CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERÍAS (CUCEI)

Departamento de ciencias computacionales

Seminario de solución de problemas de uso, adaptación, explotación de sistemas operativos

Violeta del Rocío Becerra Velázquez

Jose Pedro Reyes Alvarez

222790897

Ingeniería Informática (INNI)

D02

3.1 Scripts para optimizar

11 de mayo del 2025

Índice

Explicación del problema

En muchas ocasiones, los usuarios olvidan hacer respaldos regulares de sus archivos importantes, lo cual puede llevar a pérdidas de información valiosa ante fallas del sistema, virus o extravío de la computadora. Aunque existen soluciones avanzadas de respaldo, muchas requieren instalación de programas o conocimientos técnicos.
El problema planteado es crear una forma automática y sencilla de hacer respaldos de archivos desde una carpeta del usuario hacia una memoria USB, sin necesidad de instalar software adicional.

Solución propuesta

Se propone el desarrollo de un archivo Batch (.bat), que sea ejecutable en sistemas Windows, y que permita:

Detectar si una memoria USB está conectada.

Copiar automáticamente los archivos desde una carpeta específica del usuario hacia la USB.

Crear un archivo de log con el historial del respaldo.

Verificar si los archivos se copiaron correctamente.

Esta solución es portátil, ligera y fácil de usar, incluso sin conocimientos de programación.

Explicación del código

bat

CopyEdit

@echo off

title Respaldo Automático USB

color 0A

- Se oculta la ejecución de comandos (@echo off) y se personaliza el título y color de la ventana.

bat

CopyEdit

:: Ruta de origen

set SOURCE_FOLDER=C:\Users\josep\Documents\RespaldoPrueba

- Define la carpeta desde donde se copiarán los archivos.

bat

CopyEdit

:: Ruta de destino (USB)

set USB_DRIVE=E:

- Define la letra donde se espera que esté la memoria USB (se puede cambiar manualmente si es otra).

bat

CopyEdit

:: Verificar si la USB existe

if exist %USB_DRIVE%\ (

  echo Unidad USB detectada...

) else (

  echo No se detectó ninguna USB...

  pause

  exit /b

)

- Se comprueba si la memoria USB está disponible en la letra indicada.

bat

CopyEdit

:: Crear carpeta de respaldo dentro de la USB

set BACKUP_FOLDER=%USB_DRIVE%\Respaldo_josep

mkdir "%BACKUP_FOLDER%"

- Crea una carpeta en la USB para almacenar los archivos respaldados.

bat

CopyEdit

:: Copiar archivos de forma recursiva (/E) incluyendo archivos ocultos (/H) y sobrescribiendo sin preguntar (/Y)

xcopy "%SOURCE_FOLDER%*.*" "%BACKUP_FOLDER%\" /E /H /Y

- Copia todos los archivos y subcarpetas desde la carpeta origen hacia la USB.

bat

CopyEdit

:: Crear archivo de log

set LOG_FILE=%BACKUP_FOLDER%\log_respaldo.txt

echo Respaldo realizado el %DATE% a las %TIME% > "%LOG_FILE%"

echo Archivos respaldados desde: %SOURCE_FOLDER% >> "%LOG_FILE%"

dir /b "%BACKUP_FOLDER%" >> "%LOG_FILE%"

- Genera un archivo de texto que guarda el historial del respaldo: fecha, hora y lista de archivos.

bat

CopyEdit

:: Verificación final

if exist "%BACKUP_FOLDER%*" (

  echo Archivos respaldados correctamente.

) else (

  echo No se encontraron archivos copiados.

)

- Comprueba si efectivamente se copiaron archivos a la memoria USB.

Comandos utilizados en el script

Comando	**Descripción**
@echo off	Oculta la visualización de los comandos en la consola al ejecutarse el script.
title	Cambia el título de la ventana de la consola.
color	Cambia el color del texto y del fondo en la consola (ej. 0A = fondo negro, texto verde).
set	Define una variable de entorno (como rutas o nombres de carpeta).
if exist	Verifica si existe un archivo o carpeta (o una unidad USB en este caso).

echo	Muestra mensajes en la consola o escribe contenido en un archivo.
pause	Detiene la ejecución hasta que el usuario presione una tecla.
exit /b	Sale del script actual (sin cerrar toda la consola si se ejecuta dentro de otra).
mkdir	Crea una nueva carpeta en la ubicación especificada.
xcopy	Copia archivos y carpetas de manera avanzada (mejor que copy).
/E (con xcopy)	Copia subdirectorios, incluso si están vacíos.
/H (con xcopy)	Copia también archivos ocultos y de sistema.
/Y (con xcopy)	Evita que se pida confirmación para sobrescribir archivos.
dir /b	Lista los nombres de los archivos en formato simple (solo nombres, sin detalles).
>	Redirige la salida de un comando a un archivo nuevo (sobrescribe si existe).
>>	Redirige la salida y **agrega** contenido al final de un archivo existente.

Conclusión

Con esta actividad logré entender de una mejor manera lo que es un script y lo que se puede llegar a hacer con él. En clases anteriores, como en Administración de Servidores, habíamos visto este tema de forma teórica, pero no desarrollamos algo práctico que nos permitiera ver realmente su potencial. Gracias a esta actividad, no solo comprendí cómo funcionan los scripts en Windows, sino también cómo pueden ayudarnos a automatizar tareas repetitivas, ahorrar tiempo y minimizar errores humanos.

El hecho de crear un script que detecte una USB, copie archivos importantes y genere un respaldo con log incluido, me permitió ver que con conocimientos básicos se pueden lograr soluciones útiles para problemas reales. También entendí mejor los comandos de Batch y su lógica de ejecución, lo cual me servirá mucho en futuros proyectos o incluso en ambientes laborales. En general, esta actividad me ayudó a reforzar lo aprendido y a darle un sentido más práctico al uso de los scripts dentro del sistema operativo.

SCRIPT

¿Qué es un Script?

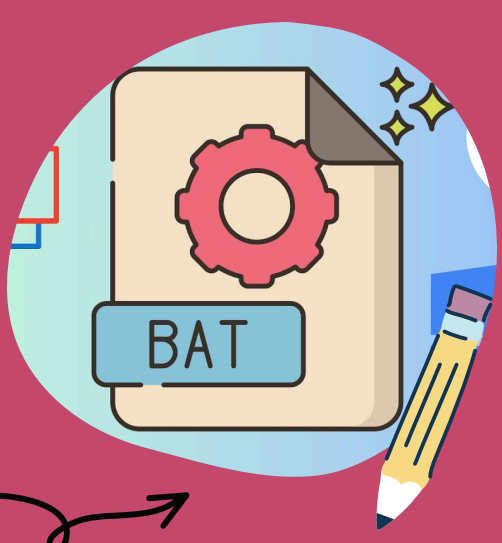
Un script es un archivo de texto que contiene una secuencia de instrucciones que un sistema operativo puede ejecutar de forma automática. Se usa principalmente para automatizar tareas y optimizar procesos del sistema.

¿Cómo Funciona un Script?

- Interpretado línea por línea: No se compila, sino que el sistema lee y ejecuta cada línea en orden.
- Automatiza tareas repetitivas: Como limpieza de archivos, respaldos, instalación de software, entre otros.
- Interactúa con el sistema: Controla procesos, gestiona archivos y manipula variables del entorno.

Elementos de un Script

- Shebang (#!): Indica el intérprete (ej. #!/bin/bash). Solo en Unix/Linux.
- Comandos: Instrucciones del sistema (ej. mkdir, echo, rm).
- Variables: Guardan datos reutilizables. Ejemplo: NOMBRE="Juan".



- Condicionales: Permiten decisiones lógicas. Ejemplo: if, else.
- Bucles: Ejecutan instrucciones múltiples veces (for, while).
- Comentarios: Líneas explicativas que no se ejecutan (# comentario).

Beneficios

- Ahorro de tiempo
- Reducción de errores humanos
- Tareas automáticas sin intervención
- Mayor eficiencia del sistema