Scripts

Módulo profesional: Implantación de Sistemas Operativos

Ciclo Formativo: C.F.G.S. Administración de Sistemas Informáticos en Red

Curso: 1º

Profesor: Anabel Serradilla Fernández

A. INTRODUCCIÓN

Shell: intérprete de comandos. Programa capaz de traducir a un lenguaje que el ordenador entienda las órdenes que nosotros le damos.

Shell script: programas escritos para la shell de Unix/Linux

Batch files: programas escritos para la shell de Windows. Se guardan con extensión .bat. Al ejecutarse un fichero .bat, Windows automáticamente inicia la consola o interprete de comandos de MSDOS y ejecuta las instrucciones que contiene de forma secuencial.

La programación de shell scipts es muy útil para resolver tareas repetitivas, típicas de los administradores de sistemas. Son ficheros de texto que contienen comandos y son directamente ejecutables por el sistema.

B. SCRIPTS DE WINDOWS

En Windows existen diferentes intérpretes de comandos de forma nativa, entre ellos mencionar los más usados: **Visual Basic Script (VBS)**, **cmd** o símbolo del sistema y el más popular a día de hoy, **PowerShell**.

Para crear un archivo batch solo es necesario un editor de texto plano, que puede ser el el Bloc de notas de Windows, el Notepad++ o cualquier otro.

En un fichero batch se puede utilizar cualquier comando DOS.

Recordatorio de algunos de los principales comandos:

COMANDO	DESCRIPCION
CALL	Llama a otro fichero Batch
CD o CHDIR	Cambia de directorio
CLS	Borra la pantalla
COLOR	Configura los colores de primer y segundo plano de la consola
COPY	Copia uno o más archivos en otro destino
DATE	Muestra o establece la fecha del sistema
DEL	Elimina uno o varios archivos
DIR	Muestra la lista de subdirectorios y archivos de un directorio
ECHO	Visualiza en pantalla mensajes. También permite activar o desactivar el eco del comando
FOR	Repite un número determinado de veces un mismo proceso
GOTO	Salta y ejecuta una nueva línea indicada por una etiqueta
HELP	Proporciona información de ayuda para los comandos de Windows
IF	Se utiliza para saltos condicionales
MOVE	Mueve un archivo
PAUSE	Detiene momentáneamente la ejecución de un programa y muestra el mensaje: *Presione una tecla para continuar...*
REM	Introduce un comentario
SHIFT	;@ Evita que una línea aparezca por pantalla
TIME	Muestra o establece la hora del sistema

Lista completa de comandos de Windows: https://ss64.com/nt/
https://www.abrirllave.com/cmd/comandos.php

Y tres elementos adicionales que utilizaremos en los ficheros batch:

COMANDO	DESCRIPCION
:ETIQ	Identifica una posición de salto
%NUM	Introduce parámetros en el fichero
%VAR%	Hace referencia a una variable

La estructura de un archivo batch es sencilla:

- Una primera línea: @echo off (no es imprescindible para que funcione)
- Las instrucciones, que pueden constar de una simple línea o de varias, Todas se irán ejecutando en su orden.
- Al final generalmente se emplea EXIT o EOF (End Of File) que cierra la ventana de la consola.

Ejemplo de script sencillo

```
@echo off
title Mi primer 'batch'
echo Hola Mundo
pause
exit
```

Otro ejemplo:

```
@echo off
echo Hola, Cuando pulses la tecla se borrara el contenido y cambiara por otro.
pause
cls
echo ¿Ves?
pause
exit
```

Función echo

La función echo activa o desactiva la característica de repetición de comandos. Si se usa sin parámetros, echo muestra el valor de echo actual.
Probar el fichero de Hola mundo con @echo=on

- echo on: activa la repetición de comandos
- echo off: desactiva la repetición de comandos

- @ delante de un comando: evita que se repita un comando determinado en un archivo por lotes
- @echo off al principio del archivo: evita que se repitan todos los comandos del archivo.

Variables de entorno

Las variables de entorno son rutas o cadenas que definen aspectos del entorno del sistema para el usuario que en cada momento tiene la sesión iniciada.

Ejemplos:

- %PROGRAMFILES% - Remite a la carpeta donde se instalan los programas. En Windows 10 la ruta por defecto es "C:\Program Files"
- %WINDIR% - Remite a la carpeta donde se instala Windows. En Windows 10 la ruta por defecto es "C:\Windows"
- %TIME% - Devuelve la hora actual del sistema
- %PATH% - Se guardan las rutas donde Windows debe buscar comandos o programas a ejecutar
- %PATHEXT% - Devuelve la lista de extensiones de archivo consideradas ejecutables
- %PROMPT% - Permite configurar el símbolo de sistema que indica al usuario que puede introducir texto
- %HOMEPATH% - Muestra la ruta principal del usuario actual

Se pueden ver todas las variables de entorno con el comando ***set***.

Comentarios

Para poner comentarios, se utiliza la orden rem.
Ejemplo:

```
rem Esto es un comentario. El intérprete lo ignora.
```

Ayuda

Para buscar ayuda acerca de un comando concreto se puede utilizar la siguiente sintaxis:

```
color /?
color ?
color help
help color
```

Variables

Para establecer el valor de una variable se utiliza **set.**

```
set ruta=\\192.168.1.2\compartida
@echo La ruta es %ruta%
@echo Las variables actuales son:
set
```

Para mostrar el contenido de una variable se pone % delante y detrás del nombre de la variable.

Para leer por teclado un valor y asignarlo a una variable se utiliza **set /p**.

Para realizar operaciones matemáticas se utiliza set /a.

```
@echo off
set anoactual=2021
set /p anonac=¿En qué año naciste? (4 dígitos)
set /a edad=anoactual-anonac-1
echo Si aún no has hecho los años, tienes %edad% años
```

Ejercicio 1: introducir dos números por pantalla y devolver su suma

Argumentos

A veces es necesario ejecutar un script pasándole valores de inicio. Estas variables que se pasan al script en el momento de su llamada se denominan argumentos. Por ejemplo, podemos crear un script de copia de seguridad que reciba como parámetros la carpeta origen y la carpeta destino.

La forma de utilizar los argumentos en la llamada es:
nombre_script argumento1 argumento2 argumento3

siendo:
%0 = nombre del archivo .bat que se está ejecutando
%1 = argumento1
%2 = argumento2
%3 = argumento3

Ejercicio 2: ejecutar un programa bat que recibe dos números y devuelve su suma

Estructuras de control

Etiquetas Goto

Las etiquetas son marcadores que indican un punto concreto del programa. Se definen con un texto precedido del carácter :
A esa etiqueta se puede hacer referencia después con el comando Goto

Ejemplo:

```
@echo off
:etiqueta
echo Hola de nuevo ;)
pause>nul
goto etiqueta
```

Nota: pause>nul redirige la salida del comando pause a un fichero especial denominado null que descarta todos los datos escritos en él.

Es posible llamar con el comando goto a una etiqueta de tres formas distintas:

```
goto label
goto :label
goto:label
```

Condicional if

Para realizar una sentencia de control if se utiliza la sintaxis:
if {not}%variable%==X (una acción) else (otra acción)

Ejemplo:

```
@echo off
title Preguntita

:inicio
set /p cuenta=Cuanto es 2+2 ?...

if %cuenta%==4 (goto bien) else (goto mal)

:bien
echo El resultado es correcto ;P
pause>nul
exit

:mal
echo El resultado es incorrecto. Vuelve a intentarlo.
goto inicio
```

if no permite usar más de un comando por cada condición. Si es necesario utilizar más de uno se hará utilizando &&

Ejemplo:

```
if exist %homedrive%\fichero.txt
(echo Se ha encontrado el fichero. Presione una tecla para eliminar el archivo)
else
(echo Fichero no encontrado &&  pause>nul && goto EOF)
pause>nul

del /f /s /q %homedrive%\fichero.txt
echo Listo
pause>nul
```

Nota: EOF (End Of File). Se va al final del fichero sin necesidad de definir una etiqueta.

Se pueden realizar comparaciones lógicas con la sintaxis:
if [/i] cadena1 opción cadena2 <comando>
donde cadena1, cadena2 pueden ser número, palabra, variable o caracteres
opción presenta las siguientes opciones:

EQU: Igual (=) NEQ: No igual (≠)	LSS: Menor que (<) LEQ: Menor o igual (≤)	GTR: Mayor que (>) GEQ: Mayor o igual (≥)

El parámetro opcional /i indica que para las comparaciones no habrá distinción entre mayúsculas y minúsculas. Por defecto if sí distingue.

Ejercicio 3: ejecutar un programa bat que pida por pantalla la edad del usuario.
Si edad < 18 se escribe un mensaje por pantalla indicando que es menor de edad
Si no se escribe un mensaje por pantalla indicando que es mayor de edad.

Ejercicio 4: ejecutar un programa bat que genere un número aleatorio del 1 al 10. El usuario debe adivinar ese número indicando números por pantalla hasta que lo acierte.

Bucle for

Para realizar bucles for se utiliza la sintaxis:
for {%% | %}<variable> in (lista) do <comando> [<opciones_comando>]
%: para llevar a cabo el bucle for en el símbolo del sistema
%%: para llevar a cabo el bucle for en un archivo por lotes

variable: nombre de un único carácter si es dentro de un archivo batch. Distingue mayúsculas y minúsculas.
lista: conjunto de elementos archivos separados por coma, punto y coma o espacio en blanco. Si en la lista aparecen comodines (? o *) el comando se ejecutará para cada uno de los archivos que cumplan la expresión regular.

```
@echo off
for %%i in ( 0 1 2 3 4 5 ) DO echo Hola no. %%i
pause
```

```
@echo off
echo Hacer ping un sitio web:
FOR %%a IN (www.google.com www.facebook.com www.twitter.com ) DO (
 echo Sitio [ %%a ] :
 ping %%a
 )
```

Mejoras en el bucle for:

FOR /r [ruta] %V IN (lista) DO comando	Recursividad
FOR /d %V IN (lista) DO comando	Directorios
FOR /l %V IN (inicio, incremento, fin) DO comando	Lista con contador

For %%dia in (lunes, martes, miércoles) do mkdir %dia%

Ejemplo:

```
REM mostrar números impares del 1 al 9 en pantalla
@echo off
set inicio=1
set final=9
set salto=2
echo Desde %inicio% hasta %final% de %salto% en %salto%
for /L %%x IN (%inicio%,%salto%,%final%) DO @echo %%x
```

Otro ejemplo:

```
REM Lanza ping a la red 192.168.1.1-254 y espera 30 ms.
REM Si luego ejecutas arp -a verás las MAC de la red.
FOR /L %%x IN (1,1,254) DO ping 192.168.1.%%x -n 1 -w 30
```

Realizar cálculos y usar funciones

Las funciones se definen del mismo modo que una etiqueta

```
@echo off
rem Funciones en Batch

call :decir “Hola mundo”
call :decir “Esto es una prueba”
call :decir palabra_suelta

goto :fin

:decir
echo.
echo %~1
goto:EOF

:fin
pause
exit
```

La diferencia entre %~1 y %1 es que el primero quita las comillas y el segundo las mantiene.
echo. añade una línea en blanco

Si la función devuelve algún valor debemos utilizar el comando set:

```
REM Funcion Area
@echo --------------------------
@echo off
set x=2
set y=5
call :Area %x% %y% result
echo/El area es: %result%
pause
goto :EOF

:Area %width% %height% result
set /a %3=%1*%2
goto :EOF
```

Varios

- Para ejecutar varios comandos en una sola línea se pone &&.
  Ejemplo:
  ```
  c: && cd \backups
  ```
  lo que nos coloca en el directorio c:\backups

- Para volcar la salida de un comando a un archivo se utiliza el símbolo '>'. Esto creará un nuevo archivo. Si el archivo existe, lo sobreescribe. Si queremos añadir algo a un archivo existente, utilizaremos '>>'

  Ejemplos:
  ```
  echo Creando un Archivo de texto > nuevo.txt
  echo Esta es la Segunda linea >> nuevo.txt
  ```

  Otro ejemplo:
  ```
  c: && cd \backups && dir *.* /s > archivos.txt
  dir d:\*.* /b >> archivos.txt
  ```

- Para pasar el contenido de un archivo de texto a un comando puedes usar:
  ```
  comando < archivo.txt
  ```

- Para mostrar el contenido de un archivo de texto en pantalla usa:
  ```
  TYPE texto.txt
  ```

- Cuando haya una cadena de caracteres que tenga espacios tienes que utilizar las comillas. En las rutas a los archivos también es recomendado utilizarlas.

- Para crear un mensaje en un batch, es decir una ventana que muestra una indicación o comentario al usuario, escribe en el batch lo siguiente:
  ```
  msg * Este Es Mi Mensaje
  ```