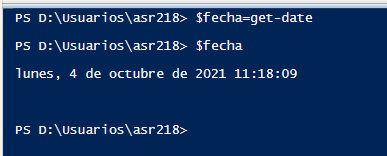
|  |  |  |
| --- | --- | --- |
| Administración de Sistemas Operativos | | |
| Tema 3-2 | **Aplicación de scripting en SO**  **Introducción a PowerShell**  **Variables** | |
| Fecha |  | |
| Nombre y Apellidos |  | Nº |
| http://freyes.svetlian.com/Los%20Arrays%20en%20Powershell.htm  <https://docs.microsoft.com/es-es/powershell/scripting/getting-started/cookbooks/working-with-files-and-folders?view=powershell-5.1>  https://fvarrui.github.io/ADD/docs/scripting/powershell | | |

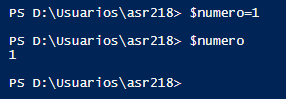
El objetivo es *manejar variables con PowerShell*. Las variables comienzan con $y podemos meter en ella cualquier casi cualquier cosa.

1. Prueba los siguientes ejemplos:

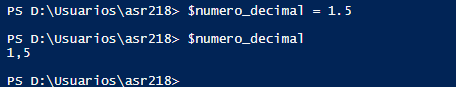
PS>$fecha = get-date



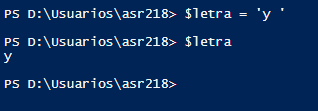
PS>$numero = 1



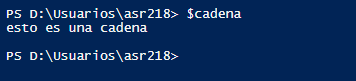
PS>$numero\_decimal = 1.5



PS>$letra = 'y '

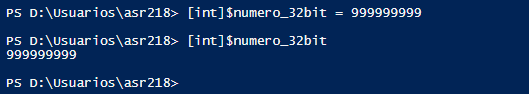


PS>$cadena = 'esto es una cadena'

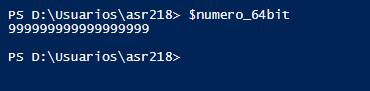


Puedes asignar el tipo concreto de variable indicándolo entre corchetes. Estos son algunos:

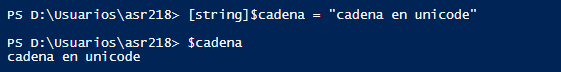
[int]$numero\_32bit = 999999999



[long]$numero\_64bit = 999999999999999999



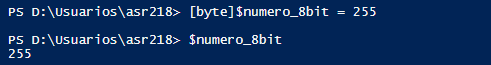
[string]$cadena = "cadena en unicode"



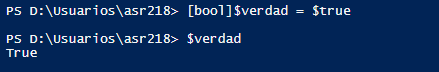
[char]$caracter = "a"



[byte]$numero\_8bit = 255



[bool]$verdad = $true

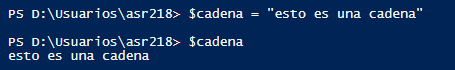


[decimal]$numero\_decimal = 467.45



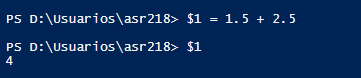
Para mostrar el valor de una variable, simplemente escríbela y pulsa ENTER

PS>$cadenaesto es una cadena



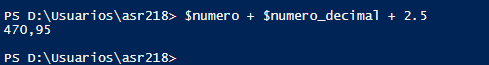
Puedes sumar dos o más números:

PS>1 + 1.5 2.5



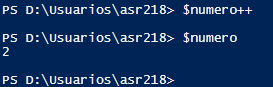
o sumar dos o más variables que contienen números:

PS>$numero + $numero\_decimal + 2.5



o incrementar directamente $numero en una unidad:

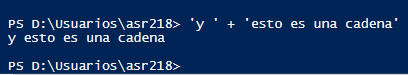
PS>$numero++



PS>$numero2 Puedes unir dos o más cadenas:

PS>'y ' + 'esto es una cadena'

y esto es una cadena



Observa que los espacios en blanco se tienen en cuenta.

Para unir dos o más variables que contienen cadenas:

PS>$letra + $cadena



y esto es una cadena.

1. Crea las siguientes variables, usando distintas maneras posibles y asignándolas los valores siguientes:

$unaCadena ="Algún texto"

$unNumero = 5 -as [double]

$unaLista = 1,2,3,4,5

$name = -join("Jo", "h", "n");

Write-Host (-join("Jo", "h", "n"))

$unaCadena 1= $unaLista -join ‘.’

$unaTablaHash = @{nom1='val1'; nom2='val2'}

[string]$cadena = $( Read-Host "Introduce un parámetro:”)

$archivo="C:\Users\usuario\Desktop\textos\texto.txt

$names = get-content $archivo

[string]$c = Read-Host "Introduce un nombre"

Set-Variable -name dia -value Domingo -description “dia festivo”

$fecha=(get-date)

1. Recupera el valor de las variables, para comprobar su definición:

echo $unaCadena

echo "Interpolación: $unaCadena"

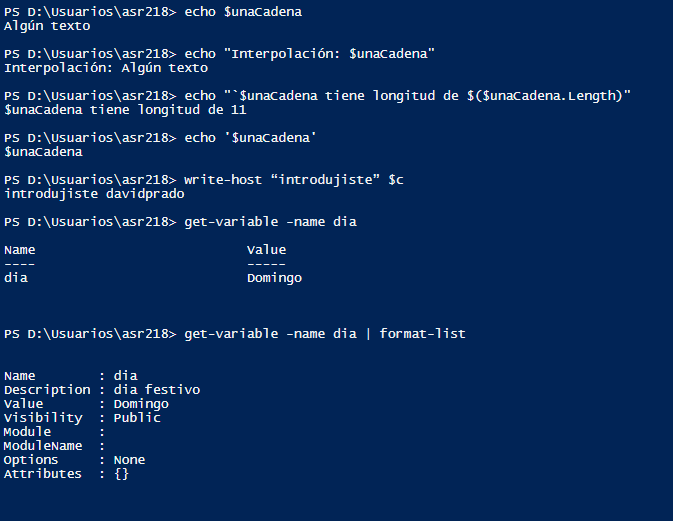
echo "`$unaCadena tiene longitud de $($unaCadena.Length)"

echo '$unaCadena'

write-host “introdujiste” $c

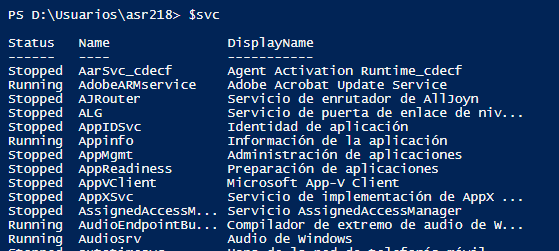
get-variable -name dia

get-variable -name dia | format-list



1. Con el comando get-member vas a ver las distintas propiedades de las variables, comprueba que propiedades se muestran en los siguientes casos:

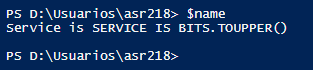
$svc = Get-Service



$svc[0].name



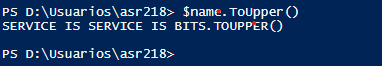
$name = $svc[1].name



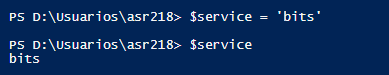
$name.length



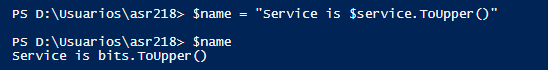
$name.ToUpper()



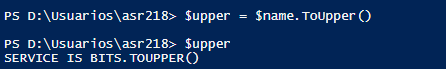
$service = 'bits'



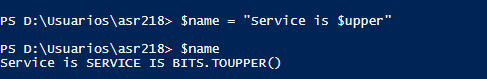
$name = "Service is $service.ToUpper()"



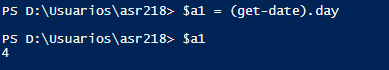
$upper = $name.ToUpper()



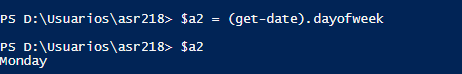
$name = "Service is $upper"



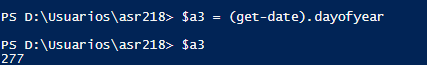
#$a1 = (get-date).day



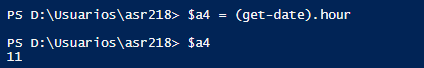
#$a2 = (get-date).dayofweek



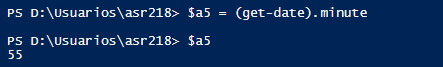
#$a3 = (get-date).dayofyear



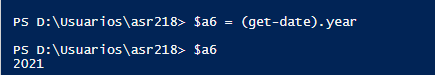
#$a4 = (get-date).hour



#$a5 = (get-date).minute



#$a6 = (get-date).year



1. Variables predefinidas.

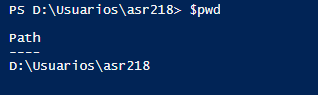
Get-ChildItem variable: # te muestra las variables definidas

Get-ChildItem env: # te muestra variables predefinidas

PS> $env:SystemRoot



Echo “tu directorio actual es “ $pwd

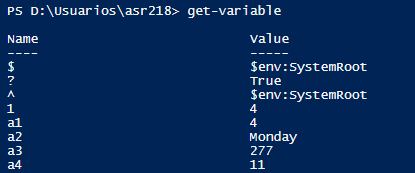


Echo “el directorio principal es” $env:SystemRoot

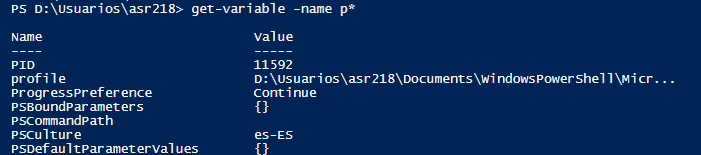


1. Más ejemplos:

>get-variable



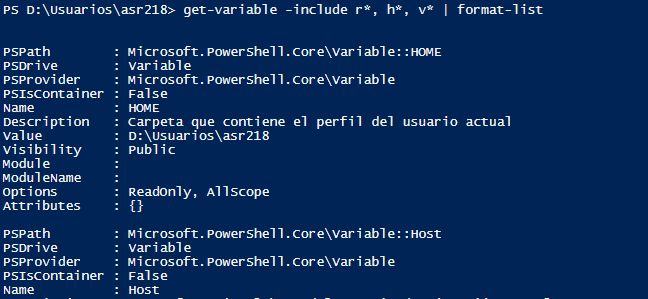
>get-variable -name p\*



>get-variable -name pshome –valueonly

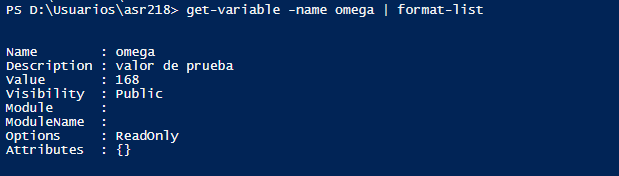


>get-variable -include r\*, h\*, v\* | format-list

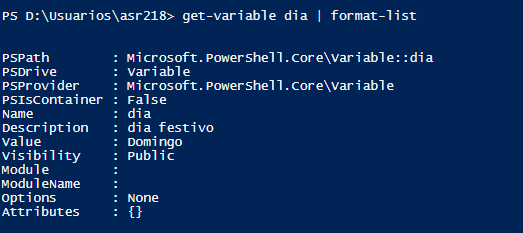


>set-variable -name omega -value 168 -description “valor de prueba” -option readonly

>get-variable -name omega | format-list

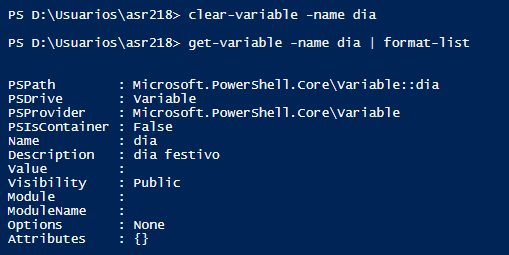


>get-variable dia | format-list



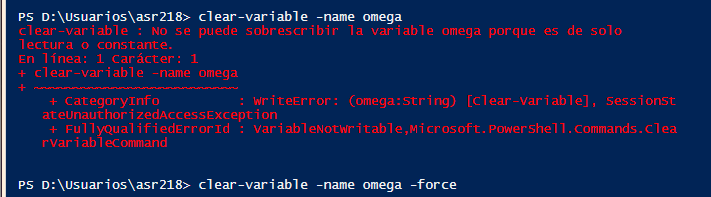
>clear-variable -name dia

>get-variable -name dia | format-list

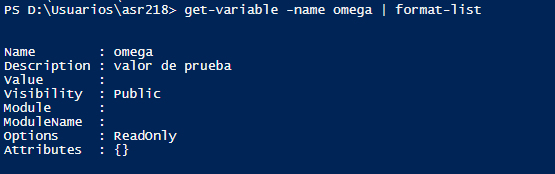


>clear-variable -name omega

>clear-variable -name omega –force

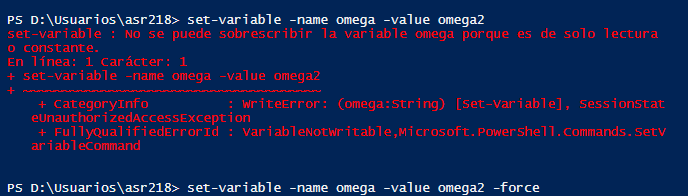


>get-variable -name omega | format-list

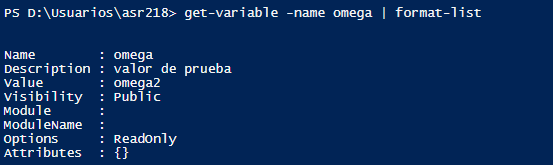


>set-variable -name omega -value omega2

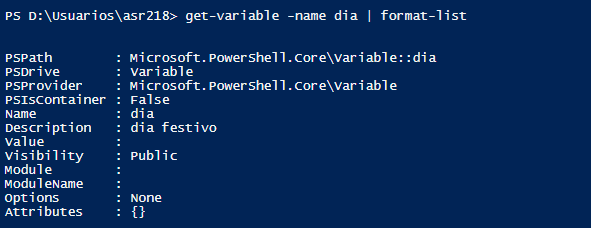
>set-variable -name omega -value omega2 –force



>get-variable -name omega | format-list

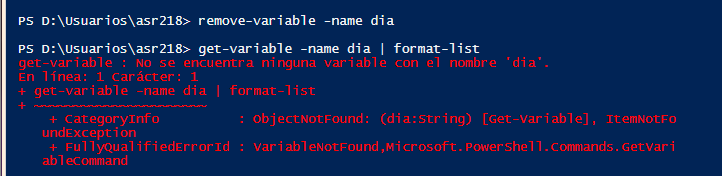


>get-variable -name dia | format-list

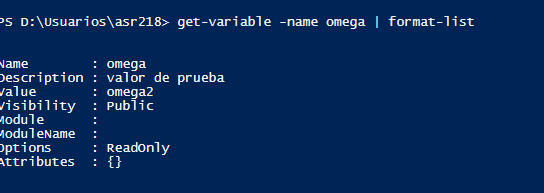


>remove-variable -name dia

>get-variable -name dia | format-list



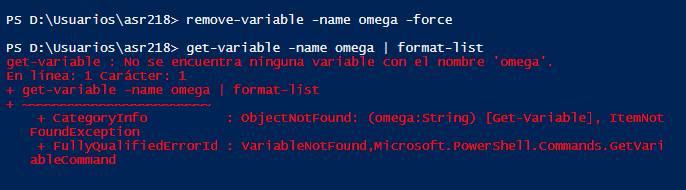
>get-variable -name omega | format-list



>remove-variable -name omega

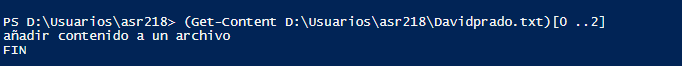
>remove-variable -name omega –force

>get-variable -name omega | format-list



>(Get-Content C:\Users\usuario\Desktops\Textos\texto.txt)[0 ..2] # lista de la fila 1 a la tres del doc texto.txt

Listar los servicios en ejecución



>Get-Service | where {$\_.Status -eq “Running”}

