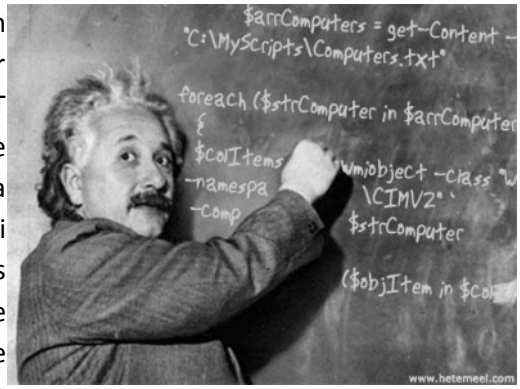
Programmation de scripts objets avec PowerShell

Objectif

L'objectif de ce document est de vous faire découvrir un langage de scripts orienté objets sous environnement Windows avec comme fil conducteur l'administration de machines et d'utilisateurs. Ce cours a été construit avec la version 2 de Powershell; ce qui peut expliquer un certain nombre de différences si vous utilisez la version 3. Par exemple, il n'y a plus d'aide en français. En effet, afin d'avoir une aide constamment à jour, Microsoft a décidé de ne maintenir qu'une aide en anglais.
Une aide en ligne est disponible à cette adresse http://technet.microsoft.com/fr-fr/library/bb978525.aspx mais vous découvrirez des commandes qui permettent d'obtenir de l'aide automatiquement.

Il est demandé que chaque étudiant envoi un mail à la fin de chaque séance à denis.pallez@unice.fr ayant comme sujet [LPSIL_ADMIN] et contenant le fichier PDF (les réponses aux questions auront été saisies dans les cadres « Réponse ») renommé de la façon suivante <Nom étudiant>_TP<N° du TP ou de la séance>_<date du TP en anglais>.zip. Exemple de nom de fichier Pallez_TP1_20130909.pdf. *Tout manquement à cette règle entraînera la note de 0 pour le TP.*

1 Introduction à Windows Powershell



Windows PowerShell™ comprend un interpréteur de commandes et un langage de script basé sur les tâches, conçu spécialement pour l'administration du système. Créé à partir de Microsoft .NET Framework, Windows PowerShell™ aide les professionnels de l'informatique et les utilisateurs chevronnés à contrôler et à automatiser l'administration du système d'exploitation Windows, ainsi que les applications s'exécutant sous Windows. Les commandes Windows PowerShell intégrées, appelées cmdlets (ou applets de commande), vous permettent de gérer les ordinateurs de votre entreprise à partir de la ligne de commande. Les providers Windows PowerShell™ vous permettent d'accéder à des magasins de données, par exemple le Registre et le magasin de certificats, aussi facilement que si vous accédiez au système de fichiers. En outre, Windows PowerShell™ dispose d'un puissant analyseur d'expressions et d'un langage de script très complet. Windows PowerShell™ comprend les fonctionnalités suivantes :

- Cmdlets qui exécutent des tâches d'administration système courantes, par exemple la gestion du Registre, des services, des processus et des journaux d'événements, ainsi que l'utilisation de l'infrastructure WMI (Windows Management Instrumentation) ;
- Langage de script basé sur les tâches et une prise en charge des scripts et des outils en ligne de commande existants ;

- Conception cohérente. Dans la mesure où les magasins de données système et les cmdlets Windows PowerShell utilisent une syntaxe et des conventions d'affectation de noms communes, les données peuvent être partagées facilement ; en outre, la sortie d'une cmdlet peut servir d'entrée pour une autre cmdlet sans nouvelle mise en forme ou manipulation ;
- Navigation simplifiée au sein du système d'exploitation à l'aide de commandes, ce qui permet aux utilisateurs de naviguer dans le Registre et d'autres magasins de données de la même façon que dans le système de fichiers ;
- Puissantes fonctionnalités de manipulation d'objets. Les objets peuvent être manipulés directement ou envoyés vers d'autres outils ou bases de données ;
- Interface extensible. Les éditeurs de logiciels indépendants et les développeurs professionnels peuvent créer des outils et utilitaires personnalisés afin d'administrer leurs logiciels.

2 Installation / Lancement de PowerShell

Pour exécuter PowerShell, tout dépend de la version de Windows sur laquelle vous travaillez. Si vous utilisez Windows7 ou Windows Server 2008 R2, vous n'avez rien à faire, c'est pré-installer. Pour des versions antérieures, vous devez le télécharger et l'installer. Pour plus de détails, rendez vous sur la page officielle de Microsoft http://microsoft.com/powershell. Cette page contient les liens pour les installateurs appropriés ainsi que de la documentation et autres documents intéressants. Powershell est déjà installé sur les machines de l'IUT.

PowerShell se présente sous deux formes :

- La console PowerShell (PS) qui interprète les commandes en ligne (Figure 1 à gauche),
- Un environnement graphique de développement de scripts – PowerShell Integrated Scripting Environment (PS-ISE) (Figure 1 à droite).

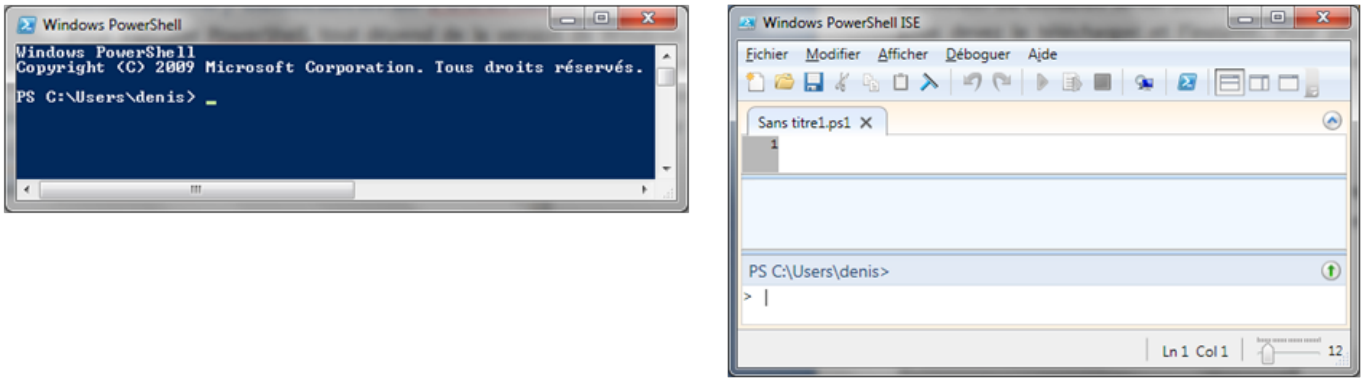


Figure 1. Interfaces de PowerShell

Pour lancer PS, aller dans le menu `Démarrer > Tous les Programmes > Accessoires > Windows PowerShell` et choisissez l'interface souhaitée. Vous pouvez également choisir entre une version 32 bits (x86) ou 64 bits. Il est également possible de lancer PS avec l'interpréteur classique de commandes de Windows (`cmd.exe`) avec la commande `powershell`.

Pour ceux qui ne connaîtraient pas la notion de ligne de commande, visitez les pages suivantes : http://fr.wikipedia.org/wiki/Interpr%C3%A9teur_de_commandes et http://fr.wikipedia.org/wiki/Cmd.exe.

Exercice 1 Exécutez la commande `dir` dans un environnement PowerShell et dans l'interpréteur de commandes classique de Windows (`cmd.exe`). Quelles sont les différences ?

Réponse :

L'utilisation de l'environnement de développement de scripts (PS-ISE) permet d'avoir plusieurs sessions simultanément ainsi que la possibilité de déboguer.

Exercice 2 Dans l'interpréteur de commandes Windows (`cmd.exe`), afficher l'aide (`help`) de la commande `Powershell`. Comment peut-on exécuter la commande `Get-ChildItem` de PowerShell tout en restant en ligne de commande Windows mais en minimisant la fenêtre courante ? Que fait l'instruction `Get-ChildItem` ?

Réponse :

3 Les commandlets de base

Comme vous venez de l'apercevoir précédemment, Windows PS introduit des commandes un peu spéciales appelées *cmdlet* (prononcer « command-let »). Une cmdlet est la plus petite unité rendant une fonctionnalité. Plutôt que d'être très complexe, le plupart des cmdlets sont simples et possèdent peu de propriétés associées. Une cmdlet s'utilise de la même façon qu'une commande classique. Elle n'est pas sensitive à la casse. Elles respectent un format bien précis `<verbe>-<action>`. Le verbe précise ce que fait la cmdlet en général alors que le nom précise sur quoi la cmdlet va agir. Par exemple, la cmdlet `Get-Variable` va récupérer une variable de PS et retourner sa valeur. En PS v3.0, il existe 303 cmdlets différentes.

Exercice 3 Testez la cmdlet `Get-Variable` sans préciser de variable précise. Quel est le résultat ? Quelle est la valeur de `PSHOME` ?

Réponse :

Vous trouverez ci-dessous un tableau des verbes les plus communs pour les cmdlets :

Verbe de la cmdlet	Signification
Add	Ajoute une instance d'un item
Clear	Supprime le contenu d'un item comme la valeur d'une variable
ConvertFrom / ConvertTo	Convertie un item d'un format à un autre, comme une liste de valeurs séparées par des virgules en des propriétés d'un objet
Disable / Enable	Annule / Autorise un certain paramétrage comme une connexion à distance
Export / Import	Exporte / Importe les propriétés d'un item dans un format particulier comme exporter les propriétés de la console en XML
Get	Interroge un objet comme obtenir la liste des processus
Invoke	Exécute une instance d'un item comme une expression
New / Remove	Crée / Supprime une nouvelle instance d'un item, comme une nouvelle variable ou événement
Set	Modifie les paramètres d'un objet
Start / Stop	Démarre / Arrête une instance d'un item comme un service ou un processus
Test	Test une instance d'un item pour une valeur spécifique comme tester une connexion pour savoir si elle est valide
Write	Exécute une opération d'écriture d'une instance d'un objet comme écrire un événement sur le gestionnaire de log d'événements

Afin d'avoir une idée plus précise de l'utilité des précédentes cmdlet, vous trouverez ci-dessous un tableau résumant les cmdlet souvent utilisés à des fins d'administration :

Verbe de la cmdlet	Signification
Add-Computer / Remove-Computer	Ajoute ou supprime l'appartenance d'un ordinateur dans un domaine ou groupe de travail
Checkpoint-Computer / Restore-Computer	Créé un point de restauration du système pour un ordinateur / restaure l'ordinateur
Compare-Object / Group-Object / Sort-Object / Select-Object / New-Object	Comparaison / groupement / trie / sélection / création d'objets
ConvertFrom-SecureString / ConvertTo-SecureString	Création / export de chaine sécurisées
Debug-Process	Déboguer un processus s'exécutant sur un ordinateur
Get-Alias / New-Alias / Set-Alias / Export-Alias / Import-Alias	Récupérer / créer / paramétrer / exporter / importer des alias
Get-AuthenticodeSignature / Set-AuthenticodeSignature	Récupérer / paramétrer la signature d'un objet associé à un fichier
Get-Command / Invoke-Command / Measure-Command / Trace-Command	Récupérer des informations sur / invoquer / mesurer le temps d'exécution / tracer des cmdlets
Get-EventLog / Write-EventLog / Clear-EventLog	Récupérer / écrire / effacer des événements de log
Get-ExecutionPolicy / Set-ExecutionPolicy	Traite de la politique d'exécution du shell courant
Get-Help	Devinez … ?
Get-Host	Récupère des informations de l'application hôte de PS
Get_HotFix	Récupère les modifications apportées à un ordinateur
Get-Location / Set-Location	Affiche ou sélectionne le répertoire courant
Get-Process/ Start-Process/ Stop-Process	Récupère / Démarre / Arrête un processus sur une machine
Get-PSDrive/ New-PSDrive/ Remove-PSDrive	Récupère / Crée / Supprime un disque spécifique PowerShell
Get-Service/ New-Service/ Set-Service	Récupère / Crée / Définit un service
Get-Variable/ New-Variable/ Set-Variable/ Remove-Variable/ Clear-Variable	Cmdlets pour la gestion des variables
Import-Counter/ Export-Counter	Importe / Exporte les données de compteur de performance

Read-Host/ Write-Host/ Clear-Host	Lecture dans / Ecriture dans / Efface la fenêtre hôte
Rename-Computer/ Stop-Computer/ Restart-Computer	Renomme / Arrête / Redémarre un ordinateur
Reset-ComputerMachinePassword	Réinitialise le mot de passe du compte de l'ordinateur

Exercice 4 Affichez l'aide disponible pour la cmdlet `remove-item`. Ensuite faites de même en incluant les exemples. Vous pouvez demander de l'aide sur l'aide ! Testez également la commande `help`. Quelles différences observez-vous ?

Réponse :

Si vous cherchez de l'aide pas forcément sur une instruction mais plutôt sur une instruction basique ou sur des concepts de base, utilisez l'instruction `Get-Help about` ou `help about`.

Exercice 5 Quelles sont les cmdlets PS pour exécuter ces tâches : (1) copier / (2) déplacer un fichier dans un répertoire différent du répertoire courant, (3) changer de répertoire, (4) renommer un fichier, (5) créer / supprimer (6) un répertoire, (7) supprimer un fichier, (8) afficher le contenu d'un fichier

Réponse :

La commande `Get-Help` donne de l'aide sur des choses pour lesquelles des fichiers d'aide sont disponibles dans PS. La commande `Get-Command` est complémentaire à `Get-Help` puisqu'elle donne de l'aide sur des choses qui peuvent être exécutées comme les cmdlets, les fonctions, les scripts, et même des fichiers Windows. Il est également possible d'afficher l'aide d'une seule commande disponible chez Microsoft en utilisant l'option `-online`.

Exercice 6 Exécutez la cmdlet `Get-Command ipconfig.exe | Format-List`.

Réponse :

Exercice 7 Il est également possible d'utiliser `Get-Command` pour découvrir les commandes possibles sur un thème particulier. Par exemple, vous ne vous souvenez plus des commandes liées aux processus. Saisissez la cmdlet `Get-Command *process*`. En utilisant l'aide, afficher que les fonctions.

Réponse :

Exercice 8 Récupérez la version de PS qui s'exécute sur votre machine ? Récupérer **uniquement la valeur** et pas toutes les informations (pour cela, utilisez l'aide) ? C'est un premier aperçu de la notion d'objets.

Réponse :

3.1 Les alias

Il est possible d'associer un « surnom » à une commande. Cela peut se faire à l'aide de la cmdlet `Set-Alias`.

Exercice 9 À quoi correspond l'alias `gal` et essayer de trouver l'alias de `Get-Content` ? Est-il possible d'avoir plusieurs alias pour la même commande ?

Réponse :

4 Configuration et Personnalisation de PowerShell

Par défaut, la console PS s'exécute avec un texte blanc sur fond bleu. Imaginons que cela ne vous convienne pas et que vous souhaitiez en changer. Tapez la ligne suivante pour changer la couleur de fond : `$Host.UI.RawUI.BackgroundColor="magenta"`. Saisissez ensuite l'instruction `clear`. Il est conseillé d'utiliser la touche `<TABULATION>` pour faire appel à la complétion et ainsi éviter de saisir un mot au complet ou encore de connaitre les différentes possibilités. Modifiez la couleur du texte en jaune en utilisant la propriété `ForegroundColor`.
`$Host` est une variable spéciale qui référence un objet représentant la console courante.

Exercice 10 La couleur vous parait horrible ! C'est normal, pour moi aussi. Fermez la console et relancer PS. Que constatez-vous ?

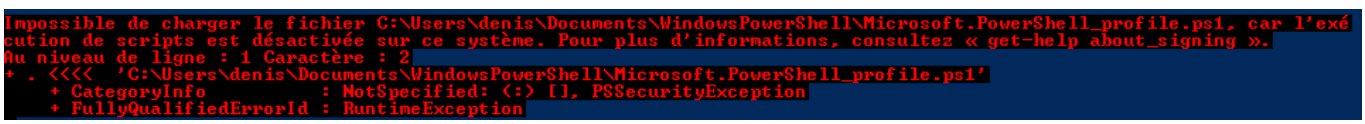
Réponse :

Comme vous ne souhaitez pas personnaliser votre console à chaque fois que vous en ouvrez une, il est possible de la personnaliser par défaut en modifiant votre *profile*. Les informations sur votre profile peuvent être contenues dans quatre fichiers différents. Pourquoi autant ? Les informations de configuration peuvent être appliquées pour tous les utilisateurs ou pour un seul. Vous pouvez exécuter PS en ligne de commande ou via un outil tiers. Les 4 fichiers correspondent aux différentes combinaisons possibles.

Exercice 11 Tapez la commande `$profile` pour savoir où se trouve votre profile. Par défaut, votre profile PS se trouve dans le fichier nommé `Microsoft.PowerShell_profile.ps1` dans le répertoire `WindowsPowerShell` de votre répertoire `MesDocuments`. Il se peut que ce fichier n'existe pas encore et dans ce cas, PS prend des valeurs par défaut. Créez ce fichier (même vide) dans le bon répertoire. Modifiez ce fichier avec la ligne suivante : `$Host.UI.RawUI.WindowTitle="Mon Windows PowerShell Personalisé!"`. Relancer PS.

Réponse :

Malheureusement, PowerShell s'exécute mais avec une erreur et l'écran suivant s'affiche :



Ceci est normal : par défaut, PS ne vous laissera pas exécuter de scripts (y compris votre profile) tant qu'il n'aura pas été signé par un certificat pour des raisons de sécurité. Ceci fera l'objet d'un travail ultérieur mais pour

l'instant, pour voir si votre profil fonctionne, il faut changer le comportement par défaut de PS en lui donnant la possibilité d'exécuter des scripts locaux avec la commande suivante : `Set-ExecutionPolicy RemoteSigned`. En acceptant la modification de la stratégie d'exécution, il se peut que vous ayez une autre erreur qui vous précise que vous n'avez pas accès à la base de registre. Pour résoudre ce problème, vous devez relancer PS en tant qu'administrateur (clic-droit sur le programme `Windows PowerShell` et choisir l'option `Exécuter en tant qu'administrateur`). Si vous souhaitez revenir comme avant, utiliser la valeur `Restricted` à la place de `RemoteSigned`.

Vous pouvez également modifier votre profile en saisissant la commande `Set-PSDebug -Strict` qui autorisera PS à se plaindre quand vous utiliserez une variable qui n'a pas été initialisée. Ensuite, saisissez la commande `$ErrorActionPreference = "stop"` qui autorisera PS à s'arrêter dés qu'un script contiendra une erreur. Pendant la saisie d'une commande, pensez à utiliser la touche `<TAB>` pour utiliser la complétion.

Pour information, PS-ISE possède également un profile contenu dans le script `Microsoft.PowerShellISE_profile.ps1` et l'objet correspondant à PS-ISE est `$psISE`.

4.1 Installation du PS Community Extension

Une librairie supplémentaire peut être installée en plus de PowerShell ; elle est disponible à cette adresse http://pscx.codeplex.com/. Elle n'est pas nécessaire pour utiliser PS mais elle permet d'ajouter des fonctionnalités assez puissantes. Par exemple, elle installe la possibilité de lancer PS n'importe où dans un explorateur Windows (« Open PowerShell here ») avec une instance de PowerShell ayant un répertoire de travail correspondant au répertoire sélectionné. Par ailleurs, il y a des fonctions qui traitent du presse-papier (`Get-Clipboard` & `Out-Clipboard`), ou encore l'envoi d'e-mails (`Send-SmtpMail`). Il est également possible de jouer avec des outils de synthèse vocale via `Out-Speech`.
Le seul inconvénient à l'installation de cette librairie est la lenteur au démarrage de PS.

4.2 Où sauvegarder mes scripts

Vous venez d'écrire un ou plusieurs scripts ou fonctions que vous souhaitez utilisez assez fréquemment. Reste à savoir où il faut les stocker. Il existe plusieurs façons de sauvegarder des scripts en PS.

La plus simple consiste à modifier votre profil et stocker les scripts à cet endroit. Toutefois, vous préférez peut être avoir des fichiers séparés pour vos scripts. Pour cela, il est possible d'ajouter les noms de fichiers au profile pour qu'ils soient téléchargés et accessibles. Il suffit d'ajouter la ligne suivante à votre profil :

```
. c:\foo\bar\myfunctions.ps1
```

Exercice 12 Une autre option consiste à créer un disque spécifique, connu uniquement de PS, pour toutes vos fonctions ou scripts. Écrivez la commande qui permet de le faire en utilisant `new-Psdrive`. Évitez toutes interactions avec l'utilisateur ; toutes les informations doivent se trouver dans la commande. Vous pourrez exécuter vos scripts en tapant `scripts:myscript.ps1` si le disque créé s'appelle `scripts`. Testez la création de ce disque en créant un script qui affiche les processus courants.

Réponse :

Exercice 13 Fermez PS. Relancer le et exécutez le dernier script créé. Que se passe-t-il ? Pourquoi ?

Réponse :

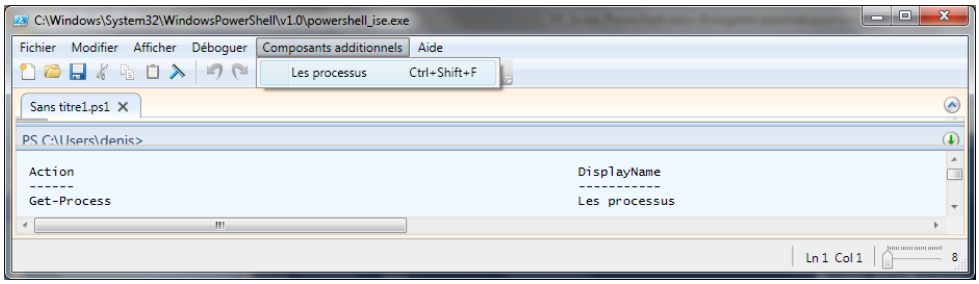
Exercice 14 Faites en sorte que le disque spécifique créé précédemment `scripts` soit connu à chaque lancement de PS (cf.). Choisissez bien le chemin que vous allez utiliser car nous allons l'utiliser dans le reste du cours.

Réponse :

4.3 Ajouter ses propres fonctions dans le menu de PS-ISE

Une des caractéristiques les plus intéressantes de PS-ISE est de pouvoir le personnaliser avec des commandes que nous utilisons souvent pour l'administration comme nous l'avons fait plus haut pour la ligne de commande. Il est donc possible de personnaliser les menus de PS-ISE. Pour cela, il suffit d'écrire sa cmdlet ou fonction et d'appliquer la méthode `Add` sur le contenu de la variable `CurrentPowerShellTab.AddOnsMenu.Submenus` (qui est une collection) de l'objet `psISE`.

Exercice 15 Modifier le profile de PS-ISE en ajoutant un sous-menu personnalisé qui ajoute une commande qui affiche les processus courants. Pour cela, utilisez la méthode `Add("Nom du sous-menu", {votre-Commande},"Raccourci-clavier")` sur la collection précédente afin d'obtenir le résultat ci-dessous :



Réponse :