POWERSHELL

Classe = Objet

Attribut (de l’objet) = Propriétés

Méthode = Possibilitées de l’objet

**Aides:**

https://theredwindows.net/index.php/2020/11/01/powershell-fonctions-simples-et-avancees-aide-integree/

**Verbes:**

Get = Récupère l’info d’une ressource

New = Créer une nouvelle ressource

Remove = Supprimer une ressource

Set = Modifie une ressource

Copy = Copie une ress.

Move = Déplacer

Compare = Affiche différences

**Noms:**

- Module = Ressource est un module

- ADUser = Ress. est un User Active Directory

- Mailbox = Ress. est une messagerie

**Option:**

-Force = Créer un fichier ou dossier même s’il existe déjà

**Méthodes:**

- .Timp() = Retire les espaces avant et après les chaînes de caractères

- .ToUpper() = Met en majuscule les caractères

- .GetType() = Donne le type de variable

- .Line = Donne les lignes

- .GetEnumerator = Affiche le tableaux de hashage par colonne

**Regex:**

. -match \d correspond à tous les chiffres

. -match ^ correspond à un début de chaîne

. -replace "(boy|girl)", "everybody" = Remplace par everybody les mots boy ou girl.

**Commandes:**

- Get-command = Liste les commandes

- Get-location = Donne l’emplacement courant

- Get-Content = cat

- Get-Childitem = ls

- Get-alias = Liste les alias

- Get-Service = Liste les services (en Administrateur)

- Measure-Object = Compte le nombre

- New-File = Créer un nouveaux fichier

- Remove-ADUser = Supprime un User AD

- Remove-Variable = Supprime une variable (Remove-Variable Var = efface $Var)

- Set-location = cd

- Where-Object = filtre

- Where-object = filtre les attribut (en haut des tableaux) Grep

- Select-object = selectionne (awk)

- Compare-object = Affiche les différences

- Get-executionPolicy = Affiche stratégie d’exécution de script

- Get-Acl = Affichage des droits d’accès

- more = Affichage paginé

- Compare-Object = diff

- Write-host = echo

- Select-Object = trie et peut prendre plusieur attributs

- Read-Host = Prend une info (read -p)

- Restart-Computer = Redémarrage

- Invoke-Expression = Exécuter une commande en utilisant la variable comme argument dans la cmdlet: $Var = “whoami”; Invoke-Expression -Command $Var; = Identifiant de l'utilisateur en cour

**Gestion de processus:**

- Get-Process : récupération d'info sur les processus

- Stop-Process : arrêter un processus

- Wait-Process : attendre la fin d'un processus

- Start-Process : lancement d'un processus local

- Invoke-Command : lancement d'une commande local ou distante

- Get-ComputerInfo : récupération d'informations matérielles et système

- Get-CimInstance : récupération d'objets WMI (Windows Management Instrumentation)

- Get-ChildItem env: = Affiche les variables d’environement

**Opérateurs logiques:**

-path = A la racine

-ItemType = Type d’objet (File; Directory…)

-Destination = Dest.

-Property = Propriété (Name;Version; Filepath…)

**Opérateurs de comparaison:**

**-eq** (equal to) et **-ne** (not equal to)

**-gt** (greater than) et **-lt** (less than)

**-ge** (greater than or equal to) et **-le** (less than or equal to)

**-like** et **-notLike** (avec les wildcards)

**-match** et **-notMatch** (avec les expressions régulières)

**-not** ou **!** inverse le code de sortie, (NON logique)

**Opérateur logique booléens:**

**- ! c1** : NON logique (vrai si c1 est faux et vice versa)

**- c1 -and c2** : ET logique (vrai si c1 et c2 vrai)

**- c1 -or c2** : OU logique (vrai si l’une des 2 conditions ou les 2 sont vraies)

**- c1 -xor c2** : OU exclusif logique ( vrai si uniquement l’une des 2 conditions est vraies)

**Opérateurs:**

**- Test-Path p** : vrai si p existe

**Syntaxes:**

- Créer un fichier = New-Item Nomdufichier -itemtype File

- Créer un dossier = New-Item Nomdudossier -ItemType Directory

- Déplacer un fichier = Move-Item nomdufichier -Destination Nomdufichier

- Effacer un fichier = Remove-Item nomdufichier

- Sélectionner un objet en spécifiant une propriété = select-Object -property Name

- Créer une variable= $objVersion = Get-Host | Select-Object -property - version).version Variable que devient toute cette commande.

- Démarrer un service (par son nom) = Start-Service -Name nom/du/service

- Arrêter un service (par son nom) = Stop-Service -Name nom/du/service

- Démarrer un processus = Start-Service -Name nom/du/processus

- Arrêt un processus = Stop-Process -Name nom/du/processus

- Modifier stratégie d’exécution de script = Set-ExecutionPolicy -Scope LocalMachine ExecutionPolicy Unrestricted← Paramètres de stratégie

- “ et ‘ sont différent, ‘ affiche simplement; “ interprète les valeurs des variable

- ` = \ en bash

- `t = Tabulation

- [int]$var=”10” = Force la variable à être interprétée en chiffres et non en caractères.

- [string]$var=10 = Rend la variable en chaîne de caractère.

- .\Nom\_du\_Script = Lancer le script

- $(Get-Host).Version = Affiche la ligne version de la commande Get-host

- $(Get-Host).Version.Build = Affiche la colonne build de la ligne version de la commande Get-host

—-----------------------------

if (condition)

{

instructions

}

Else

{

instructions

}

—-------------------------------

Switch (condition)

{

valeur1 {ScriptBlock1}

valeur2 {ScriptBlock2}

…

default {ScriptBlock par défaut}

}

—-----------------------------------

For (initialisation; condition; mise-à-jour)

{

bloc d’instructions

}

—------------------------------------

Foreach (element in collection)

{

bloc d’instructions

}

Peut être utilisé derrière un pipe. % = ForEach

—------------------------------------

While (condition)

{

bloc d’instructions

}

—------------------------------------

Do

{

bloc d’instructions

}

While (condition)

Comme la boucle while mais fait un passage au minimum

—-------------------------------------

Do

{

bloc d’instructions

}

Until (condition)

Exécute jusqu'à ce que la condition soit rempli

**Variables:**

$Error action Preference = Donne l’action en réponse à une erreur produite avec les actions:

* *Silently Continue* = N’affiche pas d’erreur et continue d’exécuter les commande suivantes.
* *Continue =* Affiche une erreur et tente de continuer les commandes suivantes.
* *Inquire =*  Demande à l’utilisateur s’il doit continuer ou arrêter l’action.
* *Stop =* Termine l’action avec une erreur.

- $ARGS = $@

- $null= Variable complètement vide

- $env:username = Variable d'environnement, permet par exemple d’afficher le nom du user

- $? : Représente l'état d'exécution de la dernière opération (True ou False)

- $\_ : Contient l'objet actuel dans l'objet pipeline. Vous pouvez utiliser cette variable dans des commandes qui exécutent une action sur chaque objet ou sur des objets sélectionnés dans un pipeline.

- $ARGS : Représente un tableau des paramètres non déclarés et/ou des valeurs de paramètre qui sont passés à une fonction, un script ou un bloc de script.

- $Null : Variable automatique qui contient une valeur NULL ou vide. Vous pouvez utiliser cette variable pour représenter une valeur absente ou indéfinie dans les commandes et les scripts.

- $True / $False: Représente True ou False. Peut être utilisé dans les commandes et les scripts.

**Arborescence:**

C:\$Recycle.Bin : corbeille du système de fichier c:

C:\PerfLogs : journaux de performance

C:\Program Files : installation des programmes tiers

C:\Program Files (x86) : installation des programmes tiers (32 bits)

C:\ProgramData : données globales des programmes

C:\Recovery : récupération après incident

C:\System Volume Information : métadonnées NTFS et points de restauration

C:\Users : répertoire personnel des utilisateurs

C:\Windows : installation du système