Curso de PowerShell gratuito (enero 2025)

1. Las bases de PowerShell

El vínculo entre PowerShell y .NET

PowerShell está profundamente integrado con .NET, lo que lo convierte en una herramienta extremadamente poderosa para interactuar con el sistema operativo y otros recursos. Esta integración permite acceder a clases, métodos y propiedades de .NET directamente desde PowerShell, ofreciendo un control avanzado sobre diversas operaciones.

Por ejemplo, se puede utilizar la clase DateTime de .NET para obtener información sobre la fecha y hora actuales:
[crayon-6a9d4f057062a434085694/]
También puedes usar objetos más avanzados como System.IO para trabajar con archivos:
[crayon-6a9d4f057062f978537833/]

Cmdlets: principio y uso

Los cmdlets son comandos predefinidos diseñados para realizar tareas específicas. Están escritos en .NET y ofrecen una sintaxis simple para interactuar con el sistema. Los cmdlets tienen un formato estándar de verbo-sustantivo, como Get-Process (obtener procesos) o Set-Item (configurar un elemento).
[crayon-6a9d4f057063112238433 3/]

Tuberías

La tubería en PowerShell permite encadenar cmdlets, de forma que la salida de un comando se convierte en la entrada de

otro. Esto simplifica la creación de flujos de trabajo complejos.
[crayon-6a9d4f0570633102591074/]
La variable $PipelineVariable se puede usar para almacenar temporalmente datos intermedios dentro de la tubería:
[crayon-6a9d4f0570634981645241/]

Filtros y bucles

PowerShell incluye filtros como Where-Object para evaluar condiciones y bucles como ForEach-Object para iterar sobre colecciones.
[crayon-6a9d4f0570636223175253/]

Windows ISE

El entorno integrado de scripts (ISE) de PowerShell es una herramienta poderosa para escribir, depurar y ejecutar scripts. Ofrece características como:

- Coloreado de sintaxis.
- Finalización automática de comandos.
- Ventanas múltiples para trabajar con diferentes scripts a la vez.

Puedes abrir el ISE ejecutando:
[crayon-6a9d4f0570637780846386/]

2. Tipos y operadores

Sistema de clasificación

En PowerShell, las variables pueden almacenar datos de diferentes tipos sin necesidad de declarar el tipo explícitamente. Sin embargo, puedes definir el tipo si lo deseas:
[crayon-6a9d4f0570639121969070/]
El sistema de clasificación incluye tipos como:

- [string]: Cadenas de texto.
- [int]: Números enteros.
- [datetime]: Fechas y horas.
- [array]: Listas o colecciones.

Expresiones regulares

PowerShell soporta expresiones regulares para buscar patrones en texto. Ejemplo:
[crayon-6a9d4f057063b988221242/]
Usar expresiones regulares para reemplazar texto:
[crayon-6a9d4f057063c740273655/]

Operadores

PowerShell incluye diversos operadores para realizar operaciones matemáticas, comparaciones y manipulaciones de texto:

- Aritméticos: +, -, *, /
- Comparación: -eq, -ne, -gt, -lt, -like
- Lógicos: -and, -or, -not

[crayon-6a9d4f057063e502095967/]

3. Estructuras de control y funciones

Estructuras de control

Las estructuras de control en PowerShell permiten gestionar el flujo del script según condiciones:
[crayon-6a9d4f057063f612142297/]

Funciones

Las funciones en PowerShell encapsulan lógica reutilizable:
[crayon-6a9d4f0570641936272435/]

4. Utilización de cmdlets y módulos

Gestión de archivos

PowerShell facilita la gestión de archivos mediante cmdlets especializados como Compress-Archive y Expand-Archive, que permiten comprimir y descomprimir archivos directamente.
[crayon-6a9d4f0570642593518228/]

Cmdlets para web

Cmdlets como Invoke-WebRequest e Invoke-RestMethod son ideales para interactuar con servicios web, descargar archivos y trabajar con APIs REST.
[crayon-6a9d4f0570644328511307/]

Ejemplo práctico

Un ejemplo práctico de recuperación de un canal RSS:
[crayon-6a9d4f0570645746372394/]

5. Utilización de objetos CIM

Métodos y propiedades WMI

PowerShell permite interactuar con objetos CIM (Common Information Model) para obtener información y realizar configuraciones en el sistema operativo. Por ejemplo, puedes usar Get-CimInstance para consultar clases WMI:
[crayon-6a9d4f0570647242697858/]
Comparar Get-CimInstance con el cmdlet más antiguo Get-WmiObject:
[crayon-6a9d4f0570648195954071/]

6. Uso de .NET y COM

Clases de .NET

PowerShell permite utilizar clases .NET directamente para realizar operaciones avanzadas, como pings y autenticación.
[crayon-6a9d4f0570649755992919/]

Interfaces gráficas

PowerShell puede usar XAML para crear interfaces gráficas simples. Ejemplo básico:
[crayon-6a9d4f057064b705855227/]

Ejemplo práctico

Crear un formulario utilizando XAML:
[crayon-6a9d4f057064c665772812/]

7. Gestión de módulos y paquetes de PowerShell

Funcionamiento de módulos

Los módulos en PowerShell son colecciones de cmdlets, funciones, variables y recursos que permiten extender sus capacidades. Para trabajar con módulos:
[crayon-6a9d4f057064e086290108/]

Ejemplo práctico

Crear un usuario en Active Directory:
[crayon-6a9d4f057065066226721/]

8. Los objetos COM

Introducción a los objetos COM

Los objetos COM (Component Object Model) permiten interactuar con aplicaciones de Windows como Excel, Word y PowerPoint. Ejemplo de creación y manipulación de un libro de Excel:
[crayon-6a9d4f0570651399594701/]

Ejemplo práctico

Recuperar datos de servidores y exportarlos a Excel:
[crayon-6a9d4f0570653996980401/]

9. Algunas cosas más de PowerShell

Cmdlets Útiles

PowerShell tiene varios cmdlets menos conocidos pero muy útiles:
[crayon-6a9d4f0570654818150914/]

Ejemplo práctico

Crear un generador de contraseñas:
[crayon-6a9d4f0570656188338341/]