Scripts en Shell

Laboratorio de Sistemas Operativos I

“

Shell es el programa que provee la interfaz entre el usuario y el kernel. Es el intérprete de línea de comandos (ejecuta directamente). Shell más popular: **bash** (born again shell).

¿Qué es un Script?

- En Informática, es un guión, archivo de órdenes o archivo de procesamiento por lotes (wikipedia).
- Es un programa que se almacena en un archivo de texto plano.
- Puede ejecutar diversas tareas, como combinar comandos, interactuar con el SO o con el usuario.
- El **shell** es el intérprete de estos programas.

Generalidades de un script

- Se escriben en el lenguaje del shell.
- Al interactuar con el shell estamos creando pequeños programas.
- Pueden combinar en una única acción tareas rutinarias.
- Pueden ser de una sola línea o de varias pantallas.

Pasos para crear un script

- En consola, se abre un archivo nuevo con un editor de texto.
- Escribir el encabezamiento o shebang: **`#!/bin/bash`**
- Redactar las órdenes y guardar.
- Modificar los permisos: `$chmod u+x nombre_script`
- Ejecutar el archivo: `$./nombre_script`

Variable PATH

- **./** indica al shell que busque el archivo a ejecutar en el directorio actual.
- porque el directorio actual NO ESTÁ en la lista de directorios en la que el shell busca los comandos para ejecutar.
- La lista de directorios donde busca está definida en la variable de entorno PATH.
- Para ver su contenido: `$echo  $PATH`

Variables de Entorno

- Forman un conjunto de valores dinámicos cargados en memoria, que definen el ambiente del shell en el cual trabaja el usuario.
- Para ver lista de variables de tu entorno: `$printenv`
- Para ver el valor de una variable de entorno:
  - `$printenv VARIABLE  o  $echo $VARIABLE`
- Su valor se establece al abrir una sesión.
- Ejemplos: `USER, PWD, HOME, PATH, HOSTNAME, SHELL, LOGIN`

Comandos `export` y `unset`

- Cualquier variable puede convertirse en una variable de entorno con el comando **`export`**.
- Se define la variable y se transfiere al entorno (queda disponible para el shell) con el comando export

```
alumno@mipc:~$ export MIVAR=”LSOI”
alumno@mipc:~$ echo $MIVAR
```

- Se elimina con el comando **`unset`**.

```
alumno@mipc:~$ unset $MIVAR
```

Parámetros

- Es posible pasar a un script, desde la línea de comandos, los argumentos que necesita para su ejecución.
- Estos argumentos se denominan PARÁMETROS.
- Existen dos categorías:
  - Posicionales
  - Especiales

Parámetros posicionales

- Son los argumentos pasados al script cuando es invocado.
- Son almacenados en variables reservadas denominadas **1**, **2**, **3**, ...
- Sus valores son invocados con las expresiones **$1**, **$2**, **$3**, ...
- Ejemplo: muestra_param1

Uso de “” con parámetros:

```
$ echo “Buen día $1” --->>   Buen día Pedro
$ echo ‘Buen día $1’ --->>   Buen día $1
```

Parámetros especiales

- También son variables reservadas y permiten realizar operaciones sobre los mismos parámetros. Ejemplo: muestra_param2

Parámetro	Significado
$0	Nombre del script tal como se invoca
$*	Conjunto de todos los parámetros en 1 solo argumento
$@	Conjunto de argumentos, un argumento por parámetro
$#	Número de parámetros pasados al script
$?	Código de retorno del último comando
$$	PID del shell que ejecuta el script
$!	PID del último proceso ejecutado en 2° plano

Operaciones aritméticas: expr

- Necesita como mínimo dos operandos y un operador.
- Sintaxis: **`expr [operando1] operador [operando2]`**

Operación	Operador	Observaciones
Suma	+	
Resta	-	
Multiplicación	*	* por ser carácter especial
División	/	
Módulo	%	Resto de una división

Ejemplo: Operaciones

```
#!/bin/bash
suma=`expr $1 + $2`
echo "Suma"
echo $1 + $2 = $suma

resta=`expr $1 - $2`
echo "Diferencia"
echo $1 - $2 = $resta
```

```
multi=`expr $1 \* $2`
echo "Producto"
echo $1 x $2 = $multi

div=`expr $1 / $2`
echo "Cociente"
echo $1 / $2 = $div
```

Ejemplo: Cuadrado de un número

```
#!/bin/bash
#Cuadrado de un numero

resultado=`expr $1 \* $1`
echo "El resultado de el cuadrado de $1 es $resultado”
```

Evaluar cadenas: `test`

- Para evaluar cadenas, texto, archivos o números, se utiliza **`test`**.
- Sintaxis `test cadena1 valor cadena2`
- Permite comparar o comprobar cualidades o valores de las cadenas evaluadas para poder obtener un resultado positivo o negativo que haga actuar a tal o cual comando de uno u otro modo.
- Dentro de estructuras lógicas se suele usar sólo corchetes en lugar del comando 'test' : `[cadena1 valor cadena2 ]`

Cómo evaluar cadenas:

=	igual, las dos cadenas de texto son exactamente idénticas
!=	no igual, las cadenas de texto no son exactamente idénticas
<	es menor que (en orden alfabético ASCII)
>	es mayor que (en orden alfabético ASCII)
-n	la cadena no está vacía
-z	la cadena está vacía

Ejemplo: Comparación de cadenas (1)

```
 #!/bin/bash
# evaluación de cadenas de texto.
# parametros dos cadenas de texto como argumento
`test $1 != $2`
#mostrara 0 si son distintos y 1 si son iguales
echo $?
```

Ejemplo: Comparación de cadenas (2)

```
#!/bin/bash
echo "Ingrese primera cadena de caracteres"
read cadena1
echo "Ingrese segunda cadena de caracteres"
read cadena2
if [ $cadena1 = $cadena2 ]; then
  echo "Son iguales"
else
  echo "Son distintas"
fi
```

Evaluar números

- Con la evaluación de cadenas se podrían evaluar números, pero hay que tener en cuenta no evalúa como números
- Ejemplo [04 = 4] devolvería '1', ya que el '=' evalúa si 04 es igual a 4, y como cadena de texto, no lo es.
- Para evaluar números se usa:
  - **-lt** (menor que)
  - **-le** (menor o igual)
  - **-gt** (mayor que)
  - **-ge** (mayor o igual)
  - **-eq** (igual)
  - **-ne** (no igual)

Evaluar archivos

- **-f** indica si el fichero existe.0 si es verdadero y 1 si es falso.
- **-s** indica si el fichero existe y no esta vacio.0 si verdad, 1 si falso.
- **-r** indica si tiene permiso de lectura.0 verdad, 1 falso.
- **-w** indica si tiene permiso de escrituta.0 verdad, 1 falso.
- **-x** indica si tiene permiso de ejecución.0 verdad, 1 faslo.
- **-d** indica si es directorio. 0 verdad, 1 falso.
-
- Estos parámetros se pueden unir usando los enlazadores :
- **-a** and
- **-o** or
- **!** not

¡Gracias!

¿Preguntas?