emewinchester
www.wuolah.com/student/emewinchester

Practica 4 APSO resuelta.pdf

PRÁCTICAS RESUELTAS (incluye teoría)

2° Administración y Programación de Sistemas Operativos

Grado en Ingeniería Informática

Escuela Técnica Superior de Ingeniería
UHU - Universidad de Huelva

APSO: PRÁCTICA 4

MANEJO DE LA SHELL (IV)

Marina Delgado Pérez
GRADO INGENIERÍA INFORMÁTICA | HUELVA

escuela de negocios
CÁMARA DE SEVILLA

diferénciate

Con la mejor formación práctica

www.mastersevilla.com

Titulación de prestigio
en el sector empresarial

MÁSTER EN DIRECCIÓN Y GESTIÓN DE RECURSOS HUMANOS

Contenido

1.- TIPOS DE SHELL 2
2.- VARIABLES 2
3.- ALIAS 4
4.- OPCIONES 4
5.- FICHEROS DE ARRANQUE 4

PRELUDIO TEÓRICO

1.- TIPOS DE SHELL

Shell, también conocido como intérprete de comandos (IC) es el programa que provee una interfaz de usuario para acceder a los servicios del sistema operativo (SO).

Por defecto, al abrir la consola de comandos de Polifemo, se ejecuta el IC `bash`. Sin embargo, Unix también incorpora otros intérpretes de comandos (o tipos de Shell):

- `ksh (kshell)`
- `csh (cshell)`

Para todas las Shell de Unix sirven los mismos comandos, pues siguen el estándar POSIX.

Para pasar a cualquiera de los dos IC nombrados, tan solo hay que ejecutar el comando.

Ejemplo: prompt-> `ksh`

Debido a que nosotros entramos por defecto en el `bash`, al ejecutar cualquier otra Shell, se crea un proceso hijo, siendo la Shell bash el proceso padre.

Para salir de un IC ejecutado como proceso a partir de otro, tan solo hay que ejecutar el comando `exit`.

Para comenzar con otra Shell que no sea `bash` al iniciar la consola de comandos, se tendría que modificar el fichero de arranque.

2.- VARIABLES

Sirven para almacenar un conjunto de caracteres. Es decir, a una variable se le asigna un conjunto de caracteres. Las variables permiten que se ahorre a la hora de escribir determinados conjuntos de caracteres como, por ejemplo, una ruta.

IMPORTANTE: las variables no guardan comandos, esa es la función de los alias.

IMPORTANTE: Para trabajar con variables, se recomienda declararlas siempre en mayúsculas, para diferenciarlas de los alias.

DECLARACIÓN DE VARIABLES:

Ejemplo: "*Acceder a la carpeta donde están las prácticas*".

Para evitar escribir la ruta repetidamente, declaramos una variable:

```
D="/home/so/velez/MI"
```

Para usarla, tan solo hay que acceder a ella: (mediante el símbolo dólar **$**)

```
cp $D/p4.txt prac4
```

¡**OJO**! Se recomienda entrecomillar siempre el contenido de las variables.

El tiempo de vida de las variables declaradas se reduce al tiempo que se tenga la sesión donde fue declarada iniciada. Para hacer una variable permanente, se debe modificar el fichero de arranque y añadirla.

Para mostrar el contenido de una variable, tan solo hay que ejecutar el comando `echo`.

Ejemplo: `echo $D`

Algunas variables muy importantes y predefinidas por el sistema son:

- HISTFILE: Histórico de los comandos que se van ejecutando. Guarda la ruta del fichero .bash_history (fichero que comienza por punto "." porque es un fichero oculto).
- HOME: Almacena la ruta de nuestra carpeta personal.
- HOSTNAME: Almacena el nombre del host (servidor).
- LOGNAME: Almacena nuestro nombre de usuario.
- PATH: Variable clave para entender el funcionamiento de cualquier máquina Unix. Al lanzar el comando de ejecución de un fichero ejecutable (fichero con permiso de ejecución), la máquina busca si ese fichero se encuentra en las rutas definidas en la variable PATH. Si encuentra el fichero, se ejecuta. En caso de no encontrarlo, obtendremos un mensaje de error (*NOT FOUND*). Otra forma de ejecutar un fichero ejecutable, en caso de estar en una ruta distinta a aquellas guardadas en la variable PATH, es escribir la ruta del fichero, para que el SO pueda localizarlo. Para meter una ruta en la variable PATH, sin perder lo que ya guardaba anteriormente, se hace de la siguiente manera:

  ```
  PATH=$PATH:ruta_que_queramos_añadir
  ```

  Los dos puntos (:) sirven para separar las rutas. Se recomienda añadir la ruta `carpeta_actual`, que se hace añadiendo el punto (.) a la variable PATH. De esta manera, el SO buscará siempre en la carpeta actual en la que estemos.
- PS1: Almacena el prompt.
- PS2: Almacena el prompt secundario. Este prompt solo aparece cuando hemos ejecutado un comando y hemos dejado algún paréntesis o algunas comillas sin cerrar.
- PWD: Variable que guarda en todo momento la ruta de la carpeta en la que estamos situados actualmente.

Para mostrar la lista de variables definidas en el sistema, ejecutar:

```
set | more
```

Para borrar una variable del sistema, ejecutar:

```
unset nombre_variable_que_queremos_eliminar
```

3.- ALIAS

Los alias tienen la función de renombrar una serie de comandos. Herramienta muy potente del IC, pues no solo renombra los comandos, sino que se puede asignar un alias a un comando con sus opciones, o varios comandos relacionados mediante tuberías, etc.

Ejemplo: `copiar="cp"`

A partir de ahora, podría utilizar tanto `copiar` como `cp`, que harían lo mismo.

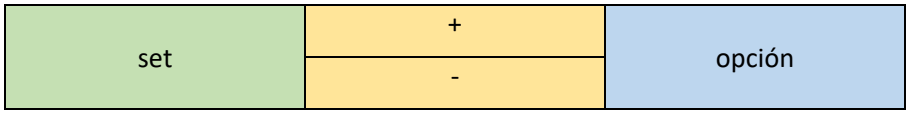
IMPORTANTE: Para trabajar con alias, se recomienda declararlas siempre en minúscula, para diferenciarlas de las variables.

Para ver una lista de todos los alias, ejecutamos el comando `alias`.

Al igual que las variables y las opciones de Shell, para mantener los cambios realizados durante una sesión, se tiene que modificar el fichero de arranque. En caso de no modificarlo, cualquier alias declarado anteriormente se habrá perdido, y tendremos que volver a declararlo.

4.- OPCIONES

Podemos personalizar el intérprete de comandos para que trabaje de una forma u otra mediante las opciones, ya sea dándolas o quitándolas. Sintaxis:



Para conocer una lista de todas las opciones con las que se puede personalizar el interprete de comandos, ejecutar `man set`.

5.- FICHEROS DE ARRANQUE

IMPORTANTE: Antes de modificar el fichero de arranque, aunque sea mínimamente, se recomienda hacer una copia de seguridad en la misma carpeta personal, en caso de perderlo o borrar información importante.

El fichero de arranque se encuentra en nuestra carpeta personal como fichero oculto (`.profile`). Para visualizarlo, tan solo hay que ejecutar `more .profile`.

En el fichero de arranque debemos guardar todas las modificaciones que queramos que se mantengan permanentemente (por ejemplo, alias y variables declaradas).

Para que los cambios realizados en el fichero `.profile` tomen efecto, hay que guardar el fichero y abrir un nuevo terminal. Ese nuevo terminal ya funcionará con los cambios realizados.

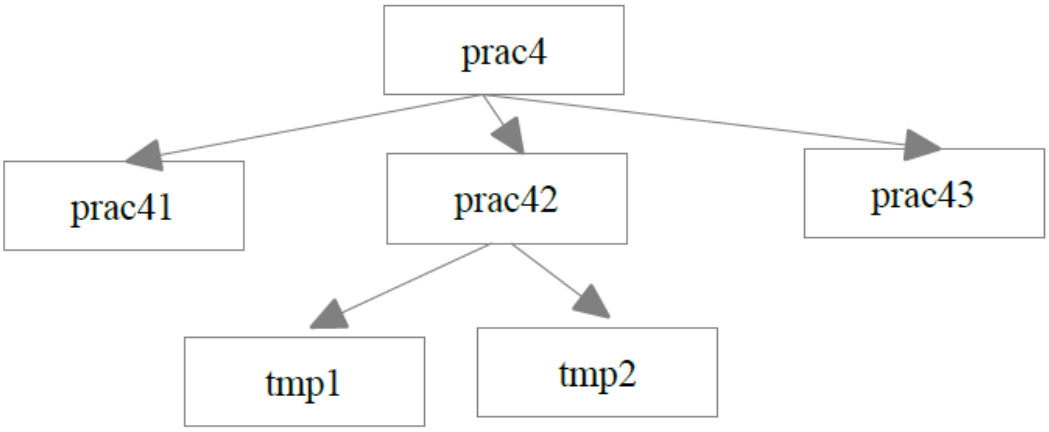
Otra forma de hacer que los cambios del .profile tomen efecto es ejecutar el siguiente comando:

```
. .profile
```

Si tan solo ejecutáramos el fichero `.profile`, se crearía un proceso hijo del bash y sólo se realizarían los cambios en ese proceso hijo. Al cerrar ese proceso, volveríamos a al punto inicial, sin los cambios realizados.

RESOLUCIÓN PRÁCTICA 3

1. Cree con un solo comando el siguiente árbol de directorios a partir de su directorio personal




```
mkdir prac4 prac4/prac41 prac4/prac42 prac4/prac42/temp1
prac4/prac42/temp2 prac4/prac43
```

2. Muévase al directorio `prac43`. Quite todo tipo de permisos para el grupo y el resto de los grupos al directorio `prac4`, `prac42`, `tmp1` y `tmp2` con un solo comando y usando rutas relativas.

```
cd prac4/prac43
chmod go-rwx .. ../prac42 ../prac42/temp1 ../prac42/temp2
```

3. Muévase al directorio `prac42`. Cree con el `cat` un fichero en el directorio `tmp1` que se llame `prueba` y contenga la frase `Este es un fichero de prueba`. Añada con el comando `cat` una nueva frase al fichero sin borrar la anterior que diga `Esta frase se ha añadido después.`

```
cd ../prac42
cat >temp1/prueba
cat >>temp1/prueba
```

4. Visualice el contenido de las primeras 52 variables del sistema. Muestre por pantalla el contenido de la variable `PATH` (sólo esa variable).

```
set | more -52 -f
echo $PATH
```

5. Cree una variable llamada `ORIGEN` que contenga la ruta absoluta al directorio `/home/so/velez/MI`. Cree dos variables llamadas `DESTINO1` y `DESTINO2` con la ruta absoluta al directorio `tmp1` y `tmp2` respectivamente del directorio `prac42`. Visualice el contenido de estas dos variables (sólo de estas dos). Use las variables `ORIGEN` y `DESTINO1` para copiar todos los ficheros del directorio `/home/so/velez/MI` que contienen una `p` en su nombre y terminan en `.txt` al directorio `tmp1` de `prac42`.

```
ORIGEN="/home/so/velez/MI"
DESTINO1="$PWD/temp1"
DESTINO2="$PWD/temp2"
echo $DESTINO1
echo $DESTINO2
cp $ORIGEN/p*.txt $DESTINO1
```

6. Muévase al directorio `prac43` con rutas relativas. Cree un alias llamado `fnuevos` que busque los ficheros (no directorios) a partir de su directorio personal a los que se haya accedido hace menos de 3 horas, empiecen por punto y terminen en `e`, `y` o `c` y visualice su contenido uno a uno.

```
cd ../prac43
alias fnuevos=”more $(find $HOME ! -type d -amin -180 -iname
".*[eyc]")”
```

7. Visualice todos los alias del sistema. Cree cinco nuevos alias:

Uno se llamara `dir` y visualizará página a página todos los ficheros del directorio en el que está (incluso los que empiezan por punto), con todos los permisos en orden alfabético.

`dirinverso` hará lo mismo pero saldrán ordenados en orden inverso.

Otro se llamará `fecha` y visualizará la fecha con el siguiente formato: `Hoy es <día de la semana>, <día del mes> de <mes> de <año>. Chao`

El otro se llamará `hora` y visualizará la hora con el siguiente formato: `Son las <hora> horas y <minuto> minutos. Chao`

El último debe conseguir que cuando ejecute `cd..` (sin el espacio en blanco entre el `cd` y los dos puntos) no produzca error.

```
alias
(por hacer)
(por hacer)
alias fecha='date "+%nHoy es %A, %d de %B de %Y. Chao.%n"'
alias hora='date "+%nSon las %k horas y %M minutos. Chao%n"'
alias cd..="cd .."
```

8. Cree un fichero llamado `usuarios` en el directorio `tmp2` con la lista de todas las sesiones que hay abiertas en este instante ordenadas alfabéticamente. Use rutas relativas.

```
who | sort >../prac42/temp2/usuarios
```

9. Muévase al directorio `prac4`. Ejecute el intérprete de comandos `ksh`. Visualice las variables activas en este interprete de comandos. Modifique el prompt para que visualice la frase `Comando::>`

```
ksh
set | more
PS1='Comando::>'
```

10. Salga del interprete de comandos `ksh` y vuelva al `bash`. Cuente con la ayuda del `who` y del `wc` el número de sesiones que hay abiertas (debe salir un número).

```
exit
who | wc -l
```

11. Modifique el prompt principal para que visualice su nombre de usuario en vez del nombre de la máquina. A continuación el directorio donde está mediante ruta absoluta y al final debe aparecer el símbolo `>` (Ejemplo: Para el usuario velez actualmente sale `velez@Polifemo:~/prac4$`. Después de modificar el prompt debe salir `velez: /home/velez/prac4>`).

```
PS1='$LOGNAME: $PWD>'
```

12. Modifique el prompt secundario para que emita el siguiente mensaje: `cierre las comillas, por favor.`

```
PS2='cierre las comillas, por favor'
```

13. Edite con el `joe` el fichero `.profile`. Modifique la variable `PATH` para que siempre se ejecuten los ficheros que se encuentran en el directorio actual. Añada la variable `ORIGEN` creada en el apartado 5 para que se pueda usar siempre que entre en una nueva sesión.

```
joe $HOME/.profile
(añadir ORIGEN y modificacion PATH, guardar)
```

14. Cierre la sesión. Vuelva a entrar. Añada los alias `hora` y `fecha` al `.profile` para que siempre se puedan ejecutar. Haga lo necesario para que os alias introducidos en el `.profile` se vuelvan activos y puedan ser usados.

```
joe .profile
(añadimos alias y guardamos)
. .profile
```

NOTA IMPORTANTE: Las cuestiones en rojo están sin resolver. Subiré la práctica totalmente terminada tras ir a tutoría y resolver las dudas. Un saludo.