**PowerShell**

PowerShell permet d’entrer des commandes en mode texte.

Commandes :  
get -ChildIItem (permet d’obtenir les objets à l’intérieur du répertoire courant)  
get -Help (permet d’obtenir de l’aide vis-à-vis d’une commande)  
New-Item (permet de créer un fichier ou un dossier)  
Tree (permet de voir les chemins de tous les directoires)  
cd \ (permet de retourner au fichier de base du disque)  
set-location (permet de déterminer le nouveau directoire courant exemple C:\Reg1, mettre .. permet de remonter d’un niveau, on peut les accumuler avec ..\..(PS : cela peut être utiliser dans d’autre commande à l’intérieur du path), si on veut remonter au complet on fait \, si on veux changer d’arborescence marquer le nom du disque Ex : X :)  
Remove-Item (permet de supprimer quelque chose simplement en disant sa position, le mot path peut être omis)  
Get-Content (Donne le contenu d’un fichier, comme si l’on faisait un double clique sur un fichier texte, le mot path peut être omis)  
Set-Content et Add-Content (Ces deux commandes permettent d’écrire dans le ficher, sauf que set-content va supprimer ce qu’il y avait dans le fichier, alors que add-content ne fait qu’ken rajouter)  
Clear-Host (ou cls nettoie ce qu’il y a sur l’écran)  
Get-date (donne la date et l’heure courante)  
Get-Alias (donne des résumés\alias\nom plus court pour les commandes)  
  
Paramètres :  
-Recurse  
-Path (pas de \ pour le premier dossier quand chemin relatif)  
-type (directory ou file)  
-name  
-include   
-\* (est un caractère joker Ex : -include A\* va donner tout les fichiers ou dossiers qui commence par A, \* remplace le reste des caractères)  
-value (ce qui sera ajouter au fichier au quel l’on souhaite ajouter quelque chose)  
-| (pinging permet de relier deux commandes)  
  
Raccourcis :  
ctrl + c Arrête toute commande en cours d’exécution

Structure des répertoires utilisés dans les notes :





Façon de nommer les fichiers : (comme Windows)

1. Chemin absolu : chemin à partir de la racine

Ex : pour nommer le fichier Fich2 en utilisant un chemin absolu :

C:\RepB12\Rep1\Rep2\Fich2

1. Chemin relatif : à partir du répertoire courant (actif)

Répertoire courant : répertoire dans lequel on est (équivalent du répertoire sélectionné en Windows)

Ex : pour nommer le fichier Fich2 en utilisant le chemin relatif avec Rep1 comme répertoire courant :

Rep2\Fich2

1. **Get-ChildItem**

Donne la liste des fichiers et répertoires dans un répertoire donné.

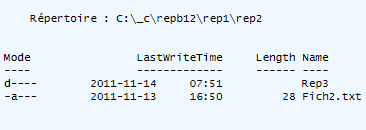
Exemples :

1. Faire afficher la liste des fichiers et répertoires sous Rep2 en utilisant un chemin absolu



**Get-ChildItem C:\RepB12\Rep1\Rep2**

Résultat :



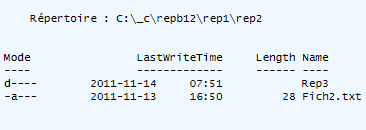
Note : résultat : 1 niveau seulement

1. Faire afficher la liste des fichiers et répertoires sous Rep2 en utilisant un chemin relatif (c’est-à-dire en utilisant le répertoire courant). Le répertoire courant est Rep1.



**Get-ChildItem Rep2**

Résultat :



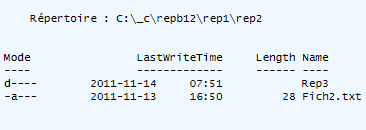
1. Soit maintenant RepB12 le répertoire courant.

Faire afficher la liste des fichiers et répertoires sous Rep2 en utilisant un chemin relatif (c’est-à-dire en utilisant le répertoire courant)



**Get-ChildItem Rep1\Rep2**

Résultat :



**Pour savoir comment utiliser une commande, il faut consulter sa syntaxe :**

**Voici un extrait de la syntaxe de la commande Get-ChildItem :**

**NOM**

Get-ChildItem

**RÉSUMÉ**

Obtient les éléments et les éléments enfants à un ou plusieurs emplacements spécifiés.

**SYNTAXE**

Get-ChildItem [[-Path] <string[]>] [[-Filter] <string>] [-Exclude <string[]>] [-Force] [-Include <string[]>] [-Name] [-Recurse] [-UseTransaction] [<CommonParameters>]

-Path <string[]>

Spécifie un chemin d'accès à un ou plusieurs emplacements. Les caractères génériques sont autorisés. L'emplacement par défaut est le répertoire actif (.).

-Recurse [<SwitchParameter>]

Obtient les éléments aux emplacements spécifiés, de même que dans tous les éléments enfants de ces emplacements.

Recurse n'a d'effet que si le chemin d'accès pointe vers un conteneur doté d'éléments enfants, tel que C:\Windows ou C:\Windows\\*. Il n'a aucun effet si le chemin pointe vers un élément n'ayant pas d'éléments enfants, tel que C:\Windows\\*.exe.

À retenir : c’est le paramètre *path* qui est affecté par la notion de chemin absolu ou relatif

Il est possible d’obtenir de l’information sur les commandes PSH en tapant Get-Help suivi du nom de la commande.

Exemple :

Si on veut de l’information sur la commande Get-ChildItem : Get-Help Get-ChildItem

La commande Get-Help peut aussi être utilisée avec les paramètres  suivants :

-examples -detailed -full (Voir Annexe)

OU BIEN, on peut taper la commande dans Google… : Get-Child-Item

VOIR LES EXEMPLES LORSQUE VOUS CONSULTEZ L’AIDE

1. **Tree (commande DOS)**

Affiche l’arborescence des répertoires

**Tree**   : affiche l’arborescence des répertoires

**Tree /f** : affiche l’arborescence des fichiers et des répertoires

1. **New-Item**

Crée un élément (ex : fichier ou répertoire)

New-Item

[[-Path] <String[]>]

[-Name <String>] (obligatoire si -Path est omis)

-Type <String>

[-Value <Object>]

[-Force]

[-Credential <PSCredential>]

[-WhatIf]

[-Confirm]

[<CommonParameters>]

Exemples :

1. Créer le répertoire RepW sous le répertoire RepZ en utilisant le chemin relatif (répertoire courant : RepX)



**New-Item -name RepW -path RepY\RepZ -itemtype directory**

Autre syntaxe correcte : sans le paramètre -name (on ajoute alors le nom au bout du chemin (path)

**New-Item -path RepY\RepZ\RepW -type directory**

Autre syntaxe correcte : sans le paramètre -path (on doit alors mettre le paramètre -name)

**New-Item -name RepW -type directory**

1. Créer le fichier Toto.txt dans le répertoire RepY en utilisant le chemin relatif (répertoire courant : RepY)



**New-Item -name Toto.txt -type file**

Note : lorsqu’on est dans le répertoire courant, on peut omettre le paramètre *path* (on aurait aussi pu utiliser *path .* )

1. **Remove-Item**

Détruit un ou des élément(s)

Exemples :

1. Détruire le répertoire RepW créé précédemment en utilisant le chemin relatif (répertoire courant : RepY)



**Remove-Item Repz\Repw**

1. Détruire le Fichier Toto créé précédemment en utilisant le chemin relatif (répertoire courant : RepX)



**Remove-Item RepY\Toto**

1. **Get-Content**

Donne le contenu d’un fichier

Exemple :

Faire afficher le contenu du fichier Fich2.txt en utilisant le chemin relatif (répertoire courant : Rep1)



**Get-content Rep2\Fich2.txt**

1. **Set-Content et Add-Content**

Ces deux commandes permettent d’écrire dans un fichier.

La première écrase ce qu’il y avait dans le fichier en écrivant et la seconde ajoute du texte au texte déjà existant.

Note : si le fichier n’existe pas, les commandes Set-Content et Add-Content vont le créer

Set-Content

[-Path] <string[]>

[-Value] <Object[]>

[-PassThru]

[-Filter <string>]

[-Include <string[]>]

[-Exclude <string[]>]

[-Force]

[-Credential <pscredential>]

[-WhatIf]

[-Confirm]

[-NoNewline]

[-Encoding <Encoding>]

[-AsByteStream]

[-Stream <string>]

[<CommonParameters>]

Exemples :

1. Remplacer tout ce qu’il y a dans le fichier Fich2.txt par la ligne suivante :ABCDE.

Utiliser le chemin absolu.



**Set-Content -path C:\RepB12\Rep1\Rep2\Fich2.txt -value "ABCDE"**

Note : on peut omettre le mot -path (ainsi que -value)

1. Ajouter au fichier Fich2.txt la ligne suivante :FGHIJ.

Cette fois, utiliser le chemin relatif (Répertoire courant : RepB12).



**Add-Content Rep1\Rep2\Fich2.txt -value "FGHIJ"**

1. **Set-Location**

Modifie le répertoire courant

Soit Rep1 le répertoire courant.

Si on veut que Rep3 devienne le répertoire courant, il faut faire la commande suivante :



**Set-Location Rep2\Rep3**

Notes :

1. .. fait remonter le répertoire courant d’un niveau

Exemple :

Soit Rep4 le répertoire courant . La commande **Set-location ..** fera en sorte que le répertoire courant devienne Rep3.

ii) on peut utiliser plusieurs fois .. dans une commande pour faire remonter de plus d’un niveau

Exemple :

Soit Rep4 le répertoire courant . La commande **Set-location ..\..** fera en sorte que le répertoire courant devienne Rep2.

iii) \ remonte à la racine

Exemple : Soit Rep4 le répertoire courant. La commande **Set-location \** fera en sorte que le répertoire courant devienne C :

iv) .. et \ peuvent aussi être utilisés dans les autres commandes

Exemple :

Faire afficher le contenu du fichier Fich2.txt en utilisant le chemin relatif (Rep4 est le répertoire courant)



**Get-Content ..\..\Fich2.txt**

v) Lorsqu’on veut changer le contexte courant pour une autre racine, on peut omettre **set-location**

Exemple :

Soit C:\repB12\rep1 le contexte courant et je veux avoir i:\ comme contexte courant :

Je peux taper **set-location** **i :** ou simplement **i :**

vi) –path . est équivalent à omettre le paramètre path dans la commande

1. **Copy-Item**

Copie un élément d’un endroit à un autre

1. **Clear-Host**

Efface le contenu de l’écran (partie résultats des commandes)

1. **Get-date**

Donne la date et l’heure courantes

**Caractères génériques (Wildcards)**

L’étoile (\*) peut être utilisée pour remplacer des caractères.

Exemple :

Si on veut faire afficher seulement les items du répertoire **C:\RepB12\Rep1\Rep2** qui commencent par A :

**Get-ChildItem C:\RepB12\Rep1\Rep2 -include A\***

**Piping**

Il est possible de joindre deux (ou plus) commandes PSH en utilisant le piping.

Le résultat de la première commande est envoyé à la deuxième commande. Le résultat de la deuxième commande est envoyé vers la troisième commande, ainsi de suite jusqu’à la fin.

Exemple :

Ajouter le contenu du fichier Fich3.txt au contenu du fichier Fich2.txt. Utiliser le chemin relatif (Répertoire courant : RepB12)



Get-content Rep1\Rep2\Rep3\Fich3.txt | Add-content Rep1\Rep2\Fich2.txt

**ANNEXE**

PS C:\\_c\repb12> **get-help get-childItem**

**NOM**

Get-ChildItem

**RÉSUMÉ**

Obtient les éléments et les éléments enfants à un ou plusieurs emplacements spécifiés.

**SYNTAXE**

Get-ChildItem [[-Path] <string[]>] [[-Filter] <string>] [-Exclude <string[]>] [-Force] [-Include <string[ ]>] [-Name] [-Recurse] [-UseTransaction] [<CommonParameters>]

Get-ChildItem [-LiteralPath] <string[]> [[-Filter] <string>] [-Exclude <string[]>] [-Force] [-Include <st ring[]>] [-Name] [-Recurse] [-UseTransaction] [<CommonParameters>]

**DESCRIPTION**

L'applet de commande Get-ChildItem obtient les éléments présents dans un ou plusieurs emplacements spécif iés. Si l'élément est un conteneur, elle obtient les éléments qui se trouvent à l'intérieur du conteneur,appelés éléments enfants. Vous pouvez utiliser le paramètre Recurse pour obtenir les éléments présents dans tous les conteneurs enfants.

L'emplacement peut être un emplacement de système de fichiers (répertoire, par exemple) ou un emplacementexposé par un autre fournisseur (ruche de Registre ou magasin de certificats, par exemple).

**LIENS CONNEXES**

Online version: http://go.microsoft.com/fwlink/?LinkID=113308

about\_Providers

Get-Item

Get-Alias

Get-Location

Get-Process

PS C:\\_c\repb12> **get-help get-childItem -detailed**

**NOM**

Get-ChildItem

**RÉSUMÉ**

Obtient les éléments et les éléments enfants à un ou plusieurs emplacements spécifiés.

**SYNTAXE**

Get-ChildItem [[-Path] <string[]>] [[-Filter] <string>] [-Exclude <string[]>] [-Force] [-Include <string[]>] [-Name] [-Recurse] [-UseTransaction] [<CommonParameters>]

Get-ChildItem [-LiteralPath] <string[]> [[-Filter] <string>] [-Exclude <string[]>] [-Force] [-Include <strin g[]>] [-Name] [-Recurse] [-UseTransaction] [<CommonParameters>]

**DESCRIPTION**

L'applet de commande Get-ChildItem obtient les éléments présents dans un ou plusieurs emplacements spécifiés . Si l'élément est un conteneur, elle obtient les éléments qui se trouvent à l'intérieur du conteneur, appelés éléments enfants. Vous pouvez utiliser le paramètre Recurse pour obtenir les éléments présents dans tous les conteneurs enfants.

L'emplacement peut être un emplacement de système de fichiers (répertoire, par exemple) ou un emplacement ex posé par un autre fournisseur (ruche de Registre ou magasin de certificats, par exemple).

**PARAMÈTRES**

-Exclude <string[]>

Omet les éléments spécifiés. La valeur de ce paramètre qualifie le paramètre Path. Entrez un élément ou un modèle de chemin d'accès, tel que « \*.txt ». Les caractères génériques sont autorisés.

-Filter <string>

Spécifie un filtre dans le format ou le langage du fournisseur. La valeur de ce paramètre qualifie le pa ramètre Path. La syntaxe du filtre, notamment l'utilisation de caractères génériques, dépend du fournisseur. Les filtres sont plus efficaces que les autres paramètres, car le fournisseur les applique lors de la récupération des objets, plutôt que de laisser le soin à Windows PowerShell de filtrer les objets unefois qu'ils ont été récupérés.

-Force [<SwitchParameter>]

Permet à l'applet de commande d'obtenir des éléments, tels que des fichiers cachés ou système qui, sinon , ne sont pas accessibles par l'utilisateur. L'implémentation est différente d'un fournisseur à l'autre.

Pour plus d'informations, consultez about\_Providers. Même en utilisant le paramètre Force, l'applet de commande ne peut pas remplacer les restrictions de sécurité.

-Include <string[]>

Récupère uniquement les éléments spécifiés. La valeur de ce paramètre qualifie le paramètre Path. Entrez un élément ou un modèle de chemin d'accès, tel que « \*.txt ». Les caractères génériques sont autorisés.

Le paramètre Include n'a d'effet que si la commande inclut le paramètre Recurse ou si le chemin d'accès mène au contenu d'un répertoire tel que C:\Windows\\*, où le caractère générique spécifie le contenu du répertoire C:\Windows.

-LiteralPath <string[]>

Spécifie un chemin d'accès à un ou plusieurs emplacements. Contrairement à Path, la valeur de LiteralPath est utilisée exactement telle que vous la tapez. Aucun caractère n'est interprété en tant que caractère générique. Si le chemin d'accès inclut des caractères d'échappement, mettez-le entre des guillemets simples. Les guillemets simples indiquent à Windows PowerShell qu'aucun caractère ne doit être interprété en tant que séquence d'échappement.

-Name [<SwitchParameter>]

Récupère uniquement les noms des éléments à partir des emplacements. Si vous dirigez la sortie de cette commande vers une autre commande, seuls les noms des éléments sont transmis.

-Path <string[]>

Spécifie un chemin d'accès à un ou plusieurs emplacements. Les caractères génériques sont autorisés. L'emplacement par défaut est le répertoire actif (.).

-Recurse [<SwitchParameter>]

Obtient les éléments aux emplacements spécifiés, de même que dans tous les éléments enfants de ces emplacements.

Recurse n'a d'effet que si le chemin d'accès pointe vers un conteneur doté d'éléments enfants, tel que C:\Windows ou C:\Windows\\*. Il n'a aucun effet si le chemin pointe vers un élément n'ayant pas d'éléments enfants, tel que C:\Windows\\*.exe.

-UseTransaction [<SwitchParameter>]

Inclut la commande dans la transaction active. Ce paramètre est valide uniquement lorsqu'une transaction est en cours. Pour plus d'informations, consultez about\_Transactions.

<CommonParameters>

Cette applet de commande prend en charge les paramètres courants : Verbose, Debug,ErrorAction, ErrorVariable, WarningAction, WarningVariable,OutBuffer et OutVariable. Pour plus d'informations, tapez« get-help about\_commonparameters ».

-------------------------- EXEMPLE 1 --------------------------

C:\PS>get-childitem

Description

-----------

Cette commande obtient les éléments enfants à l'emplacement actuel. S'il s'agit d'un répertoire de système d e fichiers, elle obtient les fichiers et les sous-répertoires du répertoire actif. Si l'élément n'a pas d'él éments enfants, cette commande retourne à l'invite de commandes sans rien afficher.

L'affichage par défaut répertorie le mode (attributs), l'heure de la dernière écriture, la taille du fichier (longueur) et le nom du fichier. Les valeurs valides pour le mode sont d (répertoire), a (archive), r (en lecture seule), h (masqué) et s (système).

-------------------------- EXEMPLE 2 --------------------------

C:\PS>get-childitem . -include \*.txt -recurse -force

Description

-----------

Cette commande récupère tous les fichiers .txt présents dans le répertoire actif et ses sous-répertoires. Lepoint (.) représente le répertoire actif et le paramètre Include spécifie l'extension du nom de fichier. Le paramètre Recurse indique à Windows PowerShell que les objets doivent être récupérés de manière récursive e t que l'objet de la commande est le répertoire spécifié et son contenu. Le paramètre force ajoute des fichie rs masqués à l'affichage.

-------------------------- EXEMPLE 3 -------------------------

C:\PS>get-childitem c:\windows\logs\\* -include \*.txt -exclude A\*

Description

-----------

Cette commande répertorie les fichiers .txt du sous-répertoire Logs, à l'exception de ceux dont le nom comme nce par la lettre A. Elle utilise le caractère générique (\*) pour indiquer le contenu du sous-répertoire Logs et non le conteneur du répertoire. Comme le paramètre Recurse n'est pas inclus dans la commande, Get-Child Item n'inclut pas automatiquement le contenu du répertoire. Vous devez par conséquent spécifier ce contenu.

-------------------------- EXEMPLE 4 --------------------------

C:\PS>get-childitem registry::hklm\software

Description

-----------

Cette commande récupère toutes les clés de Registre de la clé HKEY\_LOCAL\_MACHINE\SOFTWARE dans le Registre sur l'ordinateur local.

-------------------------- EXEMPLE 5 --------------------------

C:\PS>get-childitem -name

Description

-----------

Cette commande récupère uniquement les noms d'éléments dans le répertoire actif.

-------------------------- EXEMPLE 6 --------------------------

C:\PS>get-childitem cert:\. -recurse -codesigningcert

Description

-----------

Cette commande obtient dans le magasin de certificats tous les certificats ayant une autorité de signature de code.

La commande utilise l'applet de commande Get-ChildItem. Le chemin d'accès spécifie le lecteur Cert: exposé par le fournisseur de certificats de Windows PowerShell. La barre oblique inverse (\) spécifie un sous-répertoire du magasin de certificats, tandis que le point (.) représente le répertoire actif, c'est-à-dire le répe rtoire racine du magasin de certificats. Le paramètre Recurse spécifie une recherche récursive.

Le paramètre CodeSigningCertificate est un paramètre dynamique qui obtient uniquement les certificats ayant une autorité de signature de code. Pour plus d'informations, tapez « get-help certificate ».

-------------------------- EXEMPLE 7 --------------------------

C:\PS>get-childitem \* -include \*.exe

Description

-----------

Cette commande récupère tous les éléments du répertoire actif dont le nom de fichier est doté de l'extension « .exe ». Le caractère générique (\*) représente le contenu du répertoire actif (pas le conteneur). Lors de l'utilisation du paramètre Include sans le paramètre Recurse, le chemin d'accès doit pointer vers le contenuet non un conteneur.