Système d’exploitation - powershell

Get-help (help): donne l’aide sur une commande donnée :

Get-help get-help => donne l’aide au sujet de la commande get-help

Plusieurs paramètres sont possible :

-online : get-help ‘nom de la commande’ -online : donne l’aide internet

-detailed : get-help ‘nom commande’ -detailed : donne une aide plus détaillée

-examples : get-help ‘nom commande’ -examples : donne des exemples

Get-verb : donne la liste de tous les ‘’verbes’’ de commande powershell

Get-command (gcm): donne la liste de toutes les ‘’commandes’’ de powershell

gcm \*clear\* => donne la liste de commande possédant le mot ‘clear’

Set-location (sl) : défini le chemin de travail

Get-item (gi) : donne les info de l’éléments au chemin actuel

Get-ChildItem (gci) : donne les info des éléments présents dans le dossier du chemin actuel

-path (pas obligatoire à écrire si placé en premier) : spécifie le chemin du dossier

gci -path ‘’c:\...’’ : donne la liste des éléments présents dans ce dossier

gci -path ‘’c:\’’ -recurse -depth 2: donne la liste de tous les éléments présents dans le disque c a un éloignement de 2 de la racine (c :)

gci -path ‘’c:\’’ -hidden -directory -system -erroraction silentlycontinue => donne la liste de tous les dossiers (-directory) cachés (-hidden) système (-system) avec en cas d’erreur ne l’affiche pas et continue (-erroraction silentlycontinue)

gci ‘’c:\’’ -recurse -filter \*.log : va lister tous les fichiers log du disque c

Start-process powershell | runas => démarre powershell en mode administrateur

New-item (ni) : permet de créer un nouvel ‘item’ dans le chemin actuel

-itemtype : déféni le type d’élément créé (par défaut : un fichier (file))

(file – directory – symboliclink – junction – hardlink)

-path : chemin à partir duquel on créera l’élément

-value : lui donner une ‘valeur’ à la création (un texte par exemple)

-force : force la création de l’élément (attention, si l’élément est déjà existant, il sera écrasé par le nouveau (perte de données, d’info…)

-confirm : demande la confirmation au lancement de la commande

ni -itemtype directory -path ‘’c:\...\toto’’ => crée le dossier toto à ce chemin-là

ni -itemtype hardlink -path ‘’c:\users\azerty\desktop\toto.txt’’ -name “blabla.txt” -target c:\users\documents\blabla.txt. => crée un hardlink toto.txt

remove-item (ri) : permet de supprimer l’élément voulu (attention ce n’est pas une mise à la poubelle, mais véritablement une suppression totale !!)

set-itemproperty (sp) : défini la ‘propriété’ de l’élément ciblé

sp ‘’c:\...’’ -name attributes -value Hidden => permet ici de ‘cacher’ l’élément ciblé (c:\...) => il existe d’autre valeur : archive/normal…

clear-itemproperty : réinitialise les ‘propriétés’ de l’élément ciblé

clear-recyclebin : vider la poubelle

test-path : test le chemin défini => et retour True / False

test-path c:\\...

invoke-item (ii) : ouvre l’élément du chemin ciblé

ii ‘’c:\...’’ => va ouvrir l’élément ciblé

get-content (gc): permet de lire un fichier (d’obtenir le contenu d’un fichier dans la powershell)

gc ‘’c:\...\toto.txt’’ => va afficher dans la powershell le texte contenu dans toto.txt

gc ‘’c:\...\bidule.log’’ => permet donc d’afficher le contenu du fichier log dans la powershell

-totalcount 10 => va afficher les 10 première ligne du fichier

-tail 15 => affiche les 15 dernière ligne du fichier

-encoding utf8 => encode le contenu du fichier en UTF8 lors de sa lecture

Add-content : ajoute du contenu au fichier

Add-content ‘’c:\...\toto.txt’’ -value ‘haha hoho’ => va ajouter haha hoho au fichier texte

Set-content : ajoute du contenu au fichier en écrasant le précédant !

Select-string (sls) : recherche un mot (une chaine de caractère) dans un fichier et affiche la ligne où se trouve cette chaine de caractère

Sls ‘’c:\...\toto.txt’’ -pattern ‘’haha’’ => va chercher et afficher la ligne où se trouve « haha »

Get-history (h) : affiche votre historique des commandes effectuées (avec un numéro d’ID)

Invoke-history (ihy) : relance la commande du numéro ID voulu

Invoke-history -id 33 => relance la commande de l’historique n°33

Clear-host (clear) : efface l’écran de la powershell

Get-host : affiche les info (version…) de la powershell

Out-host : permet d’afficher à l’écran des info mais une page à la fois

Gci ‘’c:\’’ -recurse | out-host -paging => donne la liste de tous les éléments présents dans le disque c mais une page à la fois

Write-host : écrit dans le powershell

Write-host ‘’hihi hoho’’ : affichera hihi hoho dans la powershell

Out-null : effectue la commande mais n’affichera rien sur la powershell

Read-host : permet d’entrer des valeurs dans la powershell (comme un input en python)

Read-host -prompt ‘’c’est quoi ton nom ?’’ => affichera la question ‘’c’est quoi ton nom’’ et attendra que l’utilisateur rentre une valeur

Read-host -prompt ‘’c’est quoi ton nom ?’’ -asSecureString => va encrypter ce que l’utilisateur rentrera

Enable-computerrestore c:\ => active la restauration du disque c

Disable-computerrestore c:\ => désactive la restauration du disque c

Checkpoint-computer : crée un point de restauration de l’ordinateur

Checkpoint-computer -description ‘’blabla’’ -restorePointType application\_install => crée un pt de restauration

Get-computerrestorepoint : donne les point de restauration existant (listé avec un num de séquence)

Restore-computer -restorepoint 1 => restaure l’ordi avec le point de restauration ayant comme num de séquence 1