TEMA 04 – WINDOWS. ADMINISTRACIÓN Y CONFIGURACIÓN – SCRIPTS

Introducción

* Órdenes de Powershell: cmdlets (command-let)
* Al principio: 129 cmdlets básicos
* Pueden incluirse conjuntos específicos para trabajar con Active Directory, Exchange u otros roles de servidor
* 2 opciones:
  + PowerShell ISE (Integrated Scripting Environment)
  + Consola por línea de commandos
* Version actual: 5.1

Commandos y cmdlets

* Fáciles de recorder
* Convención de nomenclatura “verbo-sustantivo”
* Suele indicar claramente su función

Redirecciones y tuberías

* Obtener la salida de una acción y dirigirla a un lugar diferente del predeterminado
* Orientadas a la obtención de un archivo de texto con la salida que ofrece un cmdlet
  + >: nuevo archivo y deposita la salida. Si existe, borra el contenido previo
  + >>: lo añade al final del documento. Si no existe, lo crea
  + |: conecta la salida de un cmdlet con la entrada de otro (n veces)

Primeros pasos con PowerShell

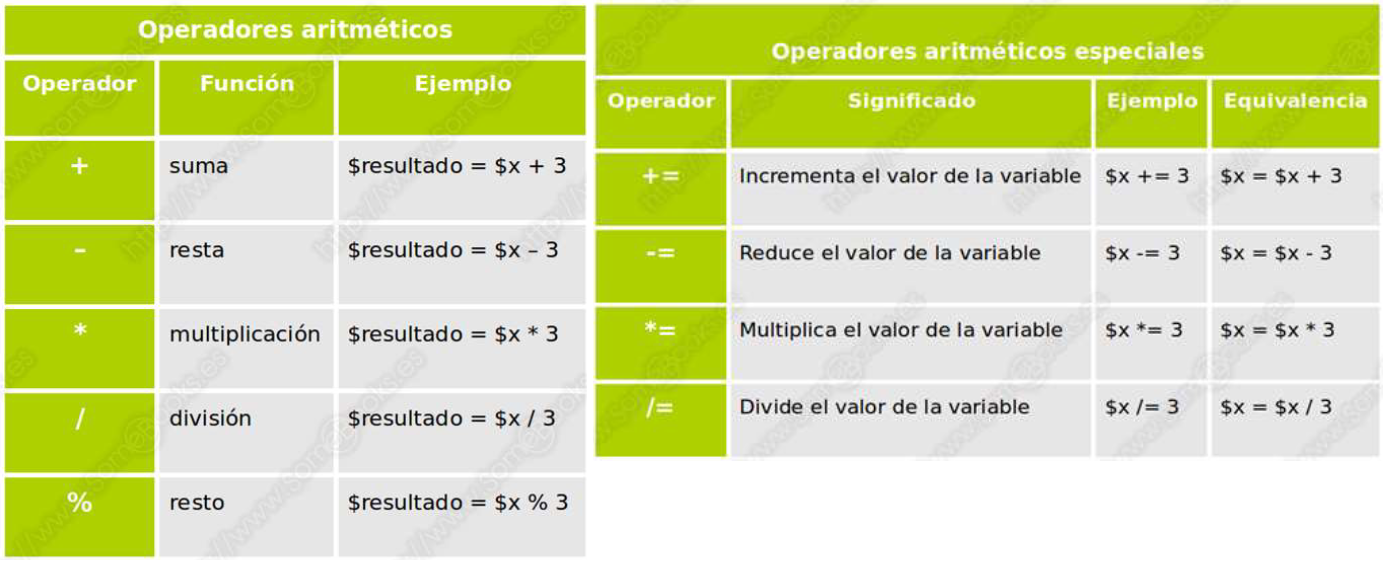
* Niveles de permisibilidad:
  + Restricted: no se permite la ejecución de scripts
  + AllSigned: deberán estar autenticados antes de ejecutarlos, es la más restrictiva
  + RemoteSigned: deberán estar autenticados los scripts que procedan de una ubi remota (p.e.: descargados)
  + Unrestricted: se ejecutará cualquier script
* Con “Get-ExecutionPolicy”
* Para establecerla: “Set-ExecutionPolicy”
* Ejecutar script: “.\script.ps1” o “ruta-absoluta\script.ps1

Variables en PowerShell

* Solamente hay que llamarla
* Restricciones:
  + Primer carácter: $
  + Letras, números o símbolos
  + Espacios en blancos: hay qe meter el nombre entre llaves {}
* También se puede con New-Variable Variable (no hace falta el $)
  + -Name (por si metemos más opciones, también se puede a secas si solo este)
  + -Value
  + -Opcion: readonly…
* Get-Variable: lista completa de las variables que se hayan definido hasta el momento
* Pedir información: $Nombre = Read-Host “¿Cómo te llamas?”
* Mostrar información: Write-Host “Bienvenido” / Write-Host $Nombre
* Tipos de datos: [string] [char] [long] [double] [decima] [datetime] [bool] {array]
* Tipo de dato de una variable: Write-Host $Nombre.GetType().Name
* Rango de valores
  + [decimal]::MinValue
  + [decimal]::MaxValue
* Establecer tipo de dato: [float] $precio = 4.99 / $precio = [float] 4.99
  + Write-Host ([int]$precio)

Observaciones básicas con variables en PowerShell

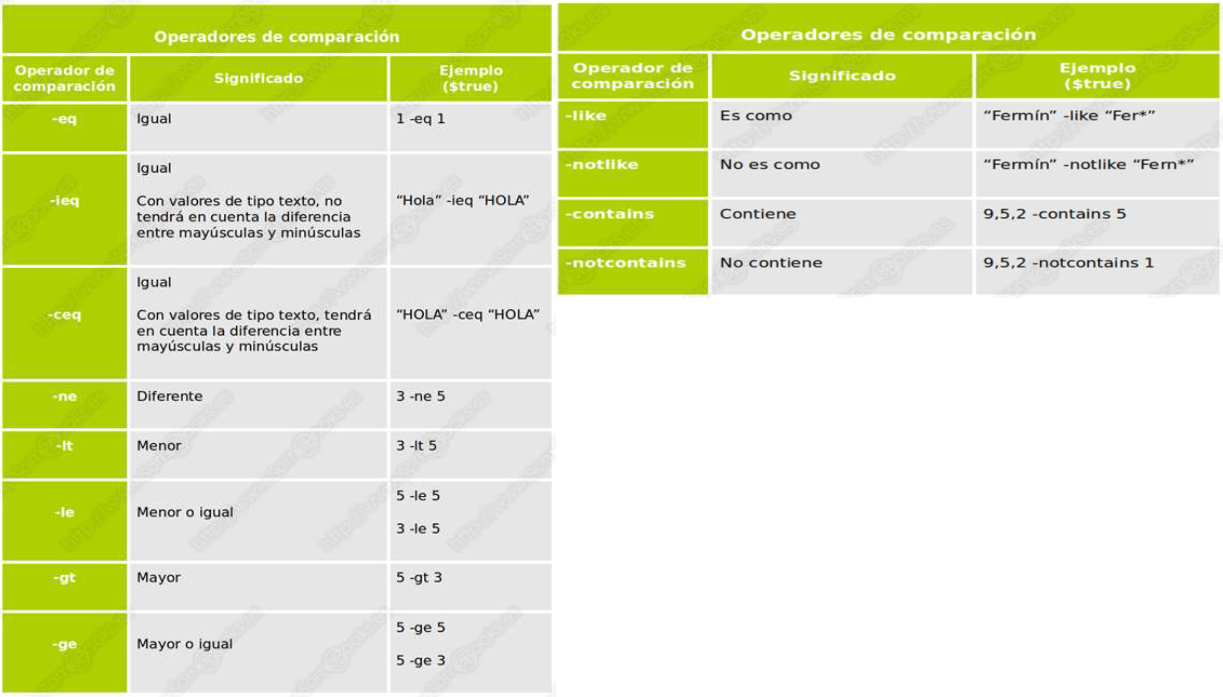
* Operar con variables de tipo texto
  + $mensaje = $saludo + “ “ + $nombre / $mensaje = “$saludo $nombre”
* Operadores aritméticos

****

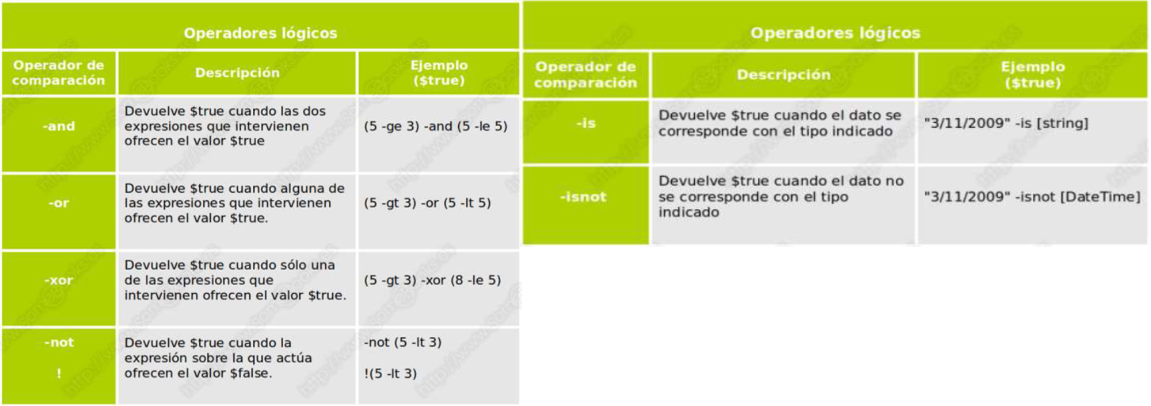
**A green and white box with white text

Description automatically generated**

* **Operadores de comparación**

****

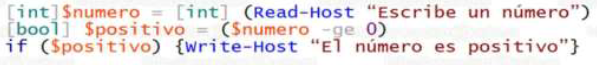
* **Operadores lógicos**



* Variables con objetos
  + PowerShell está basado en .NET Framework
    - Nuestros scripts tendrán acceso a todo el modelo de objetos subyacentes en .NET Framework
    - Cada variable es realmente un objeto de .NET Framework
    - Tendrá asociado una serie de métodos
      * Posición: var.indexOf(“o”)
      * A minúsculas: var.toLower()

Estructuras condicionales en PowerShell

* If



* If else

A close up of a sign

Description automatically generated

* If elseif

A screenshot of a computer code

Description automatically generated

* Switch

A close-up of a computer code

Description automatically generated

Estructuras repetitivas en PowerShell

* While

A close up of text

Description automatically generated

* Do while

A close up of a text

Description automatically generated

* Do until

A close up of text

Description automatically generated

* For

A close up of a sign

Description automatically generated

* Foreach

A close up of words

Description automatically generated

* Alterar el proceso de las estructuras repetitivas
  + Break

A close-up of a computer code

Description automatically generated

* + Continue

A close up of text

Description automatically generated

LISTADO CMDLETS

Gestión de archivos y directorios

* Get-ChildItem: lista los archivos y directorios (ls) (Get-ChildItem C:\Documents)
* Set-Location: cambia el Directorio actual (cd)
* Copy-Item: copia archivos o directorios (cp) (Copy-Item C:\example.txt – Destination D:)
* Move-Item: mueve archivos o directorios (mv) (Copy-Item C:\example.txt – Destination D:)
* Remove-Item: elimina archivos o directorios (Remove-Item C:\example.txt)
* New-Item: crea un nuevo archivo o directorio (New-Item C:\newfolder – ItemType Directory)

Obtener información y contenido

* Get-Content: lee el contenido de un archivo (cat) (Get-Content C:\example.txt)
* Get-History: muestra el historial de comandos
* Get-Process: lista todos los procesos en ejecución
* Get-Service: lista todos los servicios del sistema

Administración y control del sistema

* Start-Service: inicia un servidor (Start-Service -Name “wuauserv”)
* Stop-Service: detiene un servicio (Stop-Service -Name “wuauserv”)
* Restart-Service: reinicia un servidor (Restart-Service -Name “wuauserv”)
* Start-Process: inicia un proceso o aplicación (Start-Process Notepad)
* Stop-Process: termina un proceso (Stop-Process -Name “notepad”)
* Restart-Computer: reinicia la computadora
* Shutdown-Computer: apaga la computadora

Trabajo con redes

* Test-Connection: prueba la conectividad a un host (Test-Connection google.com)
* Get-NetIPAddress: muestra la información sobre las configuraciones de IP
* Resolve-DnsName: resuelve un nombre a IP (Resolve-DnsName google.com)

Automatización y Scripting

* ForEach-Object: ejecuta acciones en múltiples objetos (Get-Process | FirEach-Object {$\_.ProcessName})
* Where-Object: filtra objetos (Get-Process | Where-Object {$\_.CPU -gt 10})
* Invoke-Command: ejecuta commandos en máquinas remotas (Invoke-Command -ComputerName Server01 -ScriptBlock {Get-Childre C:\})

Ayuda y descubrimiento

* Get-Help: muestra ayuda (Get-Help Get-Process)
  + -Online: busca la info en la página de Microsoft
* Get-Command: lista cmdlets y funciones
* Show-Command: muestra una interfaz gráfica para cmdlets
* Update-Help: instala los archivos de ayuda de forma local

Variedades y utilidades

* Set-Variable: establece una variable (Set-Variable -Name “x” -Value 10)
* Get-Variable: muestra variables y valores
* ConvertTo-Json: convierte a formato JSON (Get-Process | ConvertTo-Json)