GENERALITAT VALENCIANA
Conselleria d'Educació, Investigació, Cultura i Esport
Unió Europea
Fons social europeu
L'FSE inverteix en el teu futur
IES MARÍA ENRÍQUEZ
C/ Literat Azorín, 1 46702 – GANDIA
Tfno. 962829430 – Fax. 962829431
Codi: 46004221 – CIF: Q-9655627-I
46004221@edu.gva.es
http://iesmariaenriquez.com

ADMINISTRACIÓ DE SISTEMES OPERATIUS
U5: PowerShell

CFGS ASIX
DPT INF

Windows PowerShell

Vicent Benavent

CFGS ASIX
Mòdul: Administració de Sistemes Operatius
UD5: Windows PowerShell

Tot el temari el podeu trobar a Github: ASO

1 Què és PowerShell?

Dins de l'administració de sistemes Linux hem vist la potència que tenim treballant amb el terminal/consola junt als scripts que hem desenvolupat durant el primer tema. PowerShell és la ferramenta desenvolupada per Microsoft Windows per administrar el sistemes Windows mitjançant la línia d'ordres **(CLI)**, ens ofereix una alternativa a la clàssica administració amb entorn gràfic.

1.1 Com funciona PowerShell

Com sabeu, un dels principals llenguatges de desenvolupament que ha prepara Microsoft és **.NET,** com que és un llenguatge privatiu propietat única de Microsoft s'han basat en este plataforma per desenvolupar el terminal. Per defecte va inclòs en versions de Windows Server 2008 i clients Windows 7.

Una de les principals característiques és que PowerShell no processa el text com altres Shell a Linux, sino que interpreta objectes basats en l'anteriorment esmentada plataforma .NET, el tipo d'ordre que s'utilitza és conegut com a **CMDLETS.** Estos "Command-Lets" estan preparats i desenvolupats de forma que la seua utilització resulta molt senzilla de cara l'usuari i resulta fàcil ampliar les seues possibilitats i compatibilitats. Actualment (Versió 4.0) existeixen més de 1500 cmdlets disponibles a PowerShell i amb cada versió que es desenvolupa s'inclouen de noves.

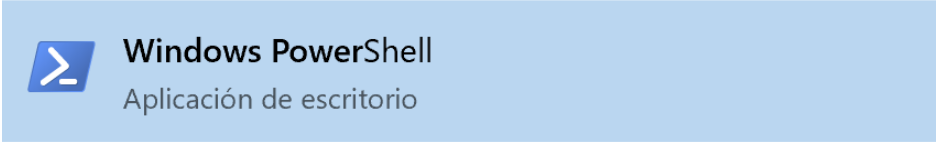
Amb este obtenim un llenguatge del que podem considerar Scripting complet que permet l'administració del sistema, a més, permet crear cmdlets propis que s'adapten a les nostres necessitats. També permet la declaració de dades de tipus simple i complexes i la sintaxis de condicions, iteracions, etc, que conté pràcticament qualsevol llenguatge de desenvolupament.

A més de les ferramentes clàssiques de navegació, llistat de contingut, etc, també permet la navegació directament sobre el registre de Windows, permetent la seua modificació. Cada almacen de dades conté un provider el qual proporciona accés a dades i components que, d'altra forma, no serien fàcilment accessibles des de la línia d'ordres. Les dades es presenten en un format coherent similar a una unitat del sistema de fitxers.

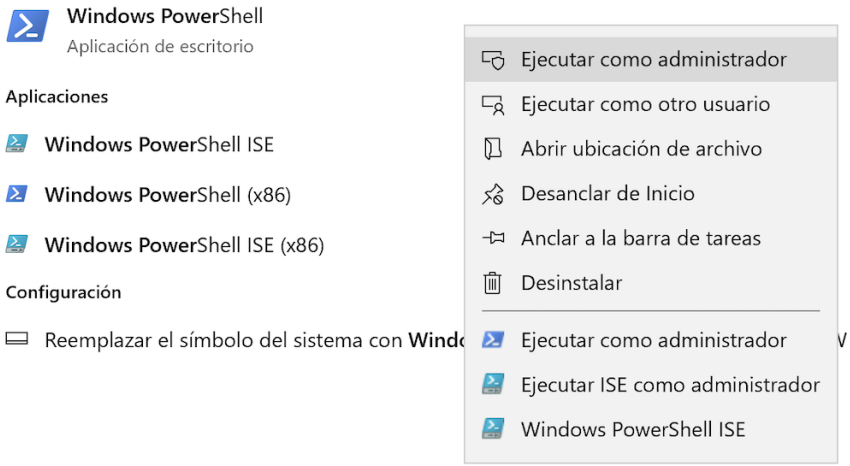
Vos deixe més informació sobre PowerShell, els cmdlet i els providers.

2 PowerShell

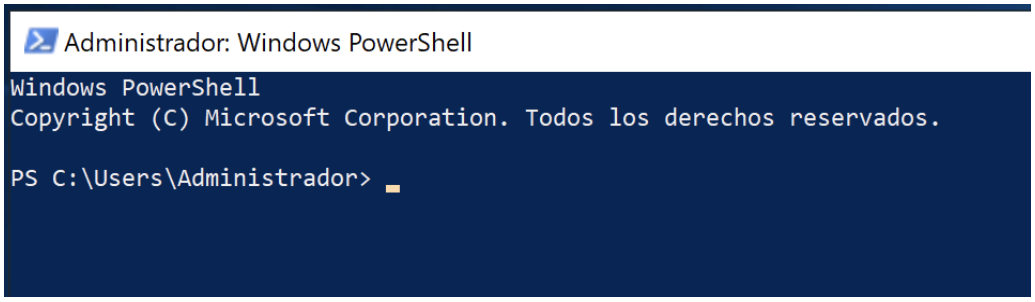
Com qualsevol altra aplicació podem trobar PowerShell al buscador d'aplicacions de Windows, en este cas fem ús de la màquina virtual que hem preparat amb Windows Server 2019. Com podeu observar a la imatge només cal executar l'aplicació, però compte, hem de fer-ho com a usuari Administrador per poder gaudir de totes les funcionalitats.



Com podem fer això? Doncs en este cas, com que som el propi usuari administrador només cal executar-la, però si ens trobem a un client de Windows on som un usuari (encara que tingam permisos d'administrador) més del sistema, cal donar-li a clic dret i "Ejecutar como administrador".



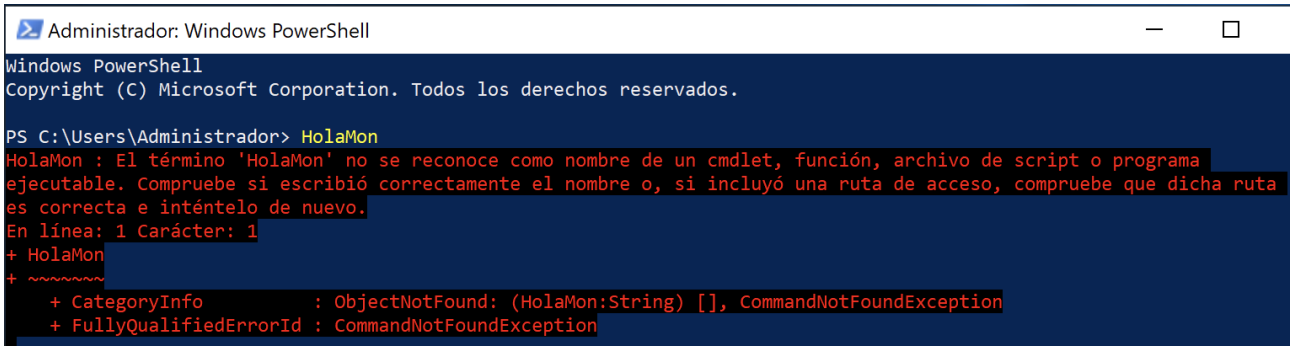
I obtindrem la següent finestra:



Els conceptes d'entrada i eixida estàndards com que es tracta d'un entorn de text són exactament igual que a Linux, el mateix terminal/consola. El primer que podem revisar és que el "prompt" ens mostra la ruta actual igual que al Shell de Linux, només canvia l'estructura del sistema de fitxers que comença amb la lletra del disc i la famosa contrabarra.

Per començar, anem a fer una prova, voreu que si intentem teclejar alguna opció que no està reconeguda com un cmdlet apareixerà un error en roig informant que no es troba l'objecte.

Per exemple, "HolaMon":



Per exemple, si fem ús d'altres cmdlet vàlids, com per exemple:

- **get-command** -> per obtenir informació d'un cmdlet o bé mostrar-los tots els disponibles
  Per exemple, buscant informació sobre get-date:

```
PS C:\Users\Administrador> get-command -name get-date

CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Cmdlet          Get-Date                                           3.1.0.0    Microsoft.PowerShell.Utility
```

- **get-date** -> per obtindre la data

```
PS C:\Users\Administrador> get-date

jueves, 13 de enero de 2022 18:18:19
```

- **get-location** -> equivalent a pwd

```
PS C:\Users\Administrador> Get-location

Path
----
C:\Users\Administrador
```

- **get-host** -> per obtindre informació sobre l'equip on estem executant

```
PS C:\Users\Administrador> get-host


Name             : ConsoleHost
Version          : 5.1.17763.1
InstanceId       : 002d7435-4cc9-49c7-8049-c560739343f6
UI               : System.Management.Automation.Internal.Host.InternalHostUserInterface
CurrentCulture   : es-ES
CurrentUICulture : es-ES
PrivateData      : Microsoft.PowerShell.ConsoleHost+ConsoleColorProxy
DebuggerEnabled  : True
IsRunspacePushed : False
Runspace         : System.Management.Automation.Runspaces.LocalRunspace
```

- **get-childitem**
- **get-history** -> mostra l'històric del PowerShell
- **get-random** -> genera un nombre aleatori
- **clear-host** -> per netejar la pantalla
- I molts més.

Si vos heu fixat, la majoria de cmdlets que hem utilitzat comencen amb "get", lògicament amb "get" estem obtenint informació, però, com podem fer per canviar la informació? Senzill, amb "set", com per exemple per canviar de directori:

```
PS C:\Users> set-location C:\Vicent2122
PS C:\Vicent2122> set-location C:\Users
PS C:\Users>
```

```
PS C:\Users> set-variable -name var -value 123
PS C:\Users> get-variable var

Name                           Value
----                           -----
var                            123
```

Teniu molta més informació ací.

Tenim diferents comoditats per treballar a PowerShell, igual que a Shell, si fem ús de les fletxes del teclat, si li donem a la tecla ↑ i ↓ apareix un a un totes les ordres que hem introduït anteriorment. També tenim l'opció de prémer F7 on ens apareixerà directament l'històric.

També compartim una de les millors característiques que té la consola Linux, es tracta de l'autocompletat d'ordres amb el tabulador. Si comencem a escriure qualsevol cmdlet reconegut per PowerShell este s'autocompleta.

Un altre cas és el de l'entrecomillat, si obrim cometes dobles però no les tanquem, agafa aquest valor com una cadena i ens permet continuant escrivint fins que tanquem les dobles cometes.

Com per exemple, anem a fer l'execució equivalent a **echo**, que és **Write-Host** i fem un exemple sense tancar dobles cometes i l'altre tancant-les.

```
PS C:\Users> Write-Host "hola mon
>> espere que esta ordre funcione
>> bé"
hola mon
espere que esta ordre funcione
bé
PS C:\Users> Write-Host "Bon dia Gandia"
Bon dia Gandia
```

Com podeu vore el comportament és exactament igual que al terminal de Linux, interpreta com una cadena tot el que es conté dins de dobles cometes.

Més similituds amb shell:

- **set-location** és equivalent a l'ordre **"cd"**, que també està disponible a PowerShell, a més funciona exactament igual.
    - L'equivalent de fer un "cd" (que ens portaria al directori principal) podem fer ús de **"sl ~"**

      ```
      PS C:\Users\Administrador> cd ..
      PS C:\Users> sl ~
      PS C:\Users\Administrador>
      ```

      Com podeu observar a la captura, independentment del directori on estem ens porta al directori personal de l'usuari que l'executa.

- **get-childitem** és equivalent a "ls", podem arrodonir esta ordre amb "gci", és més, "ls" també funciona com a shell i com no, "dir" que és el clàssic a CMD.

```
PS C:\Users\Administrador> gci


    Directorio: C:\Users\Administrador


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-r---        19/12/2021     11:56                3D Objects
d-r---        19/12/2021     11:56                Contacts
d-r---        19/12/2021     11:56                Desktop
d-r---        19/12/2021     11:56                Documents
d-r---        19/12/2021     11:56                Downloads
d-r---        19/12/2021     11:56                Favorites
d-r---        19/12/2021     11:56                Links
d-r---        19/12/2021     11:56                Music
d-r---        19/12/2021     11:56                Pictures
d-r---        19/12/2021     11:56                Saved Games
d-r---        19/12/2021     11:56                Searches
d-r---        19/12/2021     11:56                Videos
```

- **mkdir** igual que a Linux, s'utilitza per crear noves carpetes a Windows, segueix la mateixa estructura "mkdir nomCarpeta". Com bé sabeu és recomanable utilitzar el famós **"camelCase"** per interpretar millor els noms dels directoris i carpetes.

```
PS C:\Users\Administrador> mkdir carpetaDeProva


    Directorio: C:\Users\Administrador


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        14/01/2022      9:53                carpetaDeProva
```

Recomane encaridament que proveu els anteriors cmdlets i que reviseu el seu funcionament, a més, vos deixe un enllaç per si teniu curiositat de investigar un poc més sobre PowerShell.

https://programminghistorian.org/es/lecciones/introduccion-a-powershell#introducción

2 BIBLIOGRAFIA