

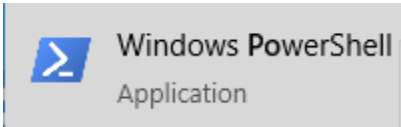
TP Powershell

POWERSHELL & SCRIPTS

TP 1

1. La console

Pour lancer la console Powershell : on recherche powershell dans la barre des tâches en bas à gauche, Windows PowerShell



Pour vérifier la bonne écriture d'une commande, on peut écrire le début de la commande suivi de la touche Tab, exemple : get-al[Tab]

```
PS C:\WINDOWS\system32> Get-Alias
```

Pour lister les éléments du dossier actif, il suffit d'écrire la commande avec un espace suivi de la touche Tab, exemple : Get-ChildItem [Tab]

```
PS C:\WINDOWS\system32> Get-ChildItem .\@AdvancedKeySettingsNotification.png
```

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

La saisie seule du caractère (-) permet de lister tous les paramètres possibles.

```
PS C:\WINDOWS\system32> Get-ChildItem .\@AdvancedKeySettingsNotification.png -Filter
```

2. Aide sur une commande

- Afficher l'aide sur la commande Get-Alias : Get-Help Get-Alias

La commande pour afficher de l'aide est : Get-Help

```

PS C:\WINDOWS\system32> Get-Help Get-Alias

NOM
    Get-Alias

RÉSUMÉ
    Gets the aliases for the current session.

SYNTAXE
    Get-Alias [-Definition <String[]>] [-Exclude <String[]>] [-Scope <String>] [<CommonParameters>]

    Get-Alias [-Name] <String[]> [-Exclude <String[]>] [-Scope <String>] [<CommonParameters>]

DESCRIPTION
    The Get-Alias cmdlet gets the aliases in the current session. This includes built-in aliases, aliases that you
    have set or imported, and aliases that you have added to your Windows PowerShell profile.

    By default, Get-Alias takes an alias and returns the command name. When you use the Definition parameter,
    Get-Alias takes a command name and returns its aliases.

    Beginning in Windows PowerShell 3.0, Get-Alias displays non-hyphenated alias names in an <alias> -> <definition>
    format to make it even easier to find the information that you need.

LIENS CONNEXES
    Online Version: http://go.microsoft.com/fwlink/?LinkId=821778
    About Aliases
    Export-Alias
    Import-Alias
    New-Alias
    Set-Alias

REMARQUES
    Pour consulter les exemples, tapez : "get-help Get-Alias -examples".
    Pour plus d'informations, tapez : "get-help Get-Alias -detailed".
    Pour obtenir des informations techniques, tapez : "get-help Get-Alias -full".
    Pour l'aide en ligne, tapez : "get-help Get-Alias -online"

```

- Afficher tous les alias dont le nom commence par la lettre g : Get-Alias g*

Pour faire ceci : Get-Alias lettre*

```

PS C:\WINDOWS\system32> Get-Alias g*

```

CommandType	Name	Version	Source
Alias	gal -> Get-Alias		
Alias	gbp -> Get-PSBreakpoint		
Alias	gc -> Get-Content		
Alias	gcb -> Get-Clipboard	3.1.0.0	Microsoft.PowerShell.Management
Alias	gci -> Get-ChildItem		
Alias	gcm -> Get-Command		
Alias	gcs -> Get-PSCallStack		
Alias	gdr -> Get-PSDrive		
Alias	ghy -> Get-History		
Alias	gi -> Get-Item		
Alias	gin -> Get-ComputerInfo	3.1.0.0	Microsoft.PowerShell.Management
Alias	gjb -> Get-Job		
Alias	gl -> Get-Location		
Alias	gm -> Get-Member		
Alias	gmo -> Get-Module		
Alias	gp -> Get-ItemProperty		
Alias	gps -> Get-Process		
Alias	gpv -> Get-ItemPropertyValue		
Alias	group -> Group-Object		
Alias	gsn -> Get-PSSession		
Alias	gsnp -> Get-PSSnapin		
Alias	gsv -> Get-Service		
Alias	gtz -> Get-TimeZone	3.1.0.0	Microsoft.PowerShell.Management
Alias	gu -> Get-Unique		
Alias	gv -> Get-Variable		
Alias	gwmi -> Get-WmiObject		

- Afficher la commande qui correspond à l'alias dont le nom est sl : Get-Alias sl

On utilise la commande : Get-Alias [nom]

```
PS C:\WINDOWS\system32> Get-Alias sl
```

CommandType	Name	Version	Source
Alias	sl -> Set-Location		

- Que fait la commande 'Get-process' ?

Cette commande permet d'afficher les processus en cours d'exécution sur un ordinateur

```
PS C:\WINDOWS\system32> Get-Process
```

Handles	NPM(K)	PM(K)	WS(K)	CPU(s)	Id	SI	ProcessName
397	23	11240	4416	0,22	5144	1	ApplicationFrameHost
372	16	3576	14828	0,09	5384	1	browser_broker
257	13	5896	10460	4,67	3196	1	conhost
330	14	1776	4496	0,55	396	0	csrss
322	16	1644	4680	0,72	472	1	csrss
393	15	4124	13332	0,61	4144	1	ctfmon
443	21	5736	15128	0,47	2352	0	dasHost
232	12	2588	12132	0,34	3844	1	dllhost
257	19	4052	13704	0,25	6036	1	dllhost
782	47	33800	42332	3,75	860	1	dwm
2015	72	33584	52644	16,70	1872	1	explorer
32	6	1688	3544	0,08	700	1	fontdrvhost
32	5	1304	2340	0,03	708	0	fontdrvhost
181	11	1764	184	0,02	552	0	GoogleCrashHandler
160	9	1736	212	0,02	5508	0	GoogleCrashHandler64

- A l'aide de la commande 'Get-PSDrive', afficher les informations du volume nommé C :

Get-PSDrive c

La commande pour faire ceci est : Get-PSDrive [volume]

```
PS C:\WINDOWS\system32> Get-PSDrive c
```

Name	Used (GB)	Free (GB)	Provider	Root	CurrentLocation
C	19,08	19,93	FileSystem	C:\	WINDOWS\system32

- Que fait la commande 'Get-command' ?

Cette commande permet d'obtenir des informations de base sur les applets de commande et d'autres éléments des commandes Windows PowerShell de la session, tels qu'alias, fonctions, filtres, scripts et applications.

```
PS C:\WINDOWS\system32> Get-command
```

CommandType	Name	Version	Source
Alias	Add-AppPackage	2.0.1.0	Appx
Alias	Add-AppPackageVolume	2.0.1.0	Appx
Alias	Add-AppProvisionedPackage	3.0	Dism
Alias	Add-ProvisionedAppPackage	3.0	Dism
Alias	Add-ProvisionedAppxPackage	3.0	Dism
Alias	Add-ProvisioningPackage	3.0	Provisioning
Alias	Add-TrustedProvisioningCertificate	3.0	Provisioning

- Que fait la commande 'Get-help restart_service' ?

Cette commande permet d'obtenir de l'aide sur la commande restart_service

```
PS C:\WINDOWS\system32> Get-help Restart-Service

NOM
    Restart-Service

RÉSUMÉ
    Stops and then starts one or more services.

SYNTAXE
    Restart-Service [-Confirm] -DisplayName <String[]> [-Exclude <String[]>] [-Force] [-Include <String[]>]
    [-PassThru] [-WhatIf] [<CommonParameters>]

    Restart-Service [-InputObject] <ServiceController[]> [-Confirm] [-Exclude <String[]>] [-Force] [-Include
    <String[]>] [-PassThru] [-WhatIf] [<CommonParameters>]

    Restart-Service [-Name] <String[]> [-Confirm] [-Exclude <String[]>] [-Force] [-Include <String[]>] [-PassThru]
    [-WhatIf] [<CommonParameters>]

DESCRIPTION
    The Restart-Service cmdlet sends a stop message and then a start message to the Windows Service Controller for a
    specified service. If a service was already stopped, it is started without notifying you of an error. You can
    specify the services by their service names or display names, or you can use the InputObject parameter to pass an
    object that represents each service that you want to restart.

LIENS CONNEXES
    Online Version: http://go.microsoft.com/fwlink/?LinkId=821629
    Get-Service
    New-Service
    Resume-Service
    Set-Service
    Start-Service
    Stop-Service
    Suspend-Service

REMARQUES
    Pour consulter les exemples, tapez : "get-help Restart-Service -examples".
    Pour plus d'informations, tapez : "get-help Restart-Service -detailed".
    Pour obtenir des informations techniques, tapez : "get-help Restart-Service -full".
    Pour l'aide en ligne, tapez : "get-help Restart-Service -online"
```

3. Pipeline

Exercice 1 :

On crée un répertoire où à l'intérieur on met des fichiers test. Sur la console powershell pour se placer dans ce répertoire on utilise la commande : cd [chemin]

Dans mon cas : cd C:\Users\Windows\Desktop\Test

```
PS C:\Users\Windows\Desktop> cd C:\Users\Windows\Desktop\Test

PS C:\Users\Windows\Desktop\Test>
```

On déclare la variable nommée « variable » avec la commande : $[\text{nom_variable}] = \text{Get-ChildItem}$

Dans mon cas : $\$variable = \text{Get-ChildItem}$

```
PS C:\Users\Windows\Desktop\Test> $variable = Get-ChildItem
```

Ensuite, on renvoie le résultat de Get-ChildItem dans la variable nommée « variable » sous forme de tableau : \$variable

```
PS C:\Users\Windows\Desktop\Test> $variable

Répertoire : C:\Users\Windows\Desktop\Test

Mode                LastWriteTime         Length Name
----                -
-a----            24/03/2020   15:33             0 test.txt
-a----            24/03/2020   15:34             0 TEST2.txt
```

Pour lire la 1^{er} ligne : `$(nom_variable)[ligne_tableau]`

\$variable[1]

```
PS C:\Users\Windows\Desktop\Test> $variable[1]

Répertoire : C:\Users\Windows\Desktop\Test

Mode                LastWriteTime         Length Name
----                -
-a----            24/03/2020   15:34             0 TEST2.txt
```

Pour renvoyer ce résultat à Get-Member et obtenir la liste des propriétés et méthode applicable à l'objet `$variable[1]` | `$(nom_variable)[ligne_tableau]` : `Get-Member`

\$variable[1] | Get-Member

```
PS C:\Users\Windows\Desktop\Test> $variable[1] | Get-Member

TypeName : System.IO.FileInfo

Name      MemberType Definition
-----
LinkType  CodeProperty System.String LinkType{get=GetLinkType;}
Mode      CodeProperty System.String Mode{get=Mode;}
Target    CodeProperty System.Collections.Generic.IEnumerable`1[[System.String, mscorlib, Version=4.0.0.0, Culture=neutra...
AppendText Method      System.IO.StreamWriter AppendText()
CopyTo    Method      System.IO.FileInfo CopyTo(string destFileName), System.IO.FileInfo CopyTo(string destFileName, boo...
Create     Method      System.IO.FileStream Create()
CreateObjRef Method      System.Runtime.Remoting.ObjRef CreateObjRef(type requestedType)
CreateText Method      System.IO.StreamWriter CreateText()
Decrypt   Method      void Decrypt()
Delete    Method      void Delete()
```

En utilisant la propriété `root` qui figure dans la liste on obtient comme résultat le mode d'accès autorisé, la date dernière écriture et le répertoire : `$(nom_variable)[ligne_tableau].root`

\$variable[1].root

```
PS C:\Users\Windows> $variable[1].root

Mode                LastWriteTime         Length Name
----                -
d--hs-            24/03/2020   14:16             C:\
```

Exercice 2 :

Nous allons lister les fichiers du répertoire précédents qui ont pour extension .txt :

Get-Childitem * -Include *.txt

```
PS C:\Users\Windows\Desktop\Test> Get-ChildItem * -Include *.txt

Répertoire : C:\Users\Windows\Desktop\Test

Mode                LastWriteTime         Length Name
----                -
-a----            24/03/2020   16:00             9 test.txt
-a----            24/03/2020   16:01            15 TEST2.txt
```

Exercice 3 :

Maintenant, nous allons lister les fichiers du répertoire dans l'ordre de la dernière modification :

Get-Childitem | Sort-Object -Property LastWriteTime -Descending | more

```
PS C:\Users\Windows\Desktop\Test> Get-ChildItem | Sort-Object -Property LastWriteTime -Descending | more

Répertoire : C:\Users\Windows\Desktop\Test

Mode                LastWriteTime         Length Name
----                -
-a----            24/03/2020   16:01            15 TEST2.txt
-a----            24/03/2020   16:00             9 test.txt
```

4. Scripts

Exercice 4 :

Pour connaître le niveau d'autorisation pour l'exécution de scripts : Get-ExecutionPolicy

```
PS C:\Users\Windows\Desktop\Test> Get-ExecutionPolicy
Restricted
```

Il existe 6 modes de sécurité :

- Restricted : Cette valeur qui est la valeur par défaut, empêche l'exécution de scripts.
- AllSigned : Requiert la signature numérique par un éditeur de l'ensemble des scripts, y compris ceux que vous créez en local sur la machine.
- RemoteSigned : Requiert la signature numérique par un éditeur des scripts téléchargés à partir d'internet.
- Unrestricted : Exécute tous les scripts, mais vous invite à autoriser l'exécution des scripts non signés téléchargés via internet.
- Bypass : Pas de blocage, ni d'avertissements, tout sera exécuté.
- Undefined : Supprime la stratégie d'exécution appliquée, mais pas dans le cas où elle est définie par une stratégie de groupe.

Pour changer la sécurité et passer en RemoteSigned : Set-ExecutionPolicy RemoteSigned

```
PS C:\Users\Windows\Desktop\Test> Set-ExecutionPolicy RemoteSigned

Modification de la stratégie d'exécution
La stratégie d'exécution permet de vous prémunir contre les scripts que vous jugez non fiables. En modifiant la stratégie d'exécution, vous
vous exposez aux risques de sécurité décrits dans la rubrique d'aide about_Execution_Policies à l'adresse
https://go.microsoft.com/fwlink/?LinkID=135170. Voulez-vous modifier la stratégie d'exécution ?
[O] Oui [T] Oui pour tout [N] Non [U] Non pour tout [S] Suspendre [?] Aide (la valeur par défaut est « N ») : o
PS C:\Users\Windows\Desktop\Test>
```

On vérifie la prise en compte du changement : Get-ExecutionPolicy

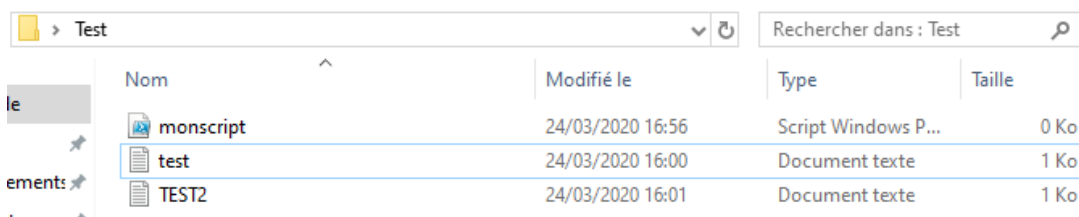
```
PS C:\Users\Windows\Desktop\Test> Get-ExecutionPolicy
RemoteSigned
```

5. Exécuter un script Powershell

On ouvre une invite de commande et on exécute la commande suivante : powershell [chemin.ps1]

Dans mon cas : powershell C:\Users\Windows\Desktop\Test\monscript.ps1

Il faut auparavant créer le script dans le repertoire, le script pour le moment est vide :



Nom	Modifié le	Type	Taille
monscript	24/03/2020 16:56	Script Windows P...	0 Ko
test	24/03/2020 16:00	Document texte	1 Ko
TEST2	24/03/2020 16:01	Document texte	1 Ko

L'invite de commande ne renvoie rien car le script est vide :

```
C:\Users\Windows>powershell C:\Users\Windows\Desktop\Test\monscript.ps1
```

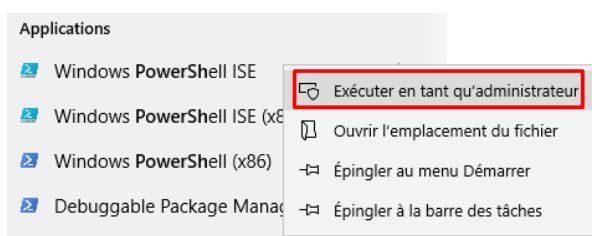
Exercice 5 :

Pour vérifier l'existence d'un dossier : Test-Path C:Windows

```
PS C:\Users\Windows\Desktop\Test> Test-Path C:Windows
False
```

Exercice 6 :

Pour lancer la console Powershell : on recherche powershell dans la barre des tâches en bas à gauche, clic droit sur Windows PowerShell ISE > Exécuter en tant qu'administrateur :

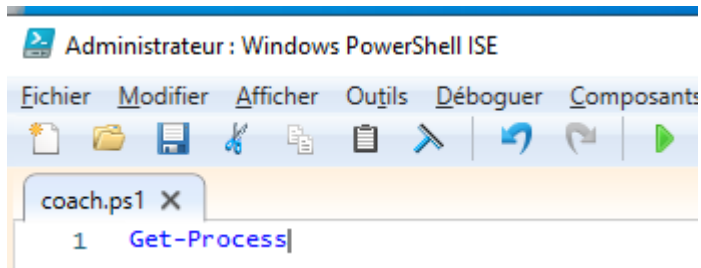


Pour connaître le niveau d'autorisation pour l'exécution de scripts : Get-ExecutionPolicy

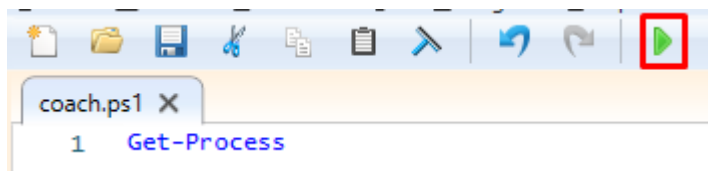
Ici Restricted

```
PS C:\Users\Windows\Desktop\Test> Get-ExecutionPolicy
Restricted
```

On créer un script nommé coach.ps1 :



On lance le script avec la flèche verte :



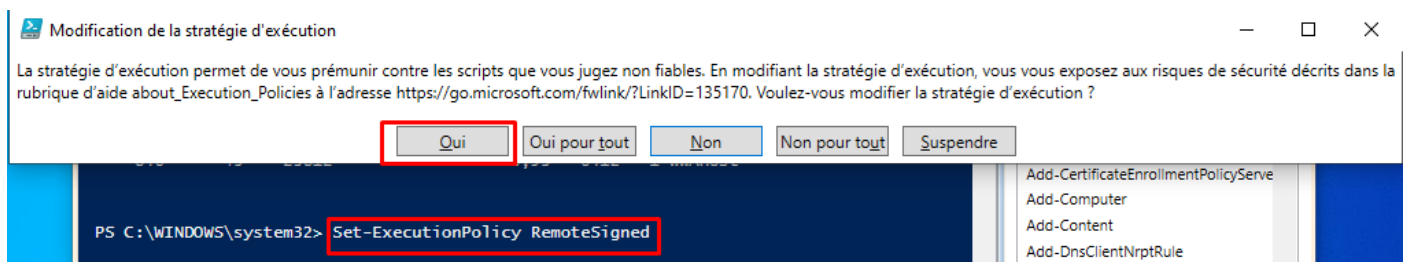
Le résultat du script est le suivant :

```
PS C:\WINDOWS\system32> C:\Users\Windows\Desktop\Test\coach.ps1
```

Handles	NPM(K)	PM(K)	WS(K)	CPU(s)	Id	SI	ProcessName
400	23	11248	7748	0,23	5144	1	ApplicationFrameHost
368	16	3544	2244	0,09	5384	1	browser_broker
338	13	1732	1456	0,89	396	0	csrss
354	16	1660	1580	1,31	472	1	csrss
412	16	5120	8484	1,52	4144	1	ctfmon
435	20	5660	2452	0,52	2352	0	dasHost
234	12	2520	4344	0,39	3844	1	dllhost
262	18	4152	7164	0,28	6036	1	dllhost
807	50	34680	37888	6,66	860	1	dwm
2239	85	43272	71584	27,84	1872	1	explorer
32	6	1748	1232	0,08	700	1	fontdrvhost
32	5	1276	184	0,03	708	0	fontdrvhost
181	11	1764	136	0,02	552	0	GoogleCrashHandler
160	9	1712	104	0,03	5508	0	GoogleCrashHandler64
0	0	60	8		0	0	Idle

Pour modifier la stratégie d'exécution : Set-ExecutionPolicy RemoteSigned

Et on valide le popup pour modifier la stratégie :



TP 2

La technologie WMI (Windows Management Instrumentation) est essentielle pour l'administration du système Windows, car elle expose une large gamme d'informations de façon uniforme. En raison du potentiel offert par WMI, l'applet de commande Windows PowerShell qui permet d'accéder aux objets WMI, `Get-WmiObject`, est l'une des plus utiles pour effectuer un véritable travail.

1. Liste des ressources

La commande : `Get-WmiObject -List` permet d'obtenir les noms des classes WMI dans l'espace de noms du référentiel WMI spécifié par le paramètre `Namespace` .

```
PS C:\Users\Windows> Get-WmiObject -List

Namespace : ROOT\cimv2

Name           Methods           Properties
-----
CIM_Indication    {}                {CorrelatedIndications, IndicationFilterName, IndicationIde...
CIM_ClassIndication {}                {ClassDefinition, CorrelatedIndications, IndicationFilterNa...
CIM_ClassDeletion {}                {ClassDefinition, CorrelatedIndications, IndicationFilterNa...
CIM_ClassCreation {}                {ClassDefinition, CorrelatedIndications, IndicationFilterNa...
CIM_ClassModification {}               {ClassDefinition, CorrelatedIndications, IndicationFilterNa...
CIM_InstIndication {}                {CorrelatedIndications, IndicationFilterName, IndicationIde...
CIM_InstCreation  {}                {CorrelatedIndications, IndicationFilterName, IndicationIde...
CIM_InstModification {}               {CorrelatedIndications, IndicationFilterName, IndicationIde...
CIM_InstDeletion  {}                {CorrelatedIndications, IndicationFilterName, IndicationIde...
__NotifyStatus    {}                {StatusCode}
```

2. Utilisation de ressources

Cette commande permet d'afficher la configuration des adaptateurs réseau de la machines :

```
PS C:\Users\Windows> Get-WmiObject -Class Win32_NetworkAdapterConfiguration

DHCPEnabled      : True
IPAddress        :
DefaultIPGateway :
DNSDomain        :
ServiceName      : kdnic
Description      : Microsoft Kernel Debug Network Adapter
Index            : 0
.
DHCPEnabled      : True
IPAddress        : {192.168.1.17, fe80::908a:8fa:1bbc:f1a}
DefaultIPGateway : {192.168.1.254}
DNSDomain        : lan
ServiceName      : E1G60
Description      : Intel(R) PRO/1000 MT Desktop Adapter
Index            : 1
```

Cette commande avec l'option `Filter` permet de filtrer les résultats de la commande précédente. Dans ce cas, elle permet de filtrer les adaptateurs réseaux avec une adresse IP activée.

```
PS C:\Users\Windows> Get-WmiObject -Class Win32_NetworkAdapterConfiguration -Filter IPEnabled=TRUE

DHCPEnabled      : True
IPAddress        : {192.168.1.17, fe80::908a:8fa:1bbc:f1a}
DefaultIPGateway : {192.168.1.254}
DNSDomain        : lan
ServiceName      : E1G60
Description      : Intel(R) PRO/1000 MT Desktop Adapter
Index            : 1
```

La commande : `Get-WmiObject -Class Win32_PingStatus -Filter "Address='192.168.1.101'"` permet d'avoir le statut d'un ping avec comme filtre l'adresse IP 192.168.1.101

```
PS C:\Users\Windows> Get-WmiObject -Class Win32_PingStatus -Filter "Address='192.168.1.101'"
```

Source	Destination	IPV4Address	IPV6Address	Bytes	Time(ms)
DESKTOP-K1...	192.168.1.101			32	0

La commande :

```
$ips = 1..254 | ForEach-Object -Process {"192.168.1." + $_ }
```

```
Get-WmiObject -Class Win32_PingStatus -Filter "Address='$ips'"
```

Permet de retrouver les informations réseaux de la machine.

```
PS C:\WINDOWS\system32> $ip = 1..254 | ForEach-Object -Process {"192.168.1." + $_}
Get-WmiObject -Class Win32_PingStatus -Filter "Address='$ip'"
```

Source	Destination	IPV4Address	IPV6Address
DESKTOP-K1...		192.168.1.17	fe80::908a:8fa:1bbc:f1a%3

Cette commande :

```
Get-WmiObject -Class Win32_NetworkAdapterConfiguration -Filter "DHCPEnabled=True"
```

permet de filtrer les adaptateurs réseaux avec le DHCP activé.

```
PS C:\Users\Windows> Get-WmiObject -Class Win32_NetworkAdapterConfiguration -Filter "DHCPEnabled=True"
```

```
DHCPEnabled      : True
IPAddress        :
DefaultIPGateway :
DNSDomain        :
ServiceName      : kdnic
Description      : Microsoft Kernel Debug Network Adapter
Index            : 0

DHCPEnabled      : True
IPAddress        : {192.168.1.17, fe80::908a:8fa:1bbc:f1a}
DefaultIPGateway : {192.168.1.254}
DNSDomain        : lan
ServiceName      : E1G60
Description      : Intel(R) PRO/1000 MT Desktop Adapter
Index            : 1
```

3. Utilisation de ressources

Écrire un script permettant d'afficher les informations de la machine locale.

Nom de la machine, configuration réseau, nom de l'utilisateur, etc...

Pour le nom de la machine : `[Environment]::MachineName`

```
PS C:\Users\Windows> [Environment]::MachineName
DESKTOP-K1IEEGA
```

Pour la configuration réseau :

Get-NetAdapter : permet de lister les différentes cartes réseaux physiques au sein de votre machine

```
PS C:\Users\Windows> Get-NetAdapter
```

Name	InterfaceDescription	ifIndex	Status	MacAddress	LinkSpeed
Ethernet	Intel(R) PRO/1000 MT Desktop Adapter	3	Up	08-00-27-67-87-6C	1 Gbps

Get-WmiObject -Class Wmi32 NetworkAdapyerConfiguration :

```
PS C:\Users\Windows> Get-WmiObject -Class Win32_NetworkAdapterConfiguration
```

```
DHCPEnabled      : True
IPAddress        :
DefaultIPGateway :
DNSDomain        :
ServiceName      : kdnic
Description      : Microsoft Kernel Debug Network Adapter
Index            : 0

DHCPEnabled      : True
IPAddress        : {192.168.1.17, fe80::908a:8fa:1bbc:f1a}
DefaultIPGateway : {192.168.1.254}
DNSDomain        : lan
ServiceName      : E1G60
Description      : Intel(R) PRO/1000 MT Desktop Adapter
Index            : 1
```

Pour le nom d'utilisateur : [Environment]::UserName

```
PS C:\Users\Windows> [Environment]::UserName
Windows
```

4. Utilisation de service

4.1. Exemple de script 1/3

```
1. $serveur = "monserveursmtp.domaine.com"
2. $expediteur = "expe@domaine.com"
3. $destinataire = "dest@domaine.com"
4. $objet = "Objet du mail "
5. $texte = " Texte : Message "
6. $message = new-object System.Net.Mail.MailMessage $expediteur, $destinataire,
   $objet, $texte
7. $client = new-object System.Net.Mail.SmtpClient $serveur
8. $client.Credentials =
   [System.Net.CredentialCache]::DefaultNetworkCredentials;$client.Send($message)
```

La première ligne permet d'intégrer ce qu'il y a entre les guillemets dans la variable \$serveur.

Le caractère \$ permet de déclarer une variable.

Ce script permet d'envoyer des mails.

4.2. Exemple de script 1/3

```
#
$tabsservices = "pare-feu windows", "service partage réseau du lecteur windows media"
foreach ($strservices in $tabsservices)
{
Write-Host "Démarrage $strservices ..."
Start-Service -Name $strservices
}
}
```

Ce script permet de démarrer le service Pare-feu Windows et service partage réseau du lecteur Windows media.

```
#
$tabsservices = "pare-feu windows", "service partage réseau du lecteur windows media"
foreach ($strservices in $tabsservices)
{
Write-Host "Arrêt $strservices ..."
Stop-Service -Name $strservices
}
}
```

Ce script permet l'arrêt du service Pare-feu Windows et service partage réseau du lecteur Windows media.

```
#
$tabsservices = "MpsSvc", "WMPNetworkSvc"
$strstartuptype = "disabled"
foreach ($strservices in $tabsservices)
{
Write-Host "Modification $strservices ..."
Set-Service $strservices -startuptype $strstartuptype
}
}
```

Ce script permet de désactiver le MpsSvc et WMPNetworkSvc.

L'intérêt pour un administrateur réseau de travailler avec ces scripts est d'éviter de taper chaque commande une par une pour interagir avec des services.

5. A vous d'écrire !

Script 1 : Script qui affiche le nom de l'utilisateur courant ainsi que les informations la machine

The image shows a PowerShell script named 'script1.ps1' in a Notepad window. The script contains 15 lines of code that use 'Write-Host' to display system information. Below the script, a terminal window shows the execution of the script from the command prompt. The output matches the code, displaying system details such as computer name, processor information, and user details.

```
script1.ps1 X
1 echo "*****"
2 echo "*Les informations du système sont les suivantes*"
3 echo "*****"
4 echo ""
5 echo ""
6 Write-Host "Nom de l'ordinateur : "$env:ComputerName
7 Write-Host "Processor ID : "$env:PROCESSOR_IDENTIFIER
8 Write-Host "Processor Niveau : "$env:Processor_level
9 Write-Host "Processor Architect : "$env:Processor_Architecture
10 Write-Host "Sytem Name : "(Get-WmiObject Win32_OperatingSystem).Caption
11 echo ""
12 Write-Host "Nom utilisateur : "$env:UserName
13 Write-Host "Chemin par défaut : "$env:windir
14 Write-Host "Fichier Temporaire : "$env:Temp
15 Write-Host "Application local : "$env:AppData

PS C:\WINDOWS\system32> C:\Users\Windows\Desktop\script1.ps1
*****
*Les informations du système sont les suivantes*
*****

Nom de l'ordinateur : DESKTOP-K1IEEGA
Processor ID : Intel64 Family 6 Model 158 Stepping 9, GenuineIntel
Processor Niveau : 6
Processor Architect : AMD64
Sytem Name : Microsoft Windows 10 Professionel
-----
Nom utilisateur : Windows
Chemin par défaut : C:\WINDOWS
Fichier Temporaire : C:\Users\Windows\AppData\Local\Temp
Application local : C:\Users\Windows\AppData\Roaming
```

Script 2 : Script qui affiche la liste des dossiers partagés, la date du jour, l'utilisateur et le mot de passe.

```
script1.ps1  script2.ps1 X
1  echo "*****"
2  echo "*Gestion de données*"
3  echo "*****"
4  echo ""
5  echo ""
6  $share=Get-WmiObject Win32_Share
7  echo "La liste des dossiers partagés sont : "$share
8  echo ""
9  echo "-----"
10 echo ""
11 $date=Get-Date -Format "dddd dd MMMM yyyy"
12 $date2=Get-Date -Format "dd/MM/yyyy"
13 echo "La date du jour est : "$date
14 $date2
15 echo ""
16 echo "-----"
17 echo ""
18 $password=Get-Credential
19 echo "Le nom d'utilisateur est : "$env:UserName
20 echo "Le mot de passe est      : "$password.GetNetworkCredential().Password
```

```
PS C:\WINDOWS\system32> C:\Users\Windows\Desktop\script2.ps1
*****
*Gestion de données*
*****

La liste des dossiers partagés sont :

Name   Path      Description
----   -
ADMIN$ C:\WINDOWS Administration à distance
C$     C:\       Partage par défaut
IPC$   IPC$      IPC distant

-----

La date du jour est :
jeudi 26 mars 2020
26/03/2020

-----

Le nom d'utilisateur est :
Windows
Le mot de passe est      :
*****
```

Script 3 : Script qui permet d'afficher le mois en fonction du nombre qu'on écrit entre 1 et 12.

```
script1.ps1  script2.ps1  script3.ps1 X
1  echo "*****"
2  echo "*Opération sur la date*"
3  echo "*****"
4  echo ""
5  $mois = Read-Host "Indiquez un nombre entre 1 et 12"
6  if ($mois -eq 1)
7  {Write-Host "Le nombre 1 correspond au mois de Janvier"}
8  elseif ($mois -eq 2)
9  {Write-Host "Le nombre 2 correspond au mois de Février"}
10 elseif ($mois -eq 3)
11 {Write-Host "Le nombre 3 correspond au mois de Mars"}
12 elseif ($mois -eq 4)
13 {Write-Host "Le nombre 4 correspond au mois de Avril"}
14 elseif ($mois -eq 5)
15 {Write-Host "Le nombre 5 correspond au mois de Mai"}
16 elseif ($mois -eq 6)
17 {Write-Host "Le nombre 6 correspond au mois de Juin"}
18 elseif ($mois -eq 7)
19 {Write-Host "Le nombre 7 correspond au mois de Juillet"}
20 elseif ($mois -eq 8)
21 {Write-Host "Le nombre 8 correspond au mois de Août"}
22 elseif ($mois -eq 9)
23 {Write-Host "Le nombre 9 correspond au mois de Septembre"}
24 elseif ($mois -eq 10)
25 {Write-Host "Le nombre 10 correspond au mois de Octobre"}
26 elseif ($mois -eq 11)
27 {Write-Host "Le nombre 11 correspond au mois de Novembre"}
28 elseif ($mois -eq 12)
29 {Write-Host "Le nombre 12 correspond au mois de Décembre"}
30 elseif ($mois -le 0)
31 {Write-Host "Veuillez écrire un nombre compris entre 1 et 12 uniquement"}
32 elseif ($mois -ge 13)
33 {Write-Host "Veuillez écrire un nombre compris entre 1 et 12 uniquement"}
```

```
PS C:\WINDOWS\system32> C:\Users\Windows\Desktop\script3.ps1
*****
*Opération sur la date*
*****

Indiquez un nombre entre 1 et 12 : 1
Le nombre 1 correspond au mois de Janvier

PS C:\WINDOWS\system32> C:\Users\Windows\Desktop\script3.ps1
*****
*Opération sur la date*
*****

Indiquez un nombre entre 1 et 12 : 5
Le nombre 5 correspond au mois de Mai

PS C:\WINDOWS\system32> C:\Users\Windows\Desktop\script3.ps1
*****
*Opération sur la date*
*****

Indiquez un nombre entre 1 et 12 : 12
Le nombre 12 correspond au mois de Décembre

PS C:\WINDOWS\system32> C:\Users\Windows\Desktop\script3.ps1
*****
*Opération sur la date*
*****

Indiquez un nombre entre 1 et 12 : 0
Veuillez écrire un nombre compris entre 1 et 12 uniquement

PS C:\WINDOWS\system32> C:\Users\Windows\Desktop\script3.ps1
*****
*Opération sur la date*
*****

Indiquez un nombre entre 1 et 12 : 13
Veuillez écrire un nombre compris entre 1 et 12 uniquement
```