

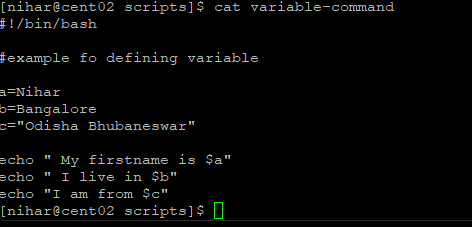
if, while, for, etc 🡺 statements

* shells scripts should have executable permission ex: -rwx r-x r-x
* shell script has to be called from absolute path ex: /home/nihar/a.sh
* or you can call from current location of the script

Basic Scripts 🡺

Graphical user interface

Description automatically generated with medium confidence



Input/Output🡺

* read
* echo

If then statement 🡺

If-then Scripts:

# Comparisons:

-eq equal to for numbers

== equal to for letters

-ne not equal to

!== not equal to for letters

-lt less than

-le less than or equal to

-gt greater than

-ge greater than or equal to

# File Operations:

-s file exists and is not empty

-f file exists and is not a directory

-d directory exists

-x file is executable

-w file is writable

-r file is readable

|  |  |
| --- | --- |
| **Check the variable** |  |
| #!/bin/bash |
| count=100  if [ $count -eq | 100 ] |
| then  echo Count is | 100 |

else

echo Count is not 100 fi

# Check if a file error.txt exist

#!/bin/bash clear

if [ -e /home/iafzal/error.txt ]

then

echo "File exist" else

echo "File does not exist"

fi

# Check if a variable value is met

#!/bin/bash

a=`date | awk '{print $1}'` if [ "$a" == Mon ]

then

echo Today is $a else

echo Today is not Monday

fi

# Check the response and then output

#!/bin/bash clear

echo

echo "What is your name?" echo

read a echo

echo Hello $a sir echo

echo "Do you like working in IT? (y/n)" read Like

echo

if [ "$Like" == y ] then

echo You are cool

elif [ "$Like" == n ] then

echo You should try IT, it’s a good field echo

fi

# Other If statements

If the output is either Monday or Tuesday

if [ “$a” = Monday ] || [ “$a” = Tuesday ]

Test if the error.txt file exist and its size is greater than zero

if test -s error.txt

if [ $? -eq 0 ] If input is equal to zero (0)

if [ -e /export/home/filename ] If file is there

if [ "$a" != "" ] If variable does not match

if [ error\_code != "0" ] If file not equal to zero (0)

***for* loop Scripts:**

# Simple for loop output

#!/bin/bash

for i in 1 2 3 4 5 do

echo "Welcome $i times" done

# Simple for loop output

#!/bin/bash

for i in eat run jump play do

echo See Imran $i done

# for loop to create 5 files named 1-5

#!/bin/bash

for i in {1..5} do

touch $i done

# for loop to delete 5 files named 1-5

#!/bin/bash

for i in {1..5} do

rm $i done

**Specify days in for loop**

#!/bin/bash i=1

for day in Mon Tue Wed Thu Fri do

echo "Weekday $((i++)) : $day" done

# List all users one by one from /etc/passwd file

#!/bin/bash i=1

for username in `awk -F: '{print $1}' /etc/passwd` do

echo "Username $((i++)) : $username" done

case Scripts:

#!/bin/bash

echo

echo Please chose one of the options below echo

echo 'a = Display Date and Time' echo 'b = List file and directories' echo 'c = List users logged in'

echo 'd = Check System uptime' echo

read choices case $choices in

1. date;;
2. ls;;
3. who;;
4. uptime;;

\*) echo Invalid choice - Bye.

esac

**This script will look at your current day and tell you the state of the backup**

#!/bin/bash

NOW=$(date +"%a") case $NOW in

Mon)

echo "Full backup";; Tue|Wed|Thu|Fri)

echo "Partial backup";; Sat|Sun)

echo "No backup";;

\*) ;;

esac

do-while Script

# Script to run for a number of times

#!/bin/bash c=1

while [ $c -le 5 ] do

done

echo "Welcone $c times" (( c++ ))

# Script to run for a number of seconds

#!/bin/bash

count=0 num=10

while [ $count -lt 10 ] do

echo

echo $num seconds left to stop this process $1 echo

sleep 1

num=`expr $num - 1` count=`expr $count + 1` done

echo

echo $1 process is stopped!!! echo