R 1.04

2023 - 2024

Introduction aux systèmes d'exploitation et à leur fonctionnement

TP N°4
PowerShell
Windows

ANNE Jean-François
D'après le TP de Antoine CAMBIEN

Le but de ce TD est de se familiariser avec le PowerShell de Windows.

PowerShell Windows

A. ***Présentation***

Windows PowerShell est un interpréteur de commandes conçu pour simplifier l'administration des différents produits Microsoft. PowerShell ISE ajoute par exemple la coloration syntaxique, la complétion d'onglets, le débogage visuel et plus encore. Il s'agit essentiellement d'un langage de programmation basé sur le NET Framework ; vous pouvez l'utiliser pour saisir des commandes PowerShell uniques ou rédiger des scripts PowerShell plus longs. L'environnement Windows PowerShell comporte deux applications : la console PowerShell et l'environnement de script intégré PowerShell ISE. Ce dernier vous offre de meilleures possibilités pour écrire, exécuter et tester vos scripts.

Les commandes PowerShell sont appelées « cmdlets ». Vous pouvez saisir un seul cmdlet dans la ligne de commande ou en combiner plusieurs pour créer des scripts qui exécutent des tâches administratives complexes. Windows PowerShell comprend plus de cent cmdlets, qui vous permettent d'effectuer de nombreuses tâches administratives courantes. Vous pouvez interagir avec divers objets, notamment les utilisateurs, les groupes, la stratégie de groupe et les fichiers.

PowerShell vous permet d'obtenir facilement des données sur les paramètres système courants, d'apporter des modifications aux objets, de gérer les services et de gérer l'accès aux systèmes, y compris aux systèmes de fichiers. Une bonne connaissance des scripts PowerShell vous permet de gagner du temps sur de nombreuses fonctions administratives, sans avoir à acheter ni à implémenter d'outils tiers.

Ce document est une initiation aux commandes de base du langage de commandes PowerShell en mode console. PowerShell est disponible sur les OS récents Microsoft (Windows Server 2008 - 2012 & Windows 7-8-10).

PowerShell inclut un environnement de script intégré PowerShell Integrated Scripting Environment (**ISE**) qui permet d'exécuter des commandes interactives et de modifier et déboguer des scripts dans un environnement graphique.

Les commandes sont saisies dans la console PowerShell, équivalente à l'invite de commandes cmd.exe.

Quelques informations pour commencer :

- Les applets de commande sont simplement nommés commandes PowerShell.
- Le pipeline, utilisé seulement avec la commande Get-Member, n'est pas expliqué.
- Les concepts sur les classes et les objets ne sont pas abordés. Seule l'utilisation des propriétés et des méthodes sont présentées pour la réalisation des exercices.

Ce TP se décompose en cinq parties :

1) Petites astuces de la console
2) Obtenir de l'aide sur une commande
3) Gérer les fichiers et les dossiers
4) Accès aux propriétés et aux méthodes d'un objet

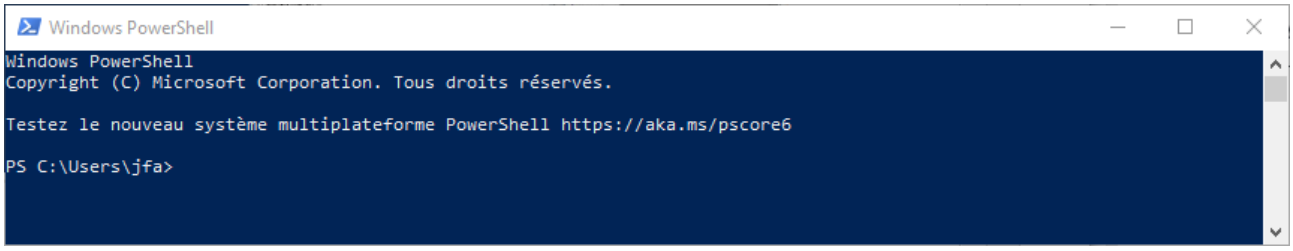
5) Accès aux informations du système

B. Lancer PowerShell

PowerShell comprend une option de ligne de commande et un environnement d'écriture de scripts intégré (ISE) :

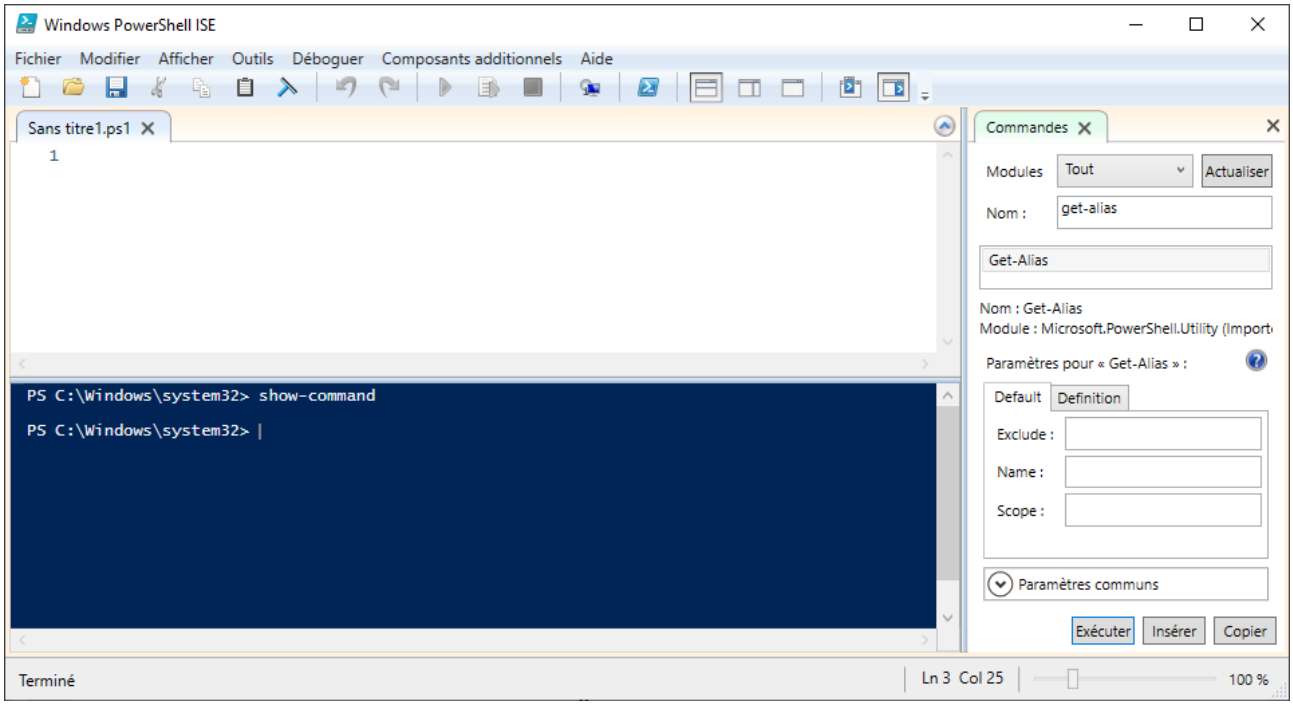
1. En ligne de commande :

Pour ouvrir la ligne de commande PowerShell, tapez **powershell.exe** dans le menu Démarrer de Windows. Un écran comme celui-ci s'affiche :



2. En mode Fenêtré :

Pour lancer l'ISE PowerShell, tapez **powershell_ise.exe** dans le menu Démarrer. L'utilisation de l'ISE Power-Shell est la meilleure façon de travailler avec le langage de script, car cet environnement offre une surbrillance syntaxique, un remplissage automatique des commandes et d'autres fonctions d'automatisation qui simplifient la rédaction et le test des scripts.



C. Petites astuces de la console

Les touches les plus intéressantes :

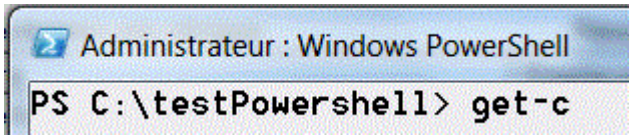
Touche	Description
[Flèche en haut] **[Flèche en bas]**	Permet de faire défiler l'historique des commandes déjà frappées.
[F8]	Fait défiler l'historique sur la ligne de commande.

[Ctrl] C	Met fin à l'exécution de l'instruction courante.

- Une fois la commande retrouvée dans l'historique, vous pouvez soit presser la touche [Entrée] pour la sélectionner et l'exécuter, soit presser la flèche droite (ou gauche) pour modifier la commande avant de l'exécuter.
- La touche tabulation [tab] permet de compléter le nom des commandes, le nom des paramètres et les chemins d'accès aux fichiers et dossiers.
- L'action successive de la touche tabulation [tab] liste les éléments commençant par les caractères spécifiés.

Exemples (ne saisir que la partie encadrée) :

Saisir assez de caractères pour restreindre la liste des commandes listées :



L'action de la touche [tab] propose la première commande puis les commandes suivantes commençant par get-c :

```
PS C:\testPowershell> Get-ChildItem
```

La saisie d'un espace et l'action de la touche [tab] liste les éléments du dossier actif :

```
PS C:\testPowershell> Get-ChildItem .\testdossier
```

Remarque : le point devant le \ représente le chemin du dossier actif (ici c:\testPowershell).

La saisie du début d'un paramètre –r :

```
PS C:\testPowershell> Get-ChildItem .\testdossier -r
```

L'action de la touche [tab] complète le nom du paramètre :

```
PS C:\testPowershell> Get-ChildItem .\testdossier -Recurse
```

Remarque : la saisie seule du caractère (-) permet de lister tous les paramètres possibles.

Il est possible bien sûr de spécifier le chemin en partant d'une lettre de volume disque :

```
PS C:\testPowershell> Get-ChildItem c:\w
```

L'action de la touche [tab] complète le nom du dossier :

```
PS C:\testPowershell> Get-ChildItem C:\Windows
```

Remarque : Le chemin peut ainsi être entièrement complété à l'aide de l'action successive de [tab].

D. ***Obtenir de l'aide sur une commande***

En vous aidant de l'annexe 1 :

- Quelle est la Commandes pour obtenir de l'aide

1. Réaliser les actions suivantes :

- Afficher l'aide sur la commande Get-Alias
- Afficher l'aide avec les exemples sur la commande Get-Alias

En vous aidant de cette aide :

- Afficher tous les alias dont le nom commence par la lettre g
- Afficher la commande qui correspond à l'alias dont le nom est sl
- Afficher tous les alias dont la définition est Get-ChildItem

(Retrouver les alias de la commande DOS et de la commande Linux pour ceux qui connaissent ces systèmes)

- A partir de l'exemple 2 de l'aide de la commande Get-PSDrive, afficher les informations du volume nommé C
- Afficher les méthodes et les propriétés des objets retournés par la commande Get-Location
- Afficher les méthodes et les propriétés des objets retournés par la commande Get-PSDrive

Remarque : L'utilisation des propriétés et des méthodes sera abordée dans la partie 4).

E. Gérer les fichiers et les dossiers

En vous aidant de l'annexe 2 : Commandes pour gérer les fichiers et les dossiers

1. Réaliser les actions suivantes :

- Afficher le chemin du dossier courant
- Se déplacer à la racine de la partition C: (chemin c:\)
- Afficher la liste des dossiers et fichiers
- Se déplacer dans le répertoire data de la partition C: (chemin c:\data)
- A cet emplacement, créer un dossier nommé testPowerShell
- Se déplacer dans le dossier c :\data\testPowerShell
- Créer un dossier nommé testdossier
- Créer un fichier nommé test1.txt, contenant la phrase "Tp PowerShell 1"
- Afficher la liste des dossiers et fichiers
- Copier le fichier test1.txt sous le nom test2.txt
- Renommer le fichier test1.txt avec le nom essai1.txt

- Copier le fichier essai1.txt dans le dossier data\testdossier\essai1.txt
- Afficher la liste des fichiers du dossier et des sous-dossiers de testPowerShell
- Copier le dossier testdossier (avec ses fichiers) dans un nouveau dossier test2dossier
- Déplacer le fichier test2.txt dans le dossier testdossier
- Supprimer le dossier test2dossier (avec ses fichiers)
- Tester l'existence du dossier c:\windows
- Afficher le contenu du dossier c:\windows
- Afficher la liste des fichiers .exe du dossier c:\windows

F. Accès aux propriétés et aux méthodes d'un objet

En vous aidant de l'annexe 3 : initiation aux variables, aux propriétés et aux méthodes des objets

- Affecter à la variable $loc, le résultat de la commande Get-Location.
- Afficher les propriétés et les méthodes de la variable $loc
- Afficher le chemin du dossier courant contenu dans cette variable.
- Afficher les informations sur le disque contenu par cette variable.
- Afficher les informations sur le 'Provider' contenu par cette variable.
- Affecter à la variable $lect, le résultat de la commande Get-PSDrive –Name C
- Afficher les propriétés et les méthodes de la variable $lect
- A partir de la variable $lect, afficher la description du lecteur C, afficher la taille en octet du volume utilisé, afficher la taille en octet du volume libre.

Remarque : pour avoir ces tailles en Go, diviser par 1GB (en utilisant l'opérateur /)

- Affecter à la variable $fichier, le résultat de la commande Get-ChildItem c:\testPowerShell\essai1.txt
- Afficher les propriétés et les méthodes de la variable $fichier
- A partir de la variable $fichier, afficher le nom du fichier, afficher la taille en octet du fichier, afficher le nom complet du fichier (avec le chemin), afficher l'extension seule du fichier, afficher la date du dernier accès.
- A l'aide d'une méthode de la variable $fichier, copier ce fichier dans un nouveau fichier nommé :

```
C:\TestPowerShell\essai2.txt
```

- A partir de la variable $fichier, supprimer le fichier essai1.txt

- Vérifier avec la commande Get-ChildItem
- Lancer notepad.exe et réduire la fenêtre du Bloc-notes
- Lancer la commande Get-Process et vérifier que le Bloc-notes soit bien dans les processus actifs
- Affecter à la variable $proc, le résultat de la commande Get-Process notepad
- Afficher les propriétés et les méthodes de la variable $proc
- A partir de la variable $proc, afficher la description du processus, afficher le chemin d'accès de l'exécutable.
- A partir de la variable $proc, supprimer (tuer) le processus du Bloc-notes

G. ***Accès aux informations du système***

En vous aidant de l'annexe 4 : Accéder aux ressources du système d'exploitation Windows

- Afficher toutes les informations concernant le contrôleur vidéo de votre système
- Affecter à la variable $video, le résultat de la commande précédente
- Afficher les propriétés et les méthodes de la variable $video
- A partir de la variable $video, afficher le nom du contrôleur, la version du driver, le mode vidéo (résolution) et le nom du processeur vidéo
- Afficher les informations concernant le système d'exploitation
- Affecter à la variable $os, le résultat de la commande précédente
- A partir de la variable $os, afficher le nom du système, le type d'architecture (32-64 bits), la date d'installation.
- Afficher les informations concernant les disques logiques de votre système
- Affecter à la variable $vol, le résultat de la commande précédente

Attention, si votre système comporte plusieurs disques logiques, la variable $vol est un tableau d'objets (voir annexe 4)

- A partir de la variable $vol, et pour le premier disque logique seulement, afficher le nom du volume, afficher la taille, afficher l'espace libre, afficher le système de fichiers.

Remarque : pour avoir ces tailles en Go, diviser par 1GB (en utilisant l'opérateur /)

H. Annexe 1 : Commandes pour obtenir de l'aide

- Afficher de l'aide sur une commande :
  `Get-Help Commande (ex : Get-Help Get-ChildItem)`

- Afficher les exemples :
  `Get-Help Commande -Examples`

- Afficher les alias :
  `Get-Alias`

- Afficher la liste des méthodes et des propriétés des objets :
  `Commande | Get-member`

I. Annexe 2 : Commandes pour gérer les fichiers et les dossiers

- Se déplacer dans les dossiers :
  `Set-Location chemin (ex : Set-Location c:\temps)`

- Afficher le chemin du dossier courant :
  `Get-Location`

- Afficher le contenu d'un dossier :
  `Get-ChildItem`

- Créer un dossier :
  `New-Item nomDossier -ItemType directory`

- Créer un fichier avec du texte :
  `New-Item nomFichier.txt -ItemType file -Value "texte"`

- Supprimer un fichier ou un dossier :
  `Remove-Item nomFichier.txt`

- Déplacer un fichier :
  `Move-Item nomFichier.txt -Destination chemin\nomFichier.txt`

- Déplacer un dossier :
  `Move-Item nomDossier -Destination chemin\nomDossier`

- Renommer un fichier ou dossier :

```
Rename-Item nomFichier.txt -NewName nomFichier2.txt
```

- Copier un fichier :

```
Copy-Item nomFichier.txt -Destination nomFichier2.txt
```

- Copier un dossier avec ses fichiers :

```
Copy-Item nomDossier -Destination nomDossier1 -Recurse
```

- Tester l'existence d'un fichier ou dossier :

```
Test-Path chemin/nomFichier.txt
```

J. Annexe 3 initiation aux variables, aux propriétés et aux méthodes des objets

Le nom d'une variable commence toujours par $, il peut inclure tout caractère alphanumérique ou le trait de soulignement.

Windows PowerShell permet de créer des variables qui sont pour l'essentiel des objets nommés. La sortie de toute commande Windows PowerShell valide peut être stockée dans une variable.

Exemple :

```
$loc = Get-Location
```

Il est possible d'utiliser Get-Member pour afficher des informations sur le contenu de variables.

Exemple :

```
$loc | Get-Member ( idem Get-Location | Get-Member )
```

Le nom de la variable suivi du point permet d'accéder aux propriétés de l'objet référencé par la variable, exemple pour la propriété Path de la variable $loc.

Exemple :

```
$loc.Path
```

Remarque :

L'usage de la touche tabulation [tab] permet de compléter le nom de la propriété.

De même, l'exécution d'une méthode (action) d'un objet :

Exemple :

```
$fichier.Delete()
```

Remarque :

Pour les méthodes, ne pas oublier les parenthèses avec ou sans paramètre.

K. Annexe 4 : Accéder aux ressources du système d'exploitation Windows

Les classes WMI (Windows Management Instrumentation) décrivent les ressources qui peuvent être gérées. Il existe des centaines de classes WMI, certaines d'entre elles contenant des dizaines de propriétés.

La commande principale est Get-WmiObject, elle permet de lire ces ressources.

Exemple pour consulter les informations suivantes :

- Graphiques :

```
Get-WmiObject win32_videocontroller
```

- Système :

```
Get-WmiObject win32_operatingsystem
```

- Disques :

```
Get-WmiObject win32_logicaldisk
```

Il est toujours possible d'affecter le résultat de la commande Get-WmiObject à une variable, et de consulter les propriétés et les méthodes de l'objet à l'aide de la commande Get-Member.

Si le résultat de la commande est un ensemble d'objets, la variable affectée est un tableau d'objet, l'accès au premier élément se fait alors de la manière suivante $var[0], au second élément : $var[1], etc..

II. Webographie :

Site Web de Windows PowerShell :

- http://www.microsoft.com/powershell

Site Openclassroom :

- https://openclassrooms.com/fr/courses/3664366-creez-votre-premier-script-avec-powershell
- https://openclassrooms.com/fr/courses/2356306-prenez-en-main-windows-server/5836391-administrez-votre-serveur-a-l-aide-de-powershell

Site netwrix :

- https://blog.netwrix.fr/2018/09/26/tutoriel-de-windows-powershell-scripting-pour-debutants/
- https://blog.netwrix.fr/tag/powershell/

Autres :

- http://stephane.lavirotte.com/teach/cours/systemes/01%20Configuration%20Comptes.pdf
- https://blog.netwrix.fr/tag/powershell/
- https://blog.netwrix.fr/2018/09/26/tutoriel-de-windows-powershell-scripting-pour-debutants/
- https://www.it-connect.fr/powershell-pour-les-debutants-1ere-partie/
- https://siocours.lycees.nouvelle-aquitaine.pro/lib/exe/fetch.php/sisr1/tp_powershell_01.odt
-

III. Aide :

Si vous n'avez pas accès à toute l'aide, exécutez PowerShell en administrateur et exécutez la commande :

Update-help