**Planifiez vos tâches avec des scripts PowerShell sur Windows Server**

Vous devez gérer plusieurs postes Windows dans votre entreprise, et des **nouveaux utilisateurs arrivent régulièrement** ? Vous aimeriez **automatiser les tâches d'administration** (création d'utilisateurs, gestion des groupes) et de **sauvegarde** de vos postes ? **PowerShell** est l'outil qu'il vous faut !

Dans ce cours, vous apprendrez à utiliser PowerShell pour administrer votre **Active Directory** et ses utilisateurs. Vous apprendrez les commandes de base, mais aussi à **créer des scripts** qui s'exécutent automatiquement pour pouvoir **planifier vos tâches** !

# Utilisez les commandes de bases de PowerShell

## Découvrez le PowerShell

**Le DOS, ancêtre du PowerShell**

Alors, pour ceux comme moi qui ont connu les débuts de l’informatique, PowerShell est le successeur des interfaces en ligne de commande DOS/Windows

Et il faut savoir que l’on peut toujours utiliser **MS-DOS**, même sur un PC avec **Windows 10** ou avec **Windows 2019 Serveur**. Il suffit de cliquer sur **Invite de commande** dans le menu *Démarrer* de la barre des tâches de Windows ou taper **cmd** dans la barre de recherche

PowerShell, ou *Windows PowerShell*, anciennement ***Microsoft Command Shell* (MSH)**, est une suite logicielle développée par **Microsoft** qui intègre une **interface en ligne de commande**, un langage de **script** nommé PowerShell ainsi qu’un kit de développement. Il est inclus depuis Windows 7 et s’appuie sur le framework Microsoft.NET.

**Utilisez PowerShell**

Pour utiliser PowerShell, vous pouvez procéder de plusieurs manières. Par exemple, il est tout à fait possible de lancer PowerShell depuis l’invite de commande **MS-DOS**, comme je vous ai montré précédemment, en tapant powershell

Le prompt passera par exemple de C:\Users\Administrateur> à PS C:\Users\Administrateur>

Il faut faire attention, car les commandes PowerShell se lancent uniquement dans le mode PowerShell, et certaines commandes **MS-DOS** ne se lancent pas dans le mode PowerShell. Il suffit de taper **cmd** dans le mode PowerShell pour revenir dans le mode **MS-DOS**, et vous verrez que votre prompt changera.

Autre possibilité pour lancer PowerShell, et c’est celle que nous utiliserons : cliquer sur ***Windows PowerShell*** dans le menu *Démarrer* de la **barre des tâches** de **Windows**

**Qu’est-ce que PowerShell ?**

PowerShell est un langage de script fondé sur la **programmation orientée objet**. Le logiciel PowerShell (fichier exécutable  powershell.exe  ) est l’interpréteur de l’interface en ligne de commande de l’environnement de développement **Windows PowerShell**.

Les commandes PowerShell sont constituées d’un **verbe** ou **préfixe** et d’un nom, séparés par un tiret. Elles peuvent être suivies de paramètres, on les appelle des ***commandlets*** ou ***cmdlets*** (***command applets*** en anglais ou ***phrases*** en français).

Le préfixe de **cmdlet** est appelé ***verbe*** car il détermine l’action à effectuer sur les entités désignées dans la phrase. Voyons-en quelques-uns des plus utiles :

* Add  permet d’**ajouter** des données ou informations sur le nom qui le suit ;
* Get  permet d’**obtenir** des données ou informations sur le nom qui le suit ;
* Read  permet de **lire** des données ou informations sur le nom qui le suit ;
* **Clear**  permet de **réinitialiser** l’affichage de l’interface ;
* **Import**et  Export  permettent d’**importer**/**exporter** des fichiers de commande ou des **Alias** ;
* New  permet de **créer** de nouveaux objets ou variables ;
* Set  permet de **définir** des données ou informations sur le nom qui le suit ;
* Write  permet de d’**écrire** des données ou informations sur le nom qui le suit et peut agir comme le compte rendu d’une commande.

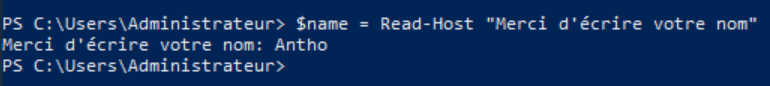
PowerShell est plus proche des langages de **script** comme **Perl** que de langages Shell comme Bash, il n’y a aucune ressemblance entre le langage PowerShell et le **Batch** des fichiers **.BAT** de **DOS/Windows**. Le but de **Microsoft** est de faire un langage de **script** complètement intégré, avec autant de fonctionnalités que celles qui existent sous Unix (ou Linux), et avec le même niveau de sécurité.

À la différence de **Linux**, le langage PowerShell n’est pas sensible à la casse, les commandes pourront s’exécuter, que vous les écriviez en **majuscules** ou en **minuscules** (et même en mélangeant des **minuscules** et des **majuscules**).

Lancez quelques actions de base avec PowerShell

**Récupérez des chaînes de caractères**

On utilise la fonction : **Read-Host**. C’est une fonction qui permet de saisir une chaîne de caractères et de l’enregistrer dans une variable



Pour afficher la variable il suffit de la taper



**Utilisez les bons types de variable**

Voyons également une autre possibilité de PowerShell : c’est la saisie de valeurs numériques. Alors, nous n’allons pas voir tous les types de variable sous PowerShell, mais en voici quelques-unes :

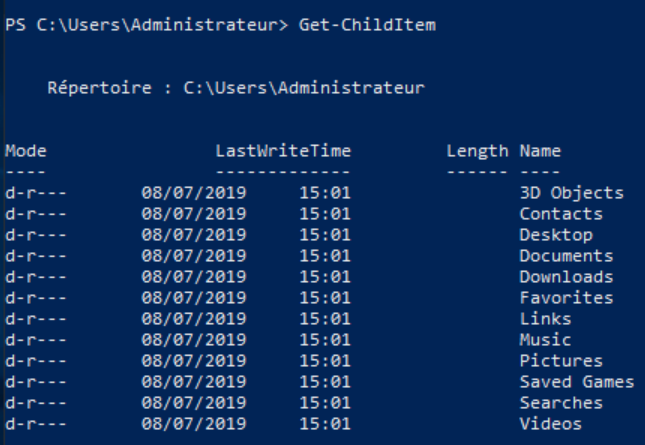
* [string]: chaîne de caractères, on l’a déjà vu précédemment ;
* [char]  : caractère Unicode sur 16 bits ;
* [byte]  : caractère sur 8 bits non signé ;
* [int]  : valeur entière sur 32 bits signée ;
* [long]  : valeur entière sur 64 bits signée ;
* [bool]  : valeur booléenne (True/False : vraie/fausse).

Il existe d’autres types de variables comme  [decimal]  (valeur décimale sur 128 bits),  [DateTime]  (date et heure) ou  [array]  (tableau de valeur)

Gérez vos fichiers avec PowerShell

**Get-Location :** pour savoir où l’on se situe, comme un **pwd** en linux

**Get-ChildItem**: Pour afficher le contenu d’un dossier, comme **dir** ou **ls**



New-Item : Pour créer un répertoire, ex :



On peut également utiliser la commande  New-Item  pour créer un fichier texte **utilisateurs.txt** en tapant :

New-Item -Name "utilisateurs.txt" -ItemType File

Je peux utiliser aussi cette commande pour créer un fichier texte **lectures.txt** et y insérer du texte en tapant :



Pour visualiser le contenu du fichier texte, tapez la commande suivante :

PS C:\Users\Administrateur> notepad.exe lectures.txt

**Manipulez vos fichiers avec PowerShell**

Nous allons copier notre fichier **lectures.txt** dans notre répertoire Vacances en tapant la commande suivante :

**Copy-Item** -Path lectures.txt -Destination Vacances

On pourrait aussi utiliser le chemin absolu ainsi :

**Copy-Item** -Path C:\Users\Administrateurs\lectures.txt -Destination C:\Users\Administrateurs\Vacances

Pour visualiser le contenu d’un répertoire je tape : **Get-ChildItem** NomDuRepertoire

Move-Item : Permet de déplacer un fichier, ex :

**Move-Item** -Path C:\Users\Administrateur\groupes.txt -Destination C:\Users\Administrateur\SAVE

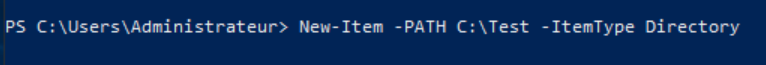
**Rename-Item** : Afin de renommer un fichier

**Rename-Item** -Path utilisateurs.txt -**NewName** utilisateurs\_save.txt

Créer un sous-dossier Algeria dans Vacances



Créer un dossier à la racine :



Pour visualiser ce que contient C



Copier le contenu d’un dossier dans un autre dossier

Gérez vos fichiers et dossiers

Partagez des fichiers

1