Curso 1º SMR

Módulo: SOM

Sesión 1: Introducción a Programación shell

¿Qué pretendemos conseguir?

Saber qué es un script shell.
Hacer ejecutables los scripts.
Realizar nuestro primer script.
Variables posicionales: qué son y cómo se utilizan.
shift.
Interactuar con el usuario con el comando read.

Desarrollo de la sesión

Introducción

Un shell es un programa interactivo que se interpone entre el usuario y el núcleo del sistema permitiendo que aquél envíe órdenes al sistema.

Estudiamos en la unidad Sesión de Linux la forma de comunicarnos con el sistema de orden en orden o como mucho, agrupándolas con ; o con redireccioamientos.

En esta unidad aprenderemos a crear programas basados en instrucciones del sistema. Para ello las incluiremos en un fichero de texto (denominado **script**) que haremos ejecutable para que puedan ser interpretados por el "intérprete" de comandos.

Estos programas, para que tengan una mayor potencia, constarán también de estructuras de flujo de control como son if, case, while o for que estudiaremos.

Existen varios programas shell que pueden instalarse en nuestros sistemas GNU/Linux. Nosotros vamos a centrar el estudio en el shell bash.

Creamos nuestro primer script

Ejercicio1: Crea el subdirectorio programasshell en el directorio HOME.
Ejercicio2: Utilizando el programa gedit, creamos, en el directorio del ejercicio1, el programa miprimer.sh. Este fichero contendrá la línea **echo "hola"**.

Después de crear el fichero intentamos ejecutarlo con ./miprimer.sh

Ejercicio3: ¿Podrías explicar por qué hay que utilizar el ./ delante del nombre del fichero?

Ejercicio4: ¿Se ejecuta algo? ¿Aparece algún mensaje de error? ¿Cuál?

Observarás que el mensaje es **permiso denegado.**

Escribe ahora, en el terminal, lo siguiente:

chmod +x miprimer.sh

Ejercicio5: Vuelve a intentar la ejecución del script.

Ahora sí se podrá. En la sesión correspondiente a permisos de ficheros y directorios trataremos el comando chmod con más amplitud. Por ahora nos bastará saber que en Linux los ficheros necesitan tener el permiso de ejecución para poder ser "ejecutados".

Esto último, sin ser falso, no es totalmente cierto. Podemos lanzar un script llamando al intérprete de comandos.

Ejercicio6: Introduce la siguiente orden en el intérprete de comandos: **chmod -x miprimer.sh**. Ahora, al ejecutarlo obtendrás el mensaje de permiso denegado. Teclea ahora **sh miprimer.sh** ¿Se ejecuta?

El comando **sh** llama a un nuevo intérprete de comandos que, esta vez, lo que hace es interpretar los comandos que se encuentren en el fichero miprimer.sh.

Cada vez que llamamos a sh decimos que estamos creando un subshell hijo del shell anterior.

Ejercicio7: teclea sh ¿qué prompt aparece? Teclea ahora exit ¿Qué ha ocurrido?

Variables posicionales

En la sesión de terminal Linux estudiamos que un mismo comando puede realizar distintas cosas dependiendo de los parámetros que utilicemos al invocarlo. Por ejemplo: ls mostrará el listado del directorio, sin embargo, ls -a mostrará los archivos y directorios que estén ocultos (por comenzar su nombre con .).

Bien, en programación shell podemos utilizar parámetros en nuestros script.

Ejercicio8: Crea, en el directorio de la primera actividad, el fichero parametros.sh. Este fichero debe contener las siguientes líneas:

```
echo $1
echo $2
```

Hazlo ejecutable. Lánzalo ¿qué ocurre?

Lánzalo ahora con ./parametros.sh hola mundo ¿Qué ocurre?
Lo que ha ocurrido es que el intérprete de comandos ha creado la variable 1 con el valor que sigue inmediatamente al nombre del script, 2 con el valor que sigue inmediatamente a 1.

Ejercicio9: Crea un script, y hazlo ejecutable, que muestre en patalla 9 variables posicionales.

Ejercicio10: Ahora modifica el script del ejercicio anterior para que muestre 10 variables posicionales. ¿Es posible? ¿Por qué?

Vamos a presentar la sentencia shift. Crea el siguiente script:

```
echo $1
shift
echo $1
```

Lo haremos ejecutable y lo lanzaremos con ./parametros.sh hola adios.

Ejercicio11: antes de lanzarlo ¿Qué crees que debe salir? ¿Qué se muestra en pantalla exactamente?

shift desplaza hacia la derecha los parámetros posicionales. Por defecto el valor es 1, pero puede utilizarse con cualquier valor entero.

Existen otras variables posicionales:

$0 → es el nombre del script
$# → número de argumentos introducidos.
$* → todos los parámetros introducidos.
$? → código de error de la última instrucción ejecutada.

Ejercicio12: Crea un script que muestre en pantalla el nombre del script.
Ejercicio13: Crea un script que indique el número de argumentos que se han introducido.
Ejercicio14: Crea un script que muestre todos los parámetros introducidos.
Ejercicio15: Ejecuta lo siguiente:

```
grep "jajajajaj" parametros.sh
echo $?
```

¿Qué valor muestra?
Ejercicio16: Ahora ejecuta esto:

```
grep "echo" parametros.sh
echo $?
```

¿Qué valor se muestra ahora? ¿Podrías explicar por qué?

Variables

Dentro de un script podremos crear y utilizar variables. La manera de trabajar con las variables es exactamente igual que en la sesión Linux:

- Utilizaremos el operador = para asignarles valor.
- Para trabajar con el contenido de la variable: $variable.

Observa el script siguiente:

```
Variable1="Pedro"
echo $Variable1
```

En la primera línea hemos creado la variable Variable1 y le hemos asignado el valor Pedro.

¿Qué pasa con las variables al salir del script?

Ejercicio17: Crea el script anterior y ejecútalo. Una vez terminado el script ¿Qué ocurre al intentar mostrar el contenido de Variable1?

Pues pasa que las variables creadas en un script se "pierden" cuando se termina el script.

También podemos solicitar al usuario un dato y almacenarlo en una variable. Utilizamos la sentencia read:

```
echo "Introduzca el nombre"
read Nombre
echo "Su nombre es $Nombre"
```

Ejercicio18: Realiza un programa que pida el nombre del usuario, sus apellidos y su edad y las muestra, separadas por comas, en pantalla.

Ejercicio19: Crea el script sumad.sh. Se le pasarán dos variables posicionales y mostrará la suma.

Ejercicio20: Crea el script sumap.sh. Borrará la pantalla, pedirá dos números al usuario y mostrará su suma.