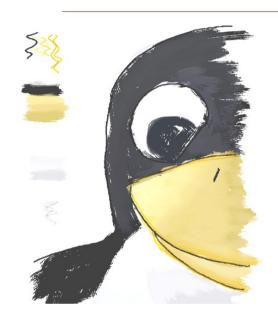
UTFSM JMC 2015





S.O Linux

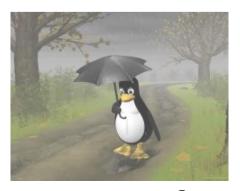
Tema 7



M Varas

Tema: Programación en BASH

- Bash
- Introducción a programación
- Sentencias básica
- Ejemplos



BASH (Bourne Again Shell)

- Bash
 - Interprete de comandos de GNU
- No sólo soporta comandos
- Soporta programación (tomar decisiones)
 - Scripts
- Son útiles para:
 - Tareas administrativas de Linux
 - Tareas tediosas o repetitivas (respaldos)
 - Hacer tareas rápidamente

BASH (Bourne Again Shell)

- Variables de entorno (\$HOME, \$PWD, etc)
- Espacio
 - Espacio o tab : Una frases
- Secuencia de caracteres (palabras)
 - una_sola_palabra, dos palabras, j0s3_m1gu3l

BASH (Bourne Again Shell)

- Metacaracter
 - | & ; () < > [espacio] , [tab]
- Operador de control
 - || comando1 || comando 2
 - && comando1 && comando2

Ejemplo: (supongamos que el archivo procesos no existe)
\$ cat procesos || Is
\$ cat procesos && Is

Tips de Teclado

- C-a Inicio Linea
- C-e Fin Linea
- C-w Cortar palabra atras
- C-u Cortar linea atras
- C-k Cortar linea adelante
- C-y Pegar

Tips de Teclado

- C-I Limpiar Terminal
- C-c Cancelar
- C-d Borrar caracter
- C-s Bloquear terminal ScrollLock
- C-q Desbloquear terminal
- C-r Buscar en el Historial
- C-d logout

```
$ variable=1
 echo $variables
$ echo $VARIABLES
$ variable=2
 echo $variable
 otra_variable=$variable
 echo $otra_variable
 variable=0 ; echo $variable
0
```

```
$ echo $HOME
/opt/mvaras
$ echo "Mi directorio es $HOME"
Mi directorio es /opt/mvaras
$ date
lun may 4 21:53:39 CLT 2009
$ fecha=date
$ echo "La fecha de hoy es $fecha"
La fecha de hoy es date
$ fecha=$(date)
$ echo "La fecha de hoy es $fecha"
La fecha de hoy es lun may 4 21:53:39 CLT 2009
```

```
$ valor1=1
$ valor2=2
$ echo $valor1+$valor2
1 + 2
$ echo $(($valor1+$valor2))
3
 echo \$((4+2))
6
 echo "\$((4+2))"
6
$ echo '$((4+2))'
$((4+2))
 let suma=3+4; echo $suma
7
```

```
$ cd /tmp
[/tmp]$ cd $HOME
$ cd /tmp
[/tmp]$ cd ~

$ echo ~
/home/mvaras
```

Repaso de comandos

- cd
- Is
- mkdir
- rm
- cp
- mv
- pwd
- cat
- tail
- head

- less
- grep
- sort
- echo
- alias
- WC

- \$HOME, ~
 - /home/mvaras
- \$PWD, ~+
 - /opt/respaldos
- \$OLDPWD, ~-
 - /tmp
- \$PATH
 - /usr/local/bin:/usr/bin:/bin

- \$LANG
 - es_CL.UTF-8
- \$USER
 - mvaras
- \$UID
 - 501
- \$RANDOM
 - 10343

- \$HISTFILE
 - \$HOME/.bash_history
- \$HISTSIZE
 - 500
- \$HOSTNAME
 - brain
- \$OSTYPE
 - linux-gnu
- \$PPID
 - 5220

- \${BASH_VERSINFO[@]}
 - 3 00 14 1 release i386-redhat-linux-gnu
- \$EDITOR
 - jed

Declarar y matar Variables

- set
- unset VARIABLE

Expansión e interpretación

```
$ echo {a,b,c}XX{1,2}
aXX1 aXX2 bXX1 bXX2 cXX1 cXX2
$ echo {5..12}
5 6 7 8 9 10 11 12
$ ls *a*
programa avanzado respaldos
$ ls a?z
aiz a3z a5z aaz aBz
```

Más ejemplos

- Crear archivo "ejemplo" con cualquier dato.
 - probar:
 - cat ejemplo
 - miejemplo=\$(cat ejemplo)
 - echo \$miejemplo

Arreglos

```
$ XX=(pato vicky black pamela hernan zapallito)
$ echo $XX
pato
$ echo ${XX[2]}
black
$ echo ${\pm XX[3]}
6
$ echo ${ #XX}
4
```

Arreglos

```
$ XX=(pato vicky black pamela hernan zapallito)
amela
$ echo ${XX[5]:2:3}
pal
$ echo ${XX[0]/a}
pto
pbto
$ echo ${XX[@]}
pato vicky black pamela hernan zapallito
```

Expansión de HISTORY

history

```
2200 cd /tmp
2201 head /etc/fstab
2202 find . -name '*ejemplo*'
```

- !2200
 - Ejecutará ese comando
- !!
 - Ejecutará el último comando

Programando... ahora si....

- man 1 bash
- help comando
 - if
 - for
 - while/until
 - case
 - select
 - function

Programando - IF

```
If condicion_verdadera;
 then
    lista de comandos1;
 else
    lista de comandos1;
fi
#!/bin/bash
if [ $HOME = "/home/mvaras" ];
  then
   echo "OK";
  else
   echo "FALSO";
fi
```

Programando - CASE

```
case expresion in
  patron1)
       lista de comandos paa patron1;
   patron2)
       lista de comandos para patron2;
    * )
       lista de comando para otro caso;
esac
#!/bin/bash
case $HISTCMD in
  1172)
    echo "Vamos en el comando 1172 del historial";;
  1174)
    echo "Vamos en el comando 1174 del historial";;
  1180)
    echo "Vamos en el comando 1180 del historial";;
  * )
    echo "No vamos ni en 1172, 1174, 1180.";;
esac
```

Programando - FOR

```
for variable in elemento1 elemento2 elemento 3;
 do
   lista de comandos;
done
for variable in (lista);
 do
   lista de comandos;
done
for i in elem1 elem2 elem3 elem4;
 do
    echo "Estamos en el elemento $i";
    touch $i;
    echo "contenido del archivo $i" > $i;
done
```

Programando - WHILE/UNTIL

```
while condicion
   do
    lista de comandos;
done

until condicion
   do
    lista de comandos;
done
```

Programando - SELECT

```
select variable in elem1 elem2 elemN
   do
        lista de comandos;
done

select variable in (lista);
   do
        lista de comandos;
done
```

Scripts - Ejemplo1

```
#!/bin/bash
TEXTO="hola mundo"
echo $TEXTO
```

Scripts – Ejemplo 2

```
#!/bin/bash
# Convertir imagenes a tamaño 150x150

for foto in $(find . -iname '*jpg');
   do
      echo "Convirtiendo $foto..."
      convert $foto -geometry 150x150 img/$foto
done
```

Scripts – Ejemplo 3

```
#!/bin/bash
# Actualizacion remota de maquinas

for maquina in $(cat computadores );
   do
     echo "Actualizando el sistema..."
     ssh $maquina "yum update && yum clean packages"
done
```

Scripts - Ejemplo 4

```
#!/bin/bash
# Existe el archivo ?, es de ejecucion ?
archivo=file.txt
if[ -f $archivo ]; then
 echo "El archivo $archivo existe"
 if[ -x $archivo ]; then
    echo "El archivo es de ejecución"
 else
    echo "El archivo $archivo NO existe"
 fi
fi
```

Scripts – Ejemplo 5

```
#!/bin/bash
i=10;
while [ $i -lt 20 ]; do
    echo $i;
    let i=$i+1;
done

#!/bin/bash
-lt: menor que
-gt: mayor que
-eq: igual a
-le: menor o igual
-ge: mayor o igual
-ne: distinto
```

Opciones útiles para if

-d archivo el archivo existe y es directorio

-e archivo el archivo existe

-f archivo el archivo existe y no es directorio

-r archivo el archivo tiene permisos de lectura

-w archivo el archivo tiene permisos de escritura

el archivo tiene permisos de ejecución

-n \$VAR \$VAR no es nulo

\$VAR es nulo

-z \$VAR

-x archivo

-lt (<): menor que

-eq (=): igual a

-ge(>=): mayor o igual

-gt (>): mayor que

-le(<=): menor o igual

-ne(!=): distinto