Powershell

Powershell est le successeur de cmd dans l'administration de Windows à la ligne de commande. Plutôt que d'utiliser des exécutables individuels ayant tous des structures différentes, Microsoft a tenté d'unifier la structure des commandes ainsi que les résultats de ces commandes.

https://learn.microsoft.com/fr-ca/powershell/

- Base
  - Types d'objets de base
  - Manipulation d'objets et commandes communes
  - Manipulation de chemins d'accès pour objets Powershell
- Administration de Windows
  - Utilisateurs et groupes
  - Applications et fonctionnalités
  - Registre Windows
- Modules supplémentaires
  - Mises à jour Windows
  - Winget

Base

Types d'objets de base

History

D'abord et avant tout, je vous suggère fortement de vous familiariser avec le type d'objet History. Il s'agit de l'historique des commandes que vous avez saisi au cours d'une session Powershell. Ces objets vous aideront à créer des scritps, à réviser vos commandes, etc.

Ces commandes sont écrites dans le fichier "%AppData%\Microsoft\Windows\PowerShell\PSReadLine"

Ces objets sont consultables à l'aide de la commande "Get-History" et sont composés des propriétés suivantes :

```
Id                : 7
CommandLine       : $mon_message = "Hello world!"
ExecutionStatus   : Completed
StartExecutionTime : 2025-01-18 4:20:04 PM
EndExecutionTime   : 2025-01-18 4:20:04 PM

Id                : 8
CommandLine       : Write-Host $mon_message
ExecutionStatus   : Completed
StartExecutionTime : 2025-01-18 4:20:10 PM
EndExecutionTime   : 2025-01-18 4:20:10 PM
```

Vous pourriez donc facilement à partir de cet historique créer un script basé sur les manipulations que vous avez fait lors de votre session si vous sélectionnez seulement la ligne de commande comme ceci :

```
PS C:\Users\alexa> Get-History | Select-Object -Property CommandLine


CommandLine
-----------
clear-history
$mon_message = "Hello world!"
Write-Host $mon_message
Get-History
```

```
Get-History | Select-Object -Property *
```

ID

L'identifiant de l'objet détermine l'ordre dans lequel les commandes ont été exécutées.

CommandLine

La commande exécutée.

ExecutionStatus

L'état de la commande à savoir si elle est en cours d'exécution (ex. un ping continu), si elle a échoué ou si elle a été complétée.

StartExecutionTime

La date et l'heure de l'exécution de la commande.

EndExecutionTime

La date et l'heure de la fin d'exécution de la commande, qu'elle ait été annulée, qu'elle ait échouée ou qu'elle se soit complétée.

Item et ChildItem

Un ChildItem peut représenter plusieurs types d'information. Ce type d'objet est généralement utilisé pour manipuler des fichiers sur un espace de stockage mais peut aussi être utilisé pour modifier des clés de registre.

On peut déduire du nom des types d'objets qu'un Item sélectionne seulement l'objet demandé tandis qu'un ChildItem sélectionnera les objets situés à l'intérieur de l'objet. Ces deux types d'objets possèdent les mêmes propriétés et peuvent être manipulés de la même façon.

```
PS C:\Users\alexa> Get-Item D:\Item | Select-Object -Property *



PSPath            : Microsoft.PowerShell.Core\FileSystem::D:\Item
PSParentPath      : Microsoft.PowerShell.Core\FileSystem::D:\
PSChildName       : Item
PSDrive           : D
PSProvider        : Microsoft.PowerShell.Core\FileSystem
PSIsContainer     : True
Mode              : d-----
BaseName          : Item
```

```
Target           : {}
LinkType         :
Name             : Item
FullName         : D:\Item
Parent           :
Exists           : True
Root             : D:\
Extension        :
CreationTime     : 2025-01-18 4:35:18 PM
CreationTimeUtc  : 2025-01-18 9:35:18 PM
LastAccessTime   : 2025-01-18 4:35:27 PM
LastAccessTimeUtc : 2025-01-18 9:35:27 PM
LastWriteTime    : 2025-01-18 4:35:26 PM
LastWriteTimeUtc  : 2025-01-18 9:35:26 PM
Attributes       : Directory
```

Lorsque l'on récupère les "ChildItem", il est aussi possible d'ajouter le paramètre "-recurse" pour sélectionner récursivement les objets dans les sous-dossiers ou sous-clés de registre.

```
PS C:\Users\alexa> Get-ChildItem D:\Item


    Directory: D:\Item


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        2025-01-18   4:40 PM                Subdirectory
-a----        2025-01-18   4:35 PM              0 ChildItem-1.txt
-a----        2025-01-18   4:38 PM              0 ChildItem-2.txt



PS C:\Users\alexa> Get-ChildItem D:\Item -Recurse


    Directory: D:\Item


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        2025-01-18   4:40 PM                Subdirectory
-a----        2025-01-18   4:35 PM              0 ChildItem-1.txt
-a----        2025-01-18   4:38 PM              0 ChildItem-2.txt
```

```
    Directory: D:\Item\Subdirectory


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        2025-01-18   4:40 PM              0 SubChildItem.txt
```

L'exemple suivant servirait à consulter le registre HKEY_CURRENT_USER.

```
PS C:\Users\alexa> Get-ChildItem HKCU:/Item


    Hive: HKEY_CURRENT_USER\Item

Name                           Property
----                           --------
ChildItem                      Value : Content
```

Content

Tout comme en Linux la commande "cat" nous permet de lire le contenu d'un fichier et la redirection de sortie > et >> nous permet d'ajouter ou modifier le contenu du fichier, le type d'objet "Content" en Powershell nous permet de poser des actions similaires. Ce sont ici les commandes Get-Content, Clear-Content, Add-Content et Set-Content qui nous intéressent.

```
# Contenu initial du fichier
PS C:\Users\alexa> Get-Content D:\Item\ChildItem-1.txt
Hello world!

# Contenu vidé du fichier
PS C:\Users\alexa> Clear-Content D:\Item\ChildItem-1.txt
PS C:\Users\alexa> Get-Content D:\Item\ChildItem-1.txt

# Contenu ajouté au fichier
PS C:\Users\alexa> Add-Content -Path D:\Item\ChildItem-1.txt -Value "Bonjour le monde!"
PS C:\Users\alexa> Get-Content D:\Item\ChildItem-1.txt
Bonjour le monde!

# Contenu du fichier remplacé
PS C:\Users\alexa> Set-Content -Path D:\Item\ChildItem-1.txt -Value "Goodbye world!"
PS C:\Users\alexa> Get-Content D:\Item\ChildItem-1.txt
Goodbye world!
```

ACL

Les ACL ou "Access Control List" sont les permissions d'accès assignée à différents fichiers. Ces listes peuvent être consultées ou modifiées en Powershell à l'aide des commandes "Get-ACL" et "Set-ACL". Contrairement à "chmod" sous Linux nous permettant seulement de modifier les autorisations du fichier pour son propriétaire, son groupe propriétaire et les autres, la manipulation des ACL avec Powershell nous permet de gérer les ACLs avancées (ex. groupes et utilisateurs supplémentaires) tout comme "setfact" et "getfacl" sous Linux.

Les propriétés les plus intéressantes de ce type d'objet sont le propriétaire, le groupe propriétaire et la propriété "AccessToString" déterminant quel utilisateur a accès au fichier.

```
# Vérification initiale des ACL (Les propriétés non pertinentes ont été retirées de
l'affichage)
PS C:\Users\alexa> Get-ACL D:\Item\ChildItem-1.txt | Select-Object -Property *

Path                   : Microsoft.PowerShell.Core\FileSystem::D:\Item\ChildItem-1.txt
Owner                  : PC\alexa
Group                  : PC\alexa
AccessToString         : BUILTIN\Administrators Allow  FullControl
                         NT AUTHORITY\SYSTEM Allow  FullControl
                         NT AUTHORITY\Authenticated Users Allow  Modify, Synchronize
                         BUILTIN\Users Allow  ReadAndExecute, Synchronize

# Ajout d'une permission
#
# Récupérer les permissions actuelles
PS C:\Users\alexa> $acl = Get-ACL D:\Item\ChildItem-1.txt
# Créer une nouvelle permission et la traduire en objet de règle
PS C:\Users\alexa> $permission = "PC\Invité","Modify","None",'None','Allow'
PS C:\Users\alexa> $rule = New-Object System.Security.AccessControl.FileSystemAccessRule
$permission
# Ajouter la règle à l'objet de permissions
PS C:\Users\alexa> $acl.AddAccessRule($rule)
# Appliquer le nouvel objet de permissions au fichier
PS C:\Users\alexa> Set-ACL -Path D:\Item\ChildItem-1.txt -AclObject $acl

# Valider les nouvelles permissions, observer que "Invité" a été ajouté aux permissions (Les
propriétés non pertinentes ont été retirées de l'affichage)
PS C:\Users\alexa> Get-ACL -Path D:\Item\ChildItem-1.txt | Select-Object -Property *
```

```
Path                   : Microsoft.PowerShell.Core\FileSystem::D:\Item\ChildItem-1.txt
Owner                  : PC\alexa
Group                  : PC\alexa
AccessToString         : PC\Invité Allow  Modify, Synchronize
                         BUILTIN\Administrators Allow  FullControl
                         NT AUTHORITY\SYSTEM Allow  FullControl
                         NT AUTHORITY\Authenticated Users Allow  Modify, Synchronize
                         BUILTIN\Users Allow  ReadAndExecute, Synchronize
```

Comme on peut l'observer ici, il arrive souvent que les objets deviennent rapidement complexes à créer et intégrer.

Service

Un autre objet pertinent à savoir manipuler dans un contexte d'automatisation est un service. À l'aide des commandes Start, Stop, Restart et Set, vous pourrez manipuler les différents services en exécution sur votre machine. Il est important pour ce type d'objet de faire la différence entre le nom du service et son nom d'affichage pour bien manipuler ces objets.

```
PS C:\Users\alexa> Get-Service | Where-Object Name -like wuauserv | Select-Object -Property *


Name                : wuauserv
RequiredServices    : {rpcss}
CanPauseAndContinue : False
CanShutdown         : False
CanStop             : False
DisplayName         : Windows Update
DependentServices   : {}
MachineName         : .
ServiceName         : wuauserv
ServicesDependedOn  : {rpcss}
ServiceHandle       :
Status              : Stopped
ServiceType         : Win32OwnProcess, Win32ShareProcess
StartType           : Manual
Site                :
Container           :
```

On peut observer ici le service de Windows Update et son type de démarrage qui est manuel. Ce service démarre seulement lorsqu'une recherche ou installation de mises à jour soit déclenchée, que ce soit manuellement ou par tâche planifiée.

Manipulation d'objets et commandes communes

Get-Help

La commande Get-Help peut vous permettre au même titre que "man" en Linux de voir tous les paramètres qu'une commande Powershell peut accepter. À sa première exécution, l'aide sera téléchargée pour une panoplie de modules et sera ensuite disponible même hors ligne.

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


NOM
    Get-Help

RÉSUMÉ
    Displays information about PowerShell commands and concepts.

SYNTAXE
    Get-Help [[-Name] <System.String>] [-Category {Alias | Cmdlet | Provider | General | FAQ |
Glossary | HelpFile |
    ScriptCommand | Function | Filter | ExternalScript | All | DefaultHelp | Workflow |
DscResource | Class |
    Configuration}] [-Component <System.String[]>] -Detailed [-Functionality
<System.String[]>] [-Path <System.String>]
    [-Role <System.String[]>] [<CommonParameters>]

    Get-Help [[-Name] <System.String>] [-Category {Alias | Cmdlet | Provider | General | FAQ |
Glossary | HelpFile |
    ScriptCommand | Function | Filter | ExternalScript | All | DefaultHelp | Workflow |
DscResource | Class |
    Configuration}] [-Component <System.String[]>] -Examples [-Functionality
<System.String[]>] [-Path <System.String>]
    [-Role <System.String[]>] [<CommonParameters>]

    Get-Help [[-Name] <System.String>] [-Category {Alias | Cmdlet | Provider | General | FAQ |
```

```
Glossary | HelpFile |
    ScriptCommand | Function | Filter | ExternalScript | All | DefaultHelp | Workflow |
DscResource | Class |
    Configuration}] [-Component <System.String[]>] [-Full] [-Functionality <System.String[]>]
[-Path <System.String>]
    [-Role <System.String[]>] [<CommonParameters>]

    Get-Help [[-Name] <System.String>] [-Category {Alias | Cmdlet | Provider | General | FAQ |
Glossary | HelpFile |
    ScriptCommand | Function | Filter | ExternalScript | All | DefaultHelp | Workflow |
DscResource | Class |
    Configuration}] [-Component <System.String[]>] [-Functionality <System.String[]>] -Online
[-Path <System.String>]
    [-Role <System.String[]>] [<CommonParameters>]

    Get-Help [[-Name] <System.String>] [-Category {Alias | Cmdlet | Provider | General | FAQ |
Glossary | HelpFile |
    ScriptCommand | Function | Filter | ExternalScript | All | DefaultHelp | Workflow |
DscResource | Class |
    Configuration}] [-Component <System.String[]>] [-Functionality <System.String[]>] -
Parameter <System.String> [-Path
    <System.String>] [-Role <System.String[]>] [<CommonParameters>]

    Get-Help [[-Name] <System.String>] [-Category {Alias | Cmdlet | Provider | General | FAQ |
Glossary | HelpFile |
    ScriptCommand | Function | Filter | ExternalScript | All | DefaultHelp | Workflow |
DscResource | Class |
    Configuration}] [-Component <System.String[]>] [-Functionality <System.String[]>] [-Path
<System.String>] [-Role
    <System.String[]>] -ShowWindow [<CommonParameters>]
```

La commande peut aussi vous présenter des exemples d'utilisation de la commande recherchée :

```
PS C:\WINDOWS\system32> Get-Help Get-Help -examples

NOM
    Get-Help

RÉSUMÉ
    Displays information about PowerShell commands and concepts.
```

```
    --- Example 1: Display basic help information about a cmdlet ---

    Get-Help Format-Table
    Get-Help -Name Format-Table
    Format-Table -?

    `Get-Help <cmdlet-name>` is the simplest and default syntax of `Get-Help` cmdlet. You can
omit the Name parameter.

    The syntax `<cmdlet-name> -?` works only for cmdlets.
```

New-Object

La commande New-Object est généralement utilisée pour déclarer un objet et le stocker dans une variable. On peut considérer les objets en Powershell comme une instanciation de classe en programmation. Un bon exemple de ceci est présent dans la section ACL des types d'objets.

Un autre cas d'utilisation commun de cette commande est pour créer un objet de type crédentiels pour éviter de redemander à l'utilisateur ses informations à chaque commande qui les nécessite. Quoi qu'il soit possible de les récupérer avec la commande Get-Credential, si on désire rester dans l'invite de commande ou charger ce type d'objet depuis un fichier, il devient plus simple d'utiliser la commande New-Object de la façon suivante :

```
# Lecture des informations d'authentification
$uname = Read-Host -Prompt "Nom d'utilisateur"
$pwd = Read-Host -Prompt "Mot de passe" -AsSecureString

# Création de l'objet
$credential = New-Object System.Management.Automation.PSCredential($uname,$pwd)
```

> Quoi qu'il soit possible d'inscrire directement des informations d'authentification dans le script, ces scripts sont généralement lisible par les postes de travail et voir même les utilisateurs. Ceci est fortement déconseillé!

New

D'autres commandes "New" servent à créer des objets à pour système d'exploitation tel que des utilisateurs locaux, des groupes locaux, des entrées dans le journal d'évènement, etc.

Par exemple, la commande New-User ajouterait un utilisateur local à votre système :

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

Name               Enabled Description
----               ------- -----------
Administrateur     False   Compte d’utilisateur d’administration
Bob                True
DefaultAccount     False   Compte utilisateur géré par le système.
Invité             False   Compte d’utilisateur invité
WDAGUtilityAccount False   Compte d’utilisateur géré et utilisé par le système pour les
scénarios Windows Defender A...

PS C:\WINDOWS\system32> $pwd = ConvertTo-SecureString "Soleil123" -AsPlainText -Force
PS C:\WINDOWS\system32> New-LocalUser -FullName "Ta mère" -Name "rita" -Password $pwd -
PasswordNeverExpires -AccountNeverExpires

Name Enabled Description
---- ------- -----------
rita True

PS C:\WINDOWS\system32> Get-LocalUser

Name               Enabled Description
----               ------- -----------
Administrateur     False   Compte d’utilisateur d’administration
Bob                True
DefaultAccount     False   Compte utilisateur géré par le système.
Invité             False   Compte d’utilisateur invité
rita               True
WDAGUtilityAccount False   Compte d’utilisateur géré et utilisé par le système pour les
scénarios Windows Defender A...
```

Add

Les commandes débutant par Add servent généralement à ajouter des entrées à des systèmes existants. Il peut s'agir de créer des fichiers, des entrées de registre, des utilisateurs dans un domaine Active Directory, des fonctionnalités Windows, des périphériques réseau, etc.

Une de ces commandes pourrait servir à ajouter un utilisateur à un groupe.

```
PS C:\WINDOWS\system32> Get-LocalGroupMember -Group "Administrateurs"

ObjectClass Name                            PrincipalSource
----------- ----                            ---------------
Utilisateur DESKTOP-D67ERFV\Administrateur Local
Utilisateur DESKTOP-D67ERFV\Bob            Local

PS C:\WINDOWS\system32> Add-LocalGroupMember -Group "Administrateurs" -Member rita
PS C:\WINDOWS\system32> Get-LocalGroupMember -Group "Administrateurs"

ObjectClass Name                            PrincipalSource
----------- ----                            ---------------
Utilisateur DESKTOP-D67ERFV\Administrateur Local
Utilisateur DESKTOP-D67ERFV\Bob            Local
Utilisateur DESKTOP-D67ERFV\rita           Local
```

Set

Les commandes "Set" sont utilisées pour modifier ou définir un ou des paramètres liés à un objet. Ces commandes remplacent la valeur mentionnée.

Une de ces commandes pourrait servir à modifier le mot de passe d'un utilisateur.

```
PS C:\WINDOWS\system32> $pwd = ConvertTo-SecureString "123Soleil" -AsPlainText -Force
PS C:\WINDOWS\system32> Set-LocalUser -Name rita -Password $pwd
```

Get

Les commandes "Get" servent à consulter les objets système ou les objets de scripts comme les variables. Elles sont souvent utilisées conjointement aux commandes "Select-Object", "Where-Object" et à des commandes d'action telles que les commandes "Set" par du "pipelining".

En voici quelques exemples :

```
# Afficher seulement certaines propriétés déterminées par Select-Object de certains objets
correspondant à un critère déterminé par Where-Object
PS C:\Users\alexa> Get-ChildItem -Path D:\Item | Where-Object Name -like "*.txt" | Select-
Object -Property FullName,LastWriteTime

FullName               LastWriteTime
```

```
--------                ------------
D:\Item\ChildItem-1.txt 2025-01-18 4:58:47 PM
D:\Item\ChildItem-2.txt 2025-01-18 4:38:23 PM

# Renommer les fichiers correspondants à ceux correspondant à la requête Where-Object à l'aide de la commande Rename-Item
PS C:\Users\alexa> Get-ChildItem -Path D:\Item | Where-Object Name -like "*.txt" | Rename-Item -NewName {$_.name -replace 'ChildItem','ObjetEnfant'}
PS C:\Users\alexa> Get-ChildItem -Path D:\Item | Where-Object Name -like "*.txt" | Select-Object -Property FullName,LastWriteTime

FullName                  LastWriteTime
--------                  ------------
D:\Item\ObjetEnfant-1.txt 2025-01-18 4:58:47 PM
D:\Item\ObjetEnfant-2.txt 2025-01-18 4:38:23 PM

# Supprimer les fichiers correspondant au filtre Where-Object
PS C:\Users\alexa> Get-ChildItem -Path D:\Item | Where-Object Name -like "*.txt" | Remove-Item
PS C:\Users\alexa> Get-ChildItem -Path D:\Item


    Directory: D:\Item

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        2025-01-18   4:40 PM                Subdirectory
```

Where-Object

La commande Where-Object généralement utilisée avec une commande de type "Get" permet de filtrer les résultats affichés en fonction d'un ou plusieurs paramètres de l'objet ainsi que les objets résultants sur lesquels la commande "pipelinée" suivante sera appliquée. Dans l'exemple précédent, le filtre récupérait uniquement les fichiers texte donc l'action de retrait n'a pas affecté la structure de dossiers.

Dans l'exemple suivant, seulement

Select-Object

La commande Select-Object permet de déterminer quels paramètres récupérer ou afficher d'un objet. Dans l'exemple précédent, Seulement le nom complet du fichier ainsi que la dernière date

de modification étaient sélectionnés et affichés.

Read-Host/Write-Host

Ces commandes sont généralement utilisées pour questionner l'utilisateur et stocker la réponse dans une variable et pour écrire de la journalisation dans la console.

Base

Manipulation de chemins d'accès pour objets Powershell

Par défaut, une session Powershell a accès aux disques locaux (ex. C:\), aux ruches de registres HKLM (*HKEY_LOCAL_MACHINE*), HKCU (*HKEY_CURRENT_USER*), aux partages réseaux accessibles dans le contexte d'exécution de la session ou du script, aux variables d'environnement (*$env:*) et quelques autres ressources.

Get-PSDrive

La commande Get-PSDrive permet de lister l'ensemble des ressources accessibles par la session/script powershell :

```
PS C:\Users\alexa> Get-PSDrive

Name           Used (GB)     Free (GB) Provider
Root
----           ---------     --------- --------      ----

A                 486.59        530.63 FileSystem    \\10.4.20.69\arsenaultja
Alias                                  Alias
C                 877.85         52.87 FileSystem
C:\
Cert                                   Certificate   \
D                 183.83         39.14 FileSystem    D:\
E                1510.09        352.93 FileSystem    E:\
Env                                    Environment
Function                               Function
HKCU                                   Registry      HKEY_CURRENT_USER
HKLM                                   Registry      HKEY_LOCAL_MACHINE
M                3705.20        757.75 FileSystem    \\10.4.20.69\Media
Variable                               Variable
WSMan                                  WSMan
```

New-PSDrive

La commande New-PSDrive permet de gagner accès à des ressources supplémentaires, qu'il s'agisse d'un lecteur réseau, d'une ruche de registre ou d'une clé USB, la commande New-PSDrive nous permet d'ajouter un chemin pour accéder à cette ressources à partir d'un script ou d'une session Powershell.

Connexion d'une ruche de registre

Dans l'exemple suivant, la commande New-PSDrive nous permettra d'accéder à la ruche de registre HKEY_USERS

```
PS C:\Users\alexa> Get-PSDrive | Where-Object Provider -like *Registry*

Name           Used (GB)     Free (GB) Provider
Root                                            CurrentLocation
----           ---------     --------- --------      ----
                                                ---------------
HKCU                                    Registry      HKEY_CURRENT_USER
HKLM                                    Registry      HKEY_LOCAL_MACHINE

PS C:\Users\alexa> New-PSDrive -Name HKU -PSProvider Registry -Root HKEY_USERS

Name           Used (GB)     Free (GB) Provider
Root                                            CurrentLocation
----           ---------     --------- --------      ----
                                                ---------------
HKU                                     Registry      HKEY_USERS

PS C:\Users\alexa> Get-PSDrive | Where-Object Provider -like *Registry*

Name           Used (GB)     Free (GB) Provider
Root                                            CurrentLocation
----           ---------     --------- --------      ----
                                                ---------------
HKCU                                    Registry      HKEY_CURRENT_USER
HKLM                                    Registry      HKEY_LOCAL_MACHINE
HKU                                     Registry      HKEY_USERS
```

Ajout d'un PSDrive référant à un stockage

Tout comme un raccourci, il est possible de créer un PSDrive faisant référence à un stockage existant. Dans l'exemple suivant, nous créons un PSDrive faisant référence au dossier de l'utilisateur nommé "USRDIR". Celui-ci nous permettra d'interagir avec le contenu de C:\Users\[mon utilisateur] en poitant USRDIR:\ plutôt que le chemin complet.

```
PS C:\Users\alexa> Get-PSDrive | Where-Object Provider -like *FileSystem*

Name           Used (GB)     Free (GB) Provider      Root
----           ---------     --------- --------      ----
A                 486.59        530.63 FileSystem    \\10.4.20.69\arsenaultja
C                 877.96         52.76 FileSystem    C:\
D                 183.83         39.14 FileSystem    D:\
E                1510.09        352.93 FileSystem    E:\

PS C:\Users\alexa> New-PSDrive -Name USRDIR -PSProvider FileSystem -Root
C:\Users\$env:UserName

Name           Used (GB)     Free (GB) Provider
Root                                            CurrentLocation
----           ---------     --------- --------      ----
                                               ---------------
USRDIR              0.00         52.98 FileSystem    C:\Users\alexa

PS C:\Users\alexa> Get-ChildItem -Path USRDIR:


    Directory: C:\Users\alexa


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        2023-12-28  10:24 PM                .android
d-----        2023-05-02   7:59 PM                .arduinoIDE
d-----        2023-12-13  11:13 AM                .cache
d-----        2025-01-25   4:26 PM                .config
d-----        2024-11-13   1:33 PM                .dbus-keyrings
d-----        2025-01-13   3:37 PM                .espressif
d-----        2024-05-17  10:25 AM                .ms-ad
d-----        2025-01-26   8:06 PM                .ssh
d-----        2025-02-05   3:56 PM                .VirtualBox
d-----        2023-06-11  12:15 PM                .vscode
d-----        2024-12-30  10:25 PM                .zenmap
d-r---        2023-11-18  10:05 PM                Contacts
d-----        2023-03-12   8:12 PM                Documents
d-r---        2025-02-08  12:14 AM                Downloads
```

```
d-r---        2023-11-18  10:05 PM                Favorites
d-----        2023-08-06   1:44 PM                GNS3
d-r---        2023-11-18  10:05 PM                Links
d-r---        2024-07-15   4:02 PM                Music
d-----        2024-09-21   2:29 PM                node_modules
dar---        2023-11-18  10:00 PM                OneDrive
d-r---        2024-09-25  10:01 PM                Saved Games
d-----        2023-03-12   8:27 PM                scoop
d-r---        2025-01-22   9:16 PM                Searches
d-----        2024-05-13   7:11 AM                Sync
d-r---        2025-01-01   5:57 PM                Videos
d-----        2025-01-26   5:12 PM                VirtualBox VMs
-a----        2024-09-22   1:24 AM    15093960192 gns3.ova
-a----        2024-08-15  10:24 AM      131138940 Templates.gns3project
```

Retrait d'un PSDrive

Il est aussi possible par la même logique de retirer un PSDrive.

```
PS C:\Users\alexa> Get-PSDrive | Where-Object Provider -like *FileSystem*

Name           Used (GB)     Free (GB) Provider      Root
----           ---------     --------- --------      ----
A                 486.59        530.63 FileSystem    \\10.4.20.69\arsenaultja
C                 877.67         53.05 FileSystem    C:\
D                 183.83         39.14 FileSystem    D:\
E                1510.09        352.93 FileSystem    E:\
F                  43.70         15.18 FileSystem    F:\
G                   0.02          0.01 FileSystem    G:\
H                3471.56        254.42 FileSystem    H:\
J                1471.71        391.29 FileSystem    J:\
M                3705.20        757.75 FileSystem    \\10.4.20.69\Media
USRDIR              0.00         53.05 FileSystem    C:\Users\alexa


PS C:\Users\alexa> Get-PSDrive | Where-Object Name -like USRDIR | Remove-PSDrive

PS C:\Users\alexa> Get-PSDrive | Where-Object Provider -like *FileSystem*

Name           Used (GB)     Free (GB) Provider      Root
----           ---------     --------- --------      ----
```

```
A                486.59        530.63 FileSystem    \\10.4.20.69\arsenaultja
C                877.67         53.05 FileSystem    C:\
D                183.83         39.14 FileSystem    D:\
E               1510.09        352.93 FileSystem    E:\
F                 43.70         15.18 FileSystem    F:\
G                  0.02          0.01 FileSystem    G:\
H               3471.56        254.42 FileSystem    H:\
J               1471.71        391.29 FileSystem    J:\
M               3705.20        757.75 FileSystem    \\10.4.20.69\Media
```

Administration de Windows

Par défaut, Powershell vient avec la majorité des commandes nécessaires à l'administration de Windows. À quelques exceptions près (entre autres la gestion de Windows Updates)

Utilisateurs et groupes

Récupération d'information sur les utilisateurs et groupes locaux

Get-LocalUser

Une commande simple comme la suivante nous permettrait de vérifier quel utilisateur a un profil local sur une machine ainsi que sa dernière date de connexion. Ce genre d'information est souvent utilisé pour nettoyer les profils locaux d'utilisateurs Active Directory inactifs.

```
PS C:\WINDOWS\system32> Get-LocalUser | Select-Object -Property Name,Enabled,LastLogon


Name               Enabled LastLogon
----               ------- ---------
Administrateur       False
DefaultAccount       False
Info                  True 2025-01-19 19:36:18
Invité               False
WDAGUtilityAccount   False
```

Les informations récupérables par cette commande sont toutefois limitées comme le démontre la commande suivante.

```
PS C:\WINDOWS\system32> Get-LocalUser | Where-Object Name -like Info | Select-Object -Property
*


AccountExpires         :
Description            :
Enabled                : True
FullName               :
PasswordChangeableDate :
PasswordExpires        :
UserMayChangePassword  : True
PasswordRequired       : False
PasswordLastSet        :
```

```
LastLogon              : 2025-01-19 19:36:18
Name                   : Info
SID                    : S-1-5-21-2716239665-1816042345-2264835360-1001
PrincipalSource        : Local
ObjectClass            : Utilisateur
```

Les dates d'expiration et de changement de mot de passe sont habituellement utilisées pour soit alerter les utilisateurs de l'expiration imminente de leur mot de passe ou encore pour effectuer des recommendations en terme de pratique de cybersécurité.

Get-Process

Quoi que Get-Process soit en lien avec l'administration du système plutôt que de ses utilisateurs, elle peut être utilisée pour déterminer quel utilisateur a une session active sur un poste de travail en regardant quelles instances d'un processus commun à tous tel que explorer.exe sont en exécution!

```
PS C:\WINDOWS\system32> Get-Process -IncludeUserName | Where-Object Name -like explorer* |
Select-Object -Property UserName,StartTime

UserName                       StartTime
--------                       ---------
Vidange-Windose\Administrateur 2025-01-19 19:59:14
Vidange-Windose\Info           2025-01-19 18:08:00
```

Get-LocalGroup

La commande Get-LocalGroup nous permet d'interroger le système au sujet des groupes locaux mais tout comme Get-LocalUsers, la commande n'est pas très bavarde.

```
PS C:\WINDOWS\system32> Get-LocalGroup -Name "Administrateurs" | Select-Object -Property *

Description     : Les membres du groupe Administrateurs disposent d’un accès complet et
illimité à l’ordinateur et au
                  domaine
Name            : Administrateurs
SID             : S-1-5-32-544
PrincipalSource : Local
ObjectClass     : Groupe
```

Get-LocalGroupMember

C'est plutôt la commande Get-LocalGroupMember qui nous permet de voir l'association entre les utilisateurs et les groupes.

```
PS C:\WINDOWS\system32> Get-LocalGroupMember -Name "Administrateurs" | Select-Object -Property
*

Name                          SID                                               PrincipalSource
ObjectClass
----                          ---                                               ---------------
-----------
Vidange-Windose\Administrateur S-1-5-21-2716239665-1816042345-2264835360-500           Local
Utilisateur
Vidange-Windose\Info          S-1-5-21-2716239665-1816042345-2264835360-1001          Local
Utilisateur
```

Pour vérifier la liste des groupes auxquels un utilisateur appartient, il faut boucler dans tous les utilisateurs et tous les groupes avec un script tel que celui-ci.

```
PS C:\WINDOWS\system32> Get-LocalUser | ForEach-Object {
>>   $user = $_
>>   return [PSCustomObject]@{
>>     "User" = $user.Name
>>     "Groups" = Get-LocalGroup | Where-Object {
>>       $user.SID -in ($_ | Get-LocalGroupMember | Select-Object -ExpandProperty "SID") |
Select-Object -Property "Name"
>>     }
>>   }
>> }

User               Groups
----               ------
Administrateur     {Administrateurs, Administrateurs Hyper-V, Duplicateurs, IIS_IUSRS...}
DefaultAccount     {Administrateurs, Administrateurs Hyper-V, Duplicateurs, IIS_IUSRS...}
Info               {Administrateurs, Administrateurs Hyper-V, Duplicateurs, IIS_IUSRS...}
Invité             {Administrateurs, Administrateurs Hyper-V, Duplicateurs, IIS_IUSRS...}
WDAGUtilityAccount {Administrateurs, Administrateurs Hyper-V, Duplicateurs, IIS_IUSRS...}
```

Si un utilisateur était mentionné, la première boucle ne serait pas nécessaire.

Gestion des utilisateurs et des groupes locaux

Puisque les objets sont simples, leur création et leur gestion l'est aussi.

New-LocalUser

En se fiant à la commande Get-LocalUser mentionnée plus haut, on peut déterminer les paramètres à saisir pour créer un nouvel utilisateur à l'aide de la commande New-LocalUser.

Si un mot de passe doit être précisé, il doit être mentionné sous forme d'une "SecureString". Pour faciliter la structure de ces commandes, on peut stocker le mot de passe dans une variable plutôt que de faire un "one-liner" mais il est aussi possible de tout effectuer dans la même commande.

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

Name               Enabled Description
----               ------- -----------
Administrateur     True    Compte d’utilisateur d’administration
DefaultAccount     False   Compte utilisateur géré par le système.
Info               True
Invité             False   Compte d’utilisateur invité
WDAGUtilityAccount False   Compte d’utilisateur géré et utilisé par le système pour les
scénarios Windows Defender Applic...

PS C:\WINDOWS\system32> $pwd = ConvertTo-SecureString -String "bobettes" -AsPlainText -Force
PS C:\WINDOWS\system32> New-LocalUser -Name "bob" -FullName "Bobby" -Password $pwd

Name Enabled Description
---- ------- -----------
bob  True

PS C:\WINDOWS\system32> Get-LocalUser

Name               Enabled Description
----               ------- -----------
Administrateur     True    Compte d’utilisateur d’administration
bob                True
DefaultAccount     False   Compte utilisateur géré par le système.
Info               True
Invité             False   Compte d’utilisateur invité
WDAGUtilityAccount False   Compte d’utilisateur géré et utilisé par le système pour les
scénarios Windows Defender Applic...
```

Set-LocalUser

La commande "Set-LocalUser" fonctionne de la même façon que la commande "New-LocalUser" à l'exception que le paramètre "-Name" qui sera mentionné représentera le compte à modifier.

```
PS C:\WINDOWS\system32> Get-LocalUser bob

Name Enabled Description
---- ------- -----------
bob  True



PS C:\WINDOWS\system32> Set-LocalUser -Name bob -Description "Ta mère"
PS C:\WINDOWS\system32> Get-LocalUser bob

Name Enabled Description
---- ------- -----------
bob  True    Ta mère
```

New-LocalGroup

Encore une fois, en se fiant à Get-LocalGroup, on peut déterminer l'ensemble des paramètres qui seront à mentionner à la création d'un groupe.

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

Name                                Description
----                                -----------
Administrateurs                     Les membres du groupe Administrateurs disposent
d’un accès complet et illimit...
Utilisateurs                        Les utilisateurs ne peuvent pas effectuer de
modifications accidentelles ou i...
Utilisateurs de gestion à distance  Les membres de ce groupe ont accès aux ressources
WMI via des protocoles de g...
Utilisateurs du Bureau à distance   Les membres de ce groupe disposent des droits
nécessaires pour ouvrir une ses...
Utilisateurs OpenSSH                Les membres de ce groupe peuvent se connecter à
cet ordinateur à l’aide de SSH.



PS C:\WINDOWS\system32> New-LocalGroup -Name "Yo Les Jeunes" -Description "Le club secret à
Bob"
```

```
Name          Description
----          -----------
Yo Les Jeunes Le club secret à Bob



PS C:\WINDOWS\system32> Get-LocalGroup

Name                                  Description
----                                  -----------
Yo Les Jeunes                         Le club secret à Bob
Administrateurs                       Les membres du groupe Administrateurs disposent
d’un accès complet et illimit...
Utilisateurs                          Les utilisateurs ne peuvent pas effectuer de
modifications accidentelles ou i...
Utilisateurs de gestion à distance    Les membres de ce groupe ont accès aux ressources
WMI via des protocoles de g...
Utilisateurs du Bureau à distance     Les membres de ce groupe disposent des droits
nécessaires pour ouvrir une ses...
Utilisateurs OpenSSH                  Les membres de ce groupe peuvent se connecter à
cet ordinateur à l’aide de SSH.
```

Set-LocalGroup

Tout comme pour Set-LocalUser, Set-LocalGroup utilise le paramètre "Name" pour déterminer le groupe à modifier.

```
PS C:\WINDOWS\system32> Get-LocalGroup -Name "Yo Les Jeunes"

Name          Description
----          -----------
Yo Les Jeunes Le club secret à Bob



PS C:\WINDOWS\system32> Set-LocalGroup -Name "Yo Les Jeunes" -Description "Le club plus tant
secret à Bob"
PS C:\WINDOWS\system32> Get-LocalGroup -Name "Yo Les Jeunes"

Name          Description
----          -----------
Yo Les Jeunes Le club plus tant secret à Bob
```

Add-LocalGroupMember

Pour attribuer des utilisateurs ou des groupes à un groupe (oui, on peut imbriquer des groupes), la commande Add-LocalGroupMember est aussi simple que de mentionner le nom du membre qui peut être le nom d'un utilisateur ou d'un groupe et de mentionner le groupe auquel assigner le membre.

On assigne ici l'utilisateur "bob" à son groupe

```
PS C:\WINDOWS\system32> Get-LocalGroupMember "Yo Les Jeunes"
PS C:\WINDOWS\system32> Add-LocalGroupMember -Group "Yo Les Jeunes" -Member "bob"
PS C:\WINDOWS\system32> Get-LocalGroupMember "Yo Les Jeunes"

ObjectClass Name                PrincipalSource
----------- ----                ---------------
Utilisateur Vidange-Windose\bob Local
```

Le principe est le même pour imbriquer un groupe dans un autre

```
PS C:\WINDOWS\system32> Add-LocalGroupMember -Group "Yo Les Jeunes" -Member "Invités"
PS C:\WINDOWS\system32> Get-LocalGroupMember "Yo Les Jeunes"

ObjectClass Name                PrincipalSource
----------- ----                ---------------
Groupe      BUILTIN\Invités     Local
Utilisateur Vidange-Windose\bob Local
```

Remove-LocalGroupMember

Le principe ici est le même que pour l'ajout!

```
PS C:\WINDOWS\system32> Get-LocalGroupMember "Yo Les Jeunes"

ObjectClass Name                PrincipalSource
----------- ----                ---------------
Groupe      BUILTIN\Invités     Local
Utilisateur Vidange-Windose\bob Local



PS C:\WINDOWS\system32> Remove-LocalGroupMember -Group "Yo Les Jeunes" -Member "bob"
PS C:\WINDOWS\system32> Get-LocalGroupMember "Yo Les Jeunes"
```

```
ObjectClass Name             PrincipalSource
----------- ----             ---------------
Groupe      BUILTIN\Invités Local
```

Applications et fonctionnalités

Récupération d'information sur les applications et fonctionnalités installées et installables

Get-AppxPackage

La commande Get-AppxPackage vous permet de récupérer les informations sur les applications UWP (Universal Windows Platform) installées sur le système.

```
PS C:\WINDOWS\system32> Get-AppxPackage | Where-Object Name -like *Copilot | Select-Object -
Property *



Name                  : Microsoft.Copilot
Publisher             : CN=Microsoft Corporation, O=Microsoft Corporation, L=Redmond,
S=Washington, C=US
PublisherId           : 8wekyb3d8bbwe
Architecture          : X64
ResourceId            :
Version               : 1.24122.48.0
PackageFamilyName     : Microsoft.Copilot_8wekyb3d8bbwe
PackageFullName       : Microsoft.Copilot_1.24122.48.0_x64__8wekyb3d8bbwe
InstallLocation       : C:\Program
Files\WindowsApps\Microsoft.Copilot_1.24122.48.0_x64__8wekyb3d8bbwe
IsFramework           : False
PackageUserInformation : {}
IsResourcePackage     : False
IsBundle              : False
IsDevelopmentMode     : False
NonRemovable          : False
Dependencies          : {Microsoft.WindowsAppRuntime.1.6_6000.373.1641.0_x64__8wekyb3d8bbwe,
                        Microsoft.VCLibs.140.00.UWPDesktop_14.0.33728.0_x64__8wekyb3d8bbwe,
                        Microsoft.VCLibs.140.00_14.0.33519.0_x64__8wekyb3d8bbwe,
                        Microsoft.Copilot_1.24122.48.0_neutral_split.language-
fr_8wekyb3d8bbwe}
```

```
IsPartiallyStaged      : False
SignatureKind          : Store
Status                 : Ok
```

Get-AppxProvisionedPackage

La commande Get-AppxProvisionedPackage vous permet de récupérer les informations sur les applications UWP (Universal Windows Platform) qui seront déployées pour tous les nouveaux utilisateurs. Cette commande peut aussi être utilisée sur un système hors ligne pour visualiser une image d'installation ou un système qui n'est pas démarré.

```
PS C:\WINDOWS\system32> Get-AppxProvisionedPackage -Online | Where-Object DisplayName -like
MSTeams | Select-Object -Property *



Version          : 24295.605.3225.8804
PackageName      : MSTeams_24295.605.3225.8804_x64__8wekyb3d8bbwe
DisplayName      : MSTeams
PublisherId      : 8wekyb3d8bbwe
MajorVersion     : 24295
MinorVersion     : 605
Build            : 3225
Revision         : 8804
Architecture     : 9
ResourceId       :
InstallLocation  : C:\Program
Files\WindowsApps\MSTeams_24295.605.3225.8804_x64__8wekyb3d8bbwe\AppxManifest.xml
Regions          :
AD;AE;AF;AG;AI;AL;AM;AO;AQ;AR;AS;AT;AU;AW;AX;AZ;BA;BB;BD;BE;BF;BG;BH;BI;BJ;BL;BM;BN;BQ;BR;B
S;BT;B

V;BW;BY;BZ;CA;CC;CD;CF;CG;CH;CI;CK;CL;CM;CO;CR;CU;CV;CW;CX;CY;CZ;DE;DJ;DK;DM;DO;DZ;EC;EE;EG;ER
;ES;ET

;FI;FJ;FK;FM;FO;FR;GA;GB;GD;GE;GF;GG;GH;GI;GL;GM;GN;GP;GQ;GR;GS;GT;GU;GW;GY;HK;HM;HN;HR;HT;HU;
ID;IE;

IL;IM;IN;IO;IQ;IR;IS;IT;JE;JM;JO;JP;KE;KG;KH;KI;KM;KN;KP;KR;KW;KY;KZ;LA;LB;LC;LI;LK;LR;LS;LT;L
U;LV;L

Y;MA;MC;MD;ME;MF;MG;MH;MK;ML;MM;MN;MO;MP;MQ;MR;MS;MT;MU;MV;MW;MX;MY;MZ;NA;NC;NE;NF;NG;NI;NL;NO
```

```
;NP;NR

;NU;NZ;OM;PA;PE;PF;PG;PH;PK;PL;PM;PN;PR;PS;PT;PW;PY;QA;RE;RO;RS;RU;RW;SA;SB;SC;SD;SE;SG;SH;SI;
SJ;SK;

SL;SM;SN;SO;SR;SS;ST;SV;SX;SY;SZ;TC;TD;TF;TG;TH;TJ;TK;TL;TM;TN;TO;TR;TT;TV;TW;TZ;UA;UG;UM;US;U
Y;UZ;V
                  A;VC;VE;VG;VI;VN;VU;WF;WS;XK;YE;YT;ZA;ZM;ZW
Path             :
Online           : True
WinPath          :
SysDrivePath     :
RestartNeeded    : False
LogPath          : C:\WINDOWS\Logs\DISM\dism.log
ScratchDirectory :
LogLevel         : WarningsInfo
```

```
PS F:\> Get-AppxProvisionedPackage -Path E:\mnt | Where-Object DisplayName -like *Outlook* |
Select-Object -Property *



Version          : 1.0.0.0
PackageName      : Microsoft.OutlookForWindows_1.0.0.0_neutral__8wekyb3d8bbwe
DisplayName      : Microsoft.OutlookForWindows
PublisherId      : 8wekyb3d8bbwe
MajorVersion     : 1
MinorVersion     : 0
Build            : 0
Revision         : 0
Architecture     : 11
ResourceId       :
InstallLocation  : %SYSTEMDRIVE%\Program

Files\WindowsApps\Microsoft.OutlookForWindows_1.0.0.0_neutral__8wekyb3d8bbwe\AppxManifest.xml
Regions          :
AD;AE;AF;AG;AI;AL;AM;AO;AQ;AR;AS;AT;AU;AW;AX;AZ;BA;BB;BD;BE;BF;BG;BH;BI;BJ;BL;BM;BN;BO;BQ;BR;B
S;BT;B

V;BW;BY;BZ;CA;CC;CD;CF;CG;CH;CI;CK;CL;CM;CO;CR;CU;CV;CW;CX;CY;CZ;DE;DJ;DK;DM;DO;DZ;EC;EE;EG;ER
;ES;ET
```

```
;FI;FJ;FK;FM;FO;FR;GA;GB;GD;GE;GF;GG;GH;GI;GL;GM;GN;GP;GQ;GR;GS;GT;GU;GW;GY;HK;HM;HN;HR;HT;HU;
ID;IE;

IL;IM;IN;IO;IQ;IR;IS;IT;JE;JM;JO;JP;KE;KG;KH;KI;KM;KN;KP;KR;KW;KY;KZ;LA;LB;LC;LI;LK;LR;LS;LT;L
U;LV;L

Y;MA;MC;MD;ME;MF;MG;MH;MK;ML;MM;MN;MO;MP;MQ;MR;MS;MT;MU;MV;MW;MX;MY;MZ;NA;NC;NE;NF;NG;NI;NL;NO
;NP;NR

;NU;NZ;OM;PA;PE;PF;PG;PH;PK;PL;PM;PN;PR;PS;PT;PW;PY;QA;RE;RO;RS;RU;RW;SA;SB;SC;SD;SE;SG;SH;SI;
SJ;SK;

SL;SM;SN;SO;SR;SS;ST;SV;SX;SY;SZ;TC;TD;TF;TG;TH;TJ;TK;TL;TM;TN;TO;TR;TT;TV;TW;TZ;UA;UG;UM;US;U
Y;UZ;V
                  A;VC;VE;VG;VI;VN;VU;WF;WS;XK;YE;YT;ZA;ZM;ZW
Path             : E:\mnt
Online           : False
WinPath          :
SysDrivePath     :
RestartNeeded    : False
LogPath          : C:\WINDOWS\Logs\DISM\dism.log
ScratchDirectory :
LogLevel         : WarningsInfo
```

Get-WindowsCapability

La commande Get-WindowsCapability nous permet de récupérer les informations sur les fonctionnalités avancées/optionnelles de Windows telles que les outils d'administration de serveurs à distance, les services s'il s'agit d'un serveur windows où les packs de languages installés et disponibles. Cette commande peut aussi être utilisée sur un système hors ligne pour visualiser une image d'installation ou un système qui n'est pas démarré.

```
PS C:\WINDOWS\system32> Get-WindowsCapability -Online | Where-Object Name -like OpenSSH* |
Select-Object -Property *



Name             : OpenSSH.Client~~~~0.0.1.0
State            : Installed
Path             :
Online           : True
```

```
WinPath          :
SysDrivePath     :
RestartNeeded    : False
LogPath          : C:\WINDOWS\Logs\DISM\dism.log
ScratchDirectory :
LogLevel         : WarningsInfo

Name             : OpenSSH.Server~~~~0.0.1.0
State            : NotPresent
Path             :
Online           : True
WinPath          :
SysDrivePath     :
RestartNeeded    : False
LogPath          : C:\WINDOWS\Logs\DISM\dism.log
ScratchDirectory :
LogLevel         : WarningsInfo
```

```
PS F:\> Get-WindowsCapability -Path E:\mnt | Where-Object State -like Installed | Select-
Object -Property Name

Name
----
App.StepsRecorder~~~~0.0.1.0
Browser.InternetExplorer~~~~0.0.11.0
DirectX.Configuration.Database~~~~0.0.1.0
Edge.Webview2.Platform~~~~
Hello.Face.20134~~~~0.0.1.0
Language.Basic~~~en-US~0.0.1.0
Language.Basic~~~fr-CA~0.0.1.0
Language.Basic~~~fr-FR~0.0.1.0
Language.Handwriting~~~fr-FR~0.0.1.0
Language.OCR~~~en-US~0.0.1.0
Language.OCR~~~fr-CA~0.0.1.0
Language.Speech~~~fr-CA~0.0.1.0
Language.TextToSpeech~~~fr-CA~0.0.1.0
MathRecognizer~~~~0.0.1.0
Media.WindowsMediaPlayer~~~~0.0.12.0
[...]
```

```
OneCoreUAP.OneSync~~~~0.0.1.0
OpenSSH.Client~~~~0.0.1.0
VBSCRIPT~~~~
Windows.Client.ProjFS~~~~
Windows.DirectoryServices.ADAM.Client.Content~~~~
Windows.HyperV.OptionalFeature.VirtualMachinePlatform.Client.Disabled~~~~
Windows.Kernel.LA57~~~~0.0.1.0
Windows.SimpleTCP.Content~~~~
Windows.Telnet.Client~~~~
Windows.TerminalServices.AppServerClient~~~~
Windows.TFTP.Client~~~~
Windows.WinOcr~~~~
Windows.WorkFolders.Client~~~~
WMIC~~~~
```

Retrait d'applications et fonctionnalités

Remove-AppxPackage

La commande Remove-AppxPackage peut être utilisée par elle même en mentionnant le nom du package ou conjointement à Get-AppxPackage pour désinstaller un package installé.

```
PS C:\WINDOWS\system32> Get-AppxPackage | Where-Object Name -like *Copilot | Remove-
AppxPackage
PS C:\WINDOWS\system32> Get-AppxPackage | Where-Object Name -like *Copilot
PS C:\WINDOWS\system32>
```

Remove-AppxProvisionedPackage

La commande Remove-AppxProvisionedPackage retire le(s) package(s) de la liste de ceux qui seront installé pour les nouveaux utilisateurs.

```
PS F:\> Get-AppxProvisionedPackage -Path E:\mnt | Where-Object DisplayName -like *Outlook* |
Remove-AppxProvisionedPackage



Path          : E:\mnt
Online        : False
RestartNeeded : False
```

```
PS F:\> Get-AppxProvisionedPackage -Path E:\mnt | Where-Object DisplayName -like *Outlook*
PS F:\>
```

Ces deux commandes pourraient être utilisées en conjonction avec un tableau pour désinstaller une liste d'application dans un script de préparation de nouveau système.

```
$list = @()
$list += '*Teams'
$list += '*BingSearch'
$list += '*BingNews'
$list += '*MicrosoftOfficeHub*'
$list += '*Game*'
$list += '*Gaming*'
$list += '*Solitaire*'
$list += '*Help*'
$list += '*Hub'
$list += '*QuickAssist*'

Foreach ($i in $list) {
Get-AppxPackage | Where-Object Name -like $i | Remove-AppxPackage -AllUsers
}
```

Ajout d'applications et fonctionnalités

Quoi qu'il soit parfois difficile de trouver les packages .msix et .msixbundle, le site suivant permet de générer des liens de téléchargement pour les fichiers d'installation des applications du Microsoft Store.

https://store.rg-adguard.net/

Add-AppxProvisionedPackage

Cette commande est utilisée pour déployer une application à tous les utilisateurs d'un système ou pour injecter une application dans une image d'installation de Windows.

Les commandes suivantes ajouteraient Microsoft PC Manager et sa dépendance à une image d'installation de Windows décapsulée avec DISM.

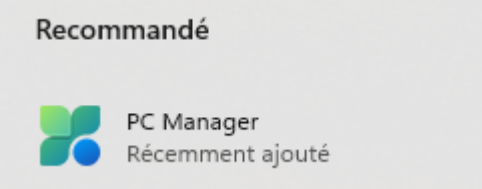
```
PS F:\> Add-AppxProvisionedPackage -PackagePath
F:\Microsoft.WindowsAppRuntime.1.5_5001.373.1736.0_x64__8wekyb3d8bbwe.Msix -Path E:\mnt -
SkipLicense
```

```
Path          : E:\mnt
Online        : False
RestartNeeded : False



PS F:\> Add-AppxProvisionedPackage -PackagePath
F:\Microsoft.MicrosoftPCManager_3.14.18.0_neutral_~_8wekyb3d8bbwe.Msixbundle -Path E:\mnt -
SkipLicense


Path          : E:\mnt
Online        : False
RestartNeeded : False

PS F:\> Get-AppxProvisionedPackage -Path E:\mnt | Where-Object DisplayName -like *Manager*


DisplayName  : Microsoft.MicrosoftPCManager
Version      : 3.14.18.0
Architecture : neutral
ResourceId   : ~
PackageName  : Microsoft.MicrosoftPCManager_3.14.18.0_neutral_~_8wekyb3d8bbwe
Regions      :
```

Les commandes suivantes installeraient Microsoft PC Manager et sa dépendance pour tous les utilisateurs d'un système sur lequel les commandes seraient exécutées.

```
PS C:\Users\Info\Desktop> Add-AppxProvisionedPackage -PackagePath
.\Microsoft.WindowsAppRuntime.1.5_5001.373.1736.0_x64__8wekyb3d8bbwe.Msix -Online -SkipLicense


Path          :
Online        : True
RestartNeeded : False
```

```
PS C:\Users\Info\Desktop> Add-AppxProvisionedPackage -PackagePath
.\Microsoft.MicrosoftPCManager_3.14.18.0_neutral_~_8wekyb3d8bbwe.Msixbundle -Online -
SkipLicense


Path          :
Online        : True
RestartNeeded : False

PS C:\Users\Info\Desktop> Get-AppxProvisionedPackage -Online | Where-Object DisplayName -like
*Manager*


DisplayName  : Microsoft.MicrosoftPCManager
Version      : 3.14.18.0
Architecture : neutral
ResourceId   : ~
PackageName  : Microsoft.MicrosoftPCManager_3.14.18.0_neutral_~_8wekyb3d8bbwe
Regions      :
```

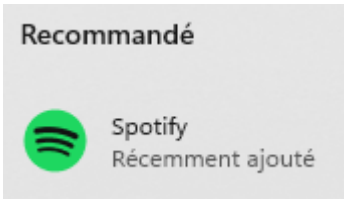


Add-AppxPackage

La commande suivante installerait Spotify et ses dépendances pour l'utilisateur exécutant la commande.

```
PS C:\Users\Info\desktop> Get-Item -Path .\Spotify\* | Add-AppxPackage
PS C:\Users\Info\desktop> Get-AppxPackage | Where-Object Name -like *Spotify* | Select-Object
-Property *


Name                  : SpotifyAB.SpotifyMusic
Publisher             : CN=453637B3-4E12-4CDF-B0D3-2A3C863BF6EF
PublisherId           : zpdnekdrzrea0
Architecture          : X64
ResourceId            :
Version               : 1.255.235.0
```

```
PackageFamilyName      : SpotifyAB.SpotifyMusic_zpdnekdrzrea0
PackageFullName        : SpotifyAB.SpotifyMusic_1.255.235.0_x64__zpdnekdrzrea0
InstallLocation        : C:\Program
Files\WindowsApps\SpotifyAB.SpotifyMusic_1.255.235.0_x64__zpdnekdrzrea0
IsFramework            : False
PackageUserInformation : {}
IsResourcePackage      : False
IsBundle               : False
IsDevelopmentMode      : False
NonRemovable           : False
Dependencies           : {Microsoft.WindowsAppRuntime.1.2_2000.802.31.0_x64__8wekyb3d8bbwe,
                         Microsoft.VCLibs.140.00_14.0.33519.0_x64__8wekyb3d8bbwe,
                         Microsoft.VCLibs.140.00.UWPDesktop_14.0.33728.0_x64__8wekyb3d8bbwe}
IsPartiallyStaged      : False
SignatureKind          : Store
Status                 : Ok
```



Registre Windows

Powershell traite les entrées de registre au même titre que les fichiers et peuvent donc être récupérés, créés et modifiés avec les mêmes commandes que les fichiers. Plutôt que de viser une lettre de lecteur et un répertoire tel que C:\, la "ruche" ou le registre sera visé avec un identifiant comme HKLM:\

Accès aux ruches du registre Windows

Par défaut, seulement HKCU (*HKEY_CURRENT_USER*) et HKLM (*HKEY_LOCAL_MACHINE*) sont accessibles. Puisque le registre est traité au même titre qu'un lecteur, il est possible d'observer ce qui est accessible dans une session ou un script powershell ainsi que de gagner accès à d'autres ruches en manipulant les PSDrives.

Get-PSDrive

Pour observer la liste des ruches de registres accessibles, il est possible d'utiliser la commande "Get-PSDrive" et de filtrer les résultats par "Provider" (type d'objet à manipuler).

```
PS C:\WINDOWS\system32> Get-PSDrive | Where-Object Provider -like *Registry*

Name           Used (GB)     Free (GB) Provider
Root                                            CurrentLocation
----           ---------     --------- --------      ----
                                                ---------------
HKCU                                  Registry      HKEY_CURRENT_USER
HKLM                                  Registry      HKEY_LOCAL_MACHINE
```

New-PSDrive

Il est possible de gagner accès aux autres ruches telles que HKCR (*HKEY_CLASSES_ROOT*) et HKU (*HKEY_USERS*) en les ajoutant comme PSDrive.

```
PS C:\WINDOWS\system32> New-PSDrive -PSProvider Registry -Name HKU -Root HKEY_USERS

Name           Used (GB)     Free (GB) Provider
Root                                            CurrentLocation
----           ---------     --------- --------      ----
```

```
                                                ---------------
HKU                                 Registry      HKEY_USERS

PS C:\WINDOWS\system32> New-PSDrive -PSProvider Registry -Name HKCR -Root HKEY_CLASSES_ROOT

Name           Used (GB)     Free (GB) Provider
Root                                                CurrentLocation
----           ---------     --------- --------      ----
                                                ---------------
HKCR                                Registry      HKEY_CLASSES_ROOT

PS C:\WINDOWS\system32> Get-PSDrive | Where-Object Provider -like *Registry*

Name           Used (GB)     Free (GB) Provider
Root                                                CurrentLocation
----           ---------     --------- --------      ----
                                                ---------------
HKCR                                Registry      HKEY_CLASSES_ROOT
HKCU                                Registry      HKEY_CURRENT_USER
HKLM                                Registry      HKEY_LOCAL_MACHINE
HKU                                 Registry      HKEY_USERS
```

Récupération d'information du registre

Get-ItemProperty

La commande Get-ItemProperty est utilisée pour récupérer l'information sur une valeur unique dans une clé. L'exemple suivant afficherait l'état du paramètre masquant les extensions de fichier connus dans l'explorateur Windows.

```
PS C:\Users\alexa> Get-ItemProperty -Path
"HKCU:\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced" -Name "HideFileExt"

HideFileExt  : 0
PSPath       :
Microsoft.PowerShell.Core\Registry::HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersio
n\Explorer
               \Advanced
PSParentPath :
Microsoft.PowerShell.Core\Registry::HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersio
```

```
n\Explorer
PSChildName  : Advanced
PSDrive      : HKCU
PSProvider   : Microsoft.PowerShell.Core\Registry
```

Sa valeur étant 0, les extensions de fichiers connues sont donc affichées dans l'explorateur de fichiers.

Elle peut être utilisée sur une clé sans mentionner de valeur pour afficher toutes les valeurs assignées à une clé. L'exemple suivant afficherait l'ensemble des applications qui démarreront à la connexion de l'utilisateur concerné.

```
PS C:\Users\alexa> Get-ItemProperty -Path
"HKCU:\Software\Microsoft\Windows\CurrentVersion\Run" | Select-Object -Property *


Discord                 : "C:\Users\alexa\AppData\Local\Discord\Update.exe" --processStart
Discord.exe
Steam                   : "C:\Program Files (x86)\Steam\steam.exe" -silent
electron.app.Base Camp™  : C:\Program Files (x86)\Mountain Base Camp\Base Camp.exe --process-
start-args --hidden
com.squirrel.Teams.Teams : C:\Users\alexa\AppData\Local\Microsoft\Teams\Update.exe --
processStart "Teams.exe"
                           --process-start-args "--system-initiated"
EADM                    : "C:\Program Files\Electronic Arts\EA Desktop\EA
Desktop\EALauncher.exe" -silent
EpicGamesLauncher       : "C:\Program Files (x86)\Epic
Games\Launcher\Portal\Binaries\Win64\EpicGamesLauncher.exe"
                           -silent -launchcontext=boot
HASS.Agent              : C:\Users\alexa\AppData\Roaming\LAB02
Research\HASS.Agent\HASS.Agent.exe
PSPath                  :
Microsoft.PowerShell.Core\Registry::HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVers
                           ion\Run
PSParentPath            :
Microsoft.PowerShell.Core\Registry::HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVers
                           ion
PSChildName             : Run
PSDrive                 : HKCU
PSProvider              : Microsoft.PowerShell.Core\Registry
```

Get-ChildItem

La commande Get-ChildItem permet de récupérer tous les éléments contenus dans une clé de registre et peut être exécutée récursivement pour aussi afficher les sous-clés. L'exemple suivant afficherait toutes les clés contenant les paramètres associés au logiciel 7-Zip ainsi que les valeurs associées à ces clés.

```
PS C:\Users\alexa> Get-ChildItem -Path "HKCU:\Software\7-Zip" -Recurse


    Hive: HKEY_CURRENT_USER\Software\7-Zip

Name                           Property
----                           --------
FM                             FolderShortcuts      : {}
                               FolderHistory        : {73, 0, 58, 0...}
                               PanelPath0           : I:\Documents\420-153-AT -
Conception\A2024\TP2_A24\
                               FlatViewArc0         : 0
                               PanelPath1           :
                               FlatViewArc1         : 0
                               ListMode             : 771
                               Position             : {162, 1, 0, 0...}
                               Panels               : {1, 0, 0, 0...}
                               ShowDots             : 1
                               ShowRealFileIcons    : 1
                               FullRow              : 1
                               ShowGrid             : 0
                               SingleClick          : 0
                               AlternativeSelection : 0
                               ShowSystemMenu       : 1
                               CopyHistory          : {70, 0, 58, 0...}


    Hive: HKEY_CURRENT_USER\Software\7-Zip\FM

Name                           Property
----                           --------
Columns                        RootFolder     : {1, 0, 0, 0...}
                               7-Zip.zip      : {1, 0, 0, 0...}
                               FSFolder       : {1, 0, 0, 0...}
                               7-Zip.Rar5     : {1, 0, 0, 0...}
                               7-Zip.Rar      : {1, 0, 0, 0...}
                               7-Zip.gzip     : {1, 0, 0, 0...}
```

```
                                7-Zip.Compound : {1, 0, 0, 0...}
                                7-Zip.7z       : {1, 0, 0, 0...}
                                7-Zip.tar      : {1, 0, 0, 0...}
                                7-Zip.Nsis     : {1, 0, 0, 0...}


    Hive: HKEY_CURRENT_USER\Software\7-Zip

Name                           Property
----                           --------
Options                        MenuIcons    : 1
                               CascadedMenu : 1
                               ContextMenu  : 2147487782
```

Création et définition de valeurs dans le registre

New-ItemProperty

La commande Set-ItemProperty permet de définir une valeur associée à une clé. La commande suivante activerait l'affichage avancé à l'écran de connexion d'un utilisateur (chaque étape plutôt que seulement "Veuillez patienter").

```
New-ItemProperty -Path "HKLM:\Software\Microsoft\Windows\CurrentVersion\Policies\System" -Name
"VerboseStatus" -Value 1
```

New-Item

La commande New-Item permet la création de nouvelles clés. Accompagné du paramètre -Force, la commande créera l'ensemble de la hiérarchie jusqu'à l'item créé mais si la clé était déjà existante, elle sera vidée. Il est donc généralement préférable de s'assurer de son existance avant de tenter sa création. Ceci peut facilement être réalisé à l'aide d'un "try catch" comme suit. Dans cet exemple, _Lab n'existe pas.

```
PS C:\Users\alexa> try {
>>     Get-Item -Path "HKCU:\_Lab" -ErrorAction Stop
>> }
>> catch {
>>     New-Item -Path "HKCU:\_Lab" -Force
>> }



    Hive: HKEY_CURRENT_USER
```

```
Name                           Property
----                           --------
_Lab
```

Dans cet exemple, _Lab existe et contient la valeur "Soleil123" nommée "Password". L'item n'est par conséquent pas remplacé.

```
PS C:\Users\alexa> try {
>>     Get-Item -Path "HKCU:\_Lab" -ErrorAction Stop
>> }
>> catch {
>>     New-Item -Path "HKCU:\_Lab" -Force
>> }


    Hive: HKEY_CURRENT_USER


Name                           Property
----                           --------
_Lab                           Password : Soleil123
```

Set-ItemProperty

La commande Set-ItemProperty permet de modifier une valeur assignée à une clé. Il est possible d'exécuter ceci conjointement à une vérification de l'existance de la clé pour s'assurer de la réussite de l'exécution de la commande comme suit. Dans cet exemple, le bouton de Windows Copilot sera retirée de la barre des tâches de l'utilisateur concerné.

```
$registryPath = "HKCU:\Software\Policies\Microsoft\Windows\WindowsCopilot" 
$registryValueName = "ShowCopilotButton"
$registryValueData = 0
try {
    Get-Item -Path $registryPath -ErrorAction Stop
}
catch {
    New-Item -Path $registryPath -Force
}
Set-ItemProperty -Path $registryPath -Name $registryValueName -Value $registryValueData
```

Modules supplémentaires

Il est possible d'ajouter des fonctionnalités à Powershell à l'aide de modules supplémentaires. Certains modules sont inclus avec des fonctionnalités avancées de Windows, d'autres sont disponibles sur Le PowerShell Gallery (https://www.powershellgallery.com/)

Mises à jour Windows

Installation

Le module "PSWindowsUpdate" permet de facilement gérer, automatiser et déclencher à distance des tâches en lien avec les mises à jour Windows. L'installation du module se déroule comme suit et doit être effectuée dans un terminal Powershell exécuté en tant qu'administrateur.

```
PS C:\WINDOWS\system32> Install-Module PSWindowsUpdate

Le fournisseur NuGet est requis pour continuer
PowerShellGet requiert le fournisseur NuGet, version 2.8.5.201 ou ultérieure, pour interagir avec les référentiels
NuGet. Le fournisseur NuGet doit être disponible dans « C:\Program Files\PackageManagement\ProviderAssemblies » ou « C:\Users\Info\AppData\Local\PackageManagement\ProviderAssemblies ». Vous pouvez également installer le fournisseur
NuGet en exécutant la commande « Install-PackageProvider -Name NuGet -MinimumVersion 2.8.5.201 -Force ». Voulez-vous
que PowerShellGet installe et importe le fournisseur NuGet maintenant ?
[O] Oui  [N] Non  [S] Suspendre  [?] Aide (la valeur par défaut est « O ») : O

Référentiel non approuvé
Vous installez les modules à partir d'un référentiel non approuvé. Si vous approuvez ce référentiel, modifiez sa valeur
 InstallationPolicy en exécutant l'applet de commande Set-PSRepository. Voulez-vous vraiment installer les modules à
partir de PSGallery ?
[O] Oui  [T] Oui pour tout  [N] Non  [U] Non pour tout  [S] Suspendre  [?] Aide (la valeur par défaut est « N ») : T

PS C:\WINDOWS\system32>
```

Une fois installé, le module peut être importé et sera prêt à l'utilisation à l'aide de la commande suivante :

```
#Permettre l'exécution de scripts
Set-ExecutionPolicy Unrestricted
#Importation du module installé
Import-Module PSWindowsUpdate
```

Utilisation

Toutes les commandes supportent l'argument "-ComputerName" pour exécuter ces actions sur un ordinateur distant. Si l'argument n'est pas mentionné, les actions seront entreprise sur la machine exécutant la commande.

Get-WindowsUpdate

La commande Get-WindowsUpdate permet de rechercher des mises à jour Windows disponibles de façon granulaire ou globale. Sans paramètres, elle cherchera toutes les mises à jour disponible. Dans le cas suivante, une mise à jour de pilote d'affichage est disponible.

```
PS C:\WINDOWS\system32> Get-WindowsUpdate -Verbose
COMMENTAIRES : VIDANGE-WINDOSE (2025-01-19 15:45:21): Connecting to Windows Update server.
Please wait...
COMMENTAIRES : Found [1] Updates in pre search
criteria                                                                    COMMENTAIRES :
Found [1] Updates in post search criteria

ComputerName Status     KB          Size Title
------------ ------     --          ---- -----
VIDANGE-W... -------               24MB Broadcom Inc. - Display - 9.17.8.9
```

Il est ensuite possible de procéder au téléchargement sans installer la mise à jour si on mentionne le paramètre -Download

```
PS C:\WINDOWS\system32> Get-WindowsUpdate -Download -Verbose
COMMENTAIRES : VIDANGE-WINDOSE (2025-01-19 15:51:08): Connecting to Windows Update server.
Please wait...
COMMENTAIRES : Found [1] Updates in pre search criteria
COMMENTAIRES : Found [1] Updates in post search criteria

Confirmer
Êtes-vous sûr de vouloir effectuer cette action ?
Opération « (2025-01-19 15:51:15) Broadcom Inc. - Display - 9.17.8.9[24MB] » en cours sur la
cible « VIDANGE-WINDOSE ».
```

```
[O] Oui  [T] Oui pour tout  [N] Non  [U] Non pour tout  [S] Suspendre  [?] Aide (la valeur par
défaut est « O ») : O

X ComputerName Result     KB          Size Title
- ------------ ------     --          ---- -----
1 VIDANGE-W... Accepted               24MB Broadcom Inc. - Display - 9.17.8.9
COMMENTAIRES : Accepted [1] Updates ready to Download
2 VIDANGE-W... Downloaded             24MB Broadcom Inc. - Display - 9.17.8.9
COMMENTAIRES : Downloaded [1] Updates ready to Install
```

On peut ensuite observer que le statut de la mise à jour a changé et qu'elle indique maintenant "D" pour indiquer qu'elle est téléchargée.

```
PS C:\WINDOWS\system32> Get-WindowsUpdate -Verbose
COMMENTAIRES : VIDANGE-WINDOSE (2025-01-19 15:51:53): Connecting to Windows Update server.
Please wait...
COMMENTAIRES : Found [1] Updates in pre search criteria
COMMENTAIRES : Found [1] Updates in post search criteria

ComputerName Status     KB          Size Title
------------ ------     --          ---- -----
VIDANGE-W... -D-----                24MB Broadcom Inc. - Display - 9.17.8.9
```

Il est maintenant possible de déclencher l'installation de la mise à jour avec le paramètre -Install

```
PS C:\WINDOWS\system32> Get-WindowsUpdate -Install -Verbose
COMMENTAIRES : VIDANGE-WINDOSE (2025-01-19 15:55:03): Connecting to Windows Update server.
Please wait...
COMMENTAIRES : Found [1] Updates in pre search criteria
COMMENTAIRES : Found [1] Updates in post search criteria

Confirmer
Êtes-vous sûr de vouloir effectuer cette action ?
Opération « (2025-01-19 15:55:09) Broadcom Inc. - Display - 9.17.8.9[24MB] » en cours sur la
cible « VIDANGE-WINDOSE ».
[O] Oui  [T] Oui pour tout  [N] Non  [U] Non pour tout  [S] Suspendre  [?] Aide (la valeur par
défaut est « O ») : T

X ComputerName Result     KB          Size Title
- ------------ ------     --          ---- -----
```

```
1 VIDANGE-W... Accepted              24MB Broadcom Inc. - Display - 9.17.8.9
COMMENTAIRES : Accepted [1] Updates ready to Download
2 VIDANGE-W... Downloaded            24MB Broadcom Inc. - Display - 9.17.8.9
COMMENTAIRES : Downloaded [1] Updates ready to Install
3 VIDANGE-W... Installed             24MB Broadcom Inc. - Display - 9.17.8.9
COMMENTAIRES : Installed [1] Updates
```

Get-WUHistory

La commande Get-WUHistory permet de consulter l'historique des mises à jour Windows installées. Le paramètre -MaxDate permet d'établir une date limite à partir de laquelle les résultats affichés s'arrêteront.

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

ComputerName Operationname  Result     Date                Title
------------ ------------   ------     ----                -----
VIDANGE-W... Installation   Succeeded  2025-01-19 15:55:22 Broadcom Inc. - Display - 9.17.8.9
VIDANGE-W... Installation   Succeeded  2025-01-19 15:37:32 Mise à jour de la sélection
disjointe pour Microsoft Defender ...
VIDANGE-W... Installation   Succeeded  2025-01-19 13:40:24 2025-01 Mise à jour cumulative pour
Windows 11 Version 24H2 po...
VIDANGE-W... Installation   Succeeded  2025-01-19 13:24:43 2025-01 Mise à jour cumulative pour
.NET Framework 3.5 pour et...
VIDANGE-W... Installation   Succeeded  2025-01-19 13:23:14 Outil de suppression de logiciels
malveillants Windows x64 - v...
VIDANGE-W... Installation   Succeeded  2025-01-19 13:15:38 Mise à jour de la sélection
disjointe pour Microsoft Defender ...
VIDANGE-W... Installation   Succeeded  2025-01-07 20:10:45 Mise à jour pour Microsoft Defender
Antivirus plateforme anti-...
VIDANGE-W... Installation   Succeeded  2025-01-07 19:20:46 2024-12 Mise à jour cumulative pour
Windows 11 Version 24H2 po...
VIDANGE-W... Installation   Succeeded  2025-01-07 18:58:18 2024-11 Mise à jour cumulative pour
.NET Framework 3.5 pour et...
VIDANGE-W... Installation   Succeeded  2025-01-07 18:55:23 Mise à jour de la sélection
disjointe pour Microsoft Defender ...
VIDANGE-W... Installation   Succeeded  2025-01-07 18:54:32 Outil de suppression de logiciels
malveillants Windows x64 - v...
VIDANGE-W... Installation   Succeeded  2025-01-07 18:51:58 Mise à jour pour Windows Security
platform - KB5007651 (versio...
VIDANGE-W... Installation   Succeeded  2025-01-07 18:32:59 Broadcom Inc. - System - 9.8.18.1
```

```
VIDANGE-W... Installation   Succeeded  2025-01-07 18:27:25 2024-11 Mise à jour pour Windows 11
Version 24H2 sur systèmes ...
```

Remove-WindowsUpdate

Vous avez effectué une mise à jour et quelque chose dans votre système a cassé? Pas de panique, vous pouvez désinstaller une mise à jour avec la commande "Remove-WindowsUpdate"!

Dans l'exemple suivant, nous désinstallerons une mise à jour pour la plateforme .NET 3.5 :

```
PS C:\WINDOWS\system32> Get-WindowsUpdate -IsInstalled

ComputerName Status     KB          Size Title
------------ ------     --          ---- -----
VIDANGE-W... -DI----    KB4052623    5MB Mise à jour pour la plateforme de logiciels anti-
programme malveillant de W...
VIDANGE-W... -DI----    KB4052623   13MB Mise à jour pour Microsoft Defender Antivirus
plateforme anti-programme mal...
VIDANGE-W... -DI----    KB5007651   38MB Mise à jour pour Windows Security platform -
KB5007651 (version 10.0.27703....
VIDANGE-W... -DI----    KB890830    76MB Outil de suppression de logiciels malveillants
Windows x64 - v5.131 (KB890830)
VIDANGE-W... -DI----    KB2267602    1GB Mise à jour de la sélection disjointe pour Microsoft
Defender Antivirus – 2...
VIDANGE-W... -DI----    KB5049622  142MB 2025-01 Mise à jour cumulative pour .NET Framework
3.5 pour et 4.8.1 pour W...
VIDANGE-W... -DI----    KB5050009   86GB 2025-01 Mise à jour cumulative pour Windows 11
Version 24H2 pour les systèm...

PS C:\WINDOWS\system32> Remove-WindowsUpdate -KBArticleID KB5049622

Confirmer
Êtes-vous sûr de vouloir effectuer cette action ?
Opération « (2025-01-19 16:23:30) Remove (WUApi) Windows Update: 2025-01 Mise à jour
cumulative pour .NET Framework 3.5
 pour et 4.8.1 pour Windows 11, version 24H2 pour les systèmes x64 (KB5049622) » en cours sur
la cible
« VIDANGE-WINDOSE ».
[O] Oui  [T] Oui pour tout  [N] Non  [U] Non pour tout  [S] Suspendre  [?] Aide (la valeur par
défaut est « O ») : O
```

Hide-WindowsUpdate

La commande Hide-WindowsUpdate permet de masquer certaines mises à jour pour éviter leur installation.

Dans l'exemple suivant, nous masquerons une mise à jour de Windows Defender :

```
PS C:\WINDOWS\system32> Hide-WindowsUpdate -KBArticleID KB2267602

Confirmer
Êtes-vous sûr de vouloir effectuer cette action ?
Opération « (2025-01-19 16:21:07) Hide Mise à jour de la sélection disjointe pour Microsoft
Defender Antivirus –
2267602 Ko (version 1.421.1442.0) – Canal actuel (large)[1GB] » en cours sur la cible
« VIDANGE-WINDOSE ».
[O] Oui  [T] Oui pour tout  [N] Non  [U] Non pour tout  [S] Suspendre  [?] Aide (la valeur par
défaut est « O ») : T

ComputerName Status     KB          Size Title
------------ ------     --          ---- -----
VIDANGE-W... ---H--     KB2267602    1GB Mise à jour de la sélection disjointe pour Microsoft
Defender Antivirus – 2...
```

Le statut "H" indique que la mise à jour est masquée et ne sera pas installée.

Winget

Winget est un module permettant l'installation automatisée d'applications disponibles dans le répertoire logiciel https://winget.run/. Il est à noter que la majorité des applications seront installées dans le %AppData% de l'utilisateur installant un logiciel. Ces logiciels sont donc unique à chaque utilisateur impliquant que si 20 utilisateurs du poste de travail désirent la même application, elle sera installée en 20 copies sur le poste. On réfère à ce type d'installation à "per-user" ou par utilisateur. La majorité des gestionnaires de logiciels du genre pour Windows fonctionnent de cette façon, l'avantage étant qu'un utilisateur n'a pas besoin de droits d'administrateur pour installer ces logiciels.

Cette méthode peut être une belle façon d'automatiser le déploiement logiciel sur un ordinateur personnel à un seul utilisateur puisqu'une simple tâche planifiée exécutant "winget update" et "winget upgrade --all" met à jour tous les logiciels pour l'utilisateur.

Il peut être installé en tant que module Powershell de la façon suivante (en tant qu'administrateur) :

```
PS C:\WINDOWS\system32> Install-Module
Microsoft.WinGet.Client
Le fournisseur NuGet est requis pour continuer
PowerShellGet requiert le fournisseur NuGet, version 2.8.5.201 ou ultérieure, pour interagir
avec les référentiels
NuGet. Le fournisseur NuGet doit être disponible dans « C:\Program
Files\PackageManagement\ProviderAssemblies » ou «
C:\Users\Info\AppData\Local\PackageManagement\ProviderAssemblies ». Vous pouvez également
installer le fournisseur
NuGet en exécutant la commande « Install-PackageProvider -Name NuGet -MinimumVersion 2.8.5.201
-Force ». Voulez-vous
que PowerShellGet installe et importe le fournisseur NuGet maintenant ?
[O] Oui  [N] Non  [S] Suspendre  [?] Aide (la valeur par défaut est « O ») : O

Référentiel non approuvé
Vous installez les modules à partir d'un référentiel non approuvé. Si vous approuvez ce
référentiel, modifiez sa valeur
 InstallationPolicy en exécutant l'applet de commande Set-PSRepository. Voulez-vous vraiment
installer les modules à
partir de PSGallery ?
```

```
[O] Oui  [T] Oui pour tout  [N] Non  [U] Non pour tout  [S] Suspendre  [?] Aide (la valeur par défaut est « N ») : T
PS C:\WINDOWS\system32>
```

À sa première utilisation, vous devrez accepter les termes et conditions du module comme suit. La commande "update" au même titre que pour "apt" sous Ubuntu et Debian vérifie les mises à jour disponibles et la commande "upgrade --all" permet d'appliquer les mises à jour disponibles. Certaines mises à jour seront silencieuses en arrière plan et d'autres afficheront une fenêtre de progression mais tout sera toujours automatisé.

```
PS C:\WINDOWS\system32> winget update
The `msstore` source requires that you view the following agreements before using.
Terms of Transaction: https://aka.ms/microsoft-store-terms-of-transaction
The source requires the current machine's 2-letter geographic region to be sent to the backend service to function properly (ex. "US").

Do you agree to all the source agreements terms?
[Y] Yes  [N] No: Y
Name                                    Id                            Version
Available           Source
-------------------------------------------------------------------------------------------------
-------------------------
Microsoft Visual C++ 2015-2022 Redistribut… Microsoft.VCRedist.2015+.x86 14.36.32532.0
14.42.34433.0       winget
Microsoft Visual C++ 2015-2022 Redistribut… Microsoft.VCRedist.2015+.x64 14.36.32532.0
14.42.34433.0       winget
Microsoft OneDrive                          Microsoft.OneDrive           24.226.1110.0004
24.244.1204.0003    winget
Microsoft Teams                             Microsoft.Teams              24295.605.3225.8804
24335.208.3315.1951 winget
App Installer                               Microsoft.AppInstaller       1.24.25199.0
1.24.25200.0        winget
Power Automate                              Microsoft.DevHome            0.0.0.0
0.1901.687.0        winget
6 upgrades available.
```

```
PS C:\WINDOWS\system32> winget upgrade --all
Name                                    Id                            Version
Available           Source
-------------------------------------------------------------------------------------------------
```

```
-------------------------
Microsoft Visual C++ 2015-2022 Redistribut… Microsoft.VCRedist.2015+.x86 14.36.32532.0
14.42.34433.0       winget
Microsoft Visual C++ 2015-2022 Redistribut… Microsoft.VCRedist.2015+.x64 14.36.32532.0
14.42.34433.0       winget
Microsoft OneDrive                          Microsoft.OneDrive           24.226.1110.0004
24.244.1204.0003    winget
Microsoft Teams                             Microsoft.Teams              24295.605.3225.8804
24335.208.3315.1951 winget
App Installer                               Microsoft.AppInstaller       1.24.25199.0
1.24.25200.0        winget
Power Automate                              Microsoft.DevHome            0.0.0.0
0.1901.687.0        winget
6 upgrades available.

Installing dependencies:
This package requires the following dependencies:
  - Packages
      Microsoft.VCLibs.Desktop.14 [>= 14.0.33728.0]
      Microsoft.UI.Xaml.2.8 [>= 8.2310.30001.0]
      Microsoft.WindowsAppRuntime.1.5
(1/6) Found Microsoft Visual C++ 2015-2022 Redistributable (x86)
[Microsoft.VCRedist.2015+.x86] Version 14.42.34433.0
Starting package install...
Successfully installed

#[...plein de barres de chargement]
```

La commande "winget --help" vous aidera à trouver les arguments nécessaires à l'utilisation du module :

```
PS C:\WINDOWS\system32> winget --help
Gestionnaire de package Windows v1.9.25200
Copyright (c) Microsoft Corporation. Tous droits réservés.

L’utilitaire de ligne de commande winget permet d’installer des applications et d’autres
packages à partir de la ligne de commande.

Utilisation : winget  [<commande>] [<options>]
```

```
Les commandes suivantes sont disponibles :
  install    Installe le package donné
  show       Affiche des informations sur un package
  source     Gérer les sources des packages
  search     Rechercher et afficher des informations de base sur les packages
  list       Afficher les packages installés
  upgrade    Affiche et effectue les mises à niveau disponibles.
  uninstall  Désinstallation du paquet donné
  hash       Assistant pour le hachage des fichiers d’installation
  validate   Valide un fichier manifeste
  settings   Ouvrir les paramètres ou définir les paramètres d’administrateur
  features   Affiche le statut des fonctionnalités expérimentales
  export     Exporte une liste des packages installés
  import     Installe tous les packages dans un fichier
  pin        Gérer les broches de package
  configure  Configure le système dans un état souhaité
  download   Télécharge le programme d’installation à partir d’un package donné
  repair     Répare le package sélectionné
```

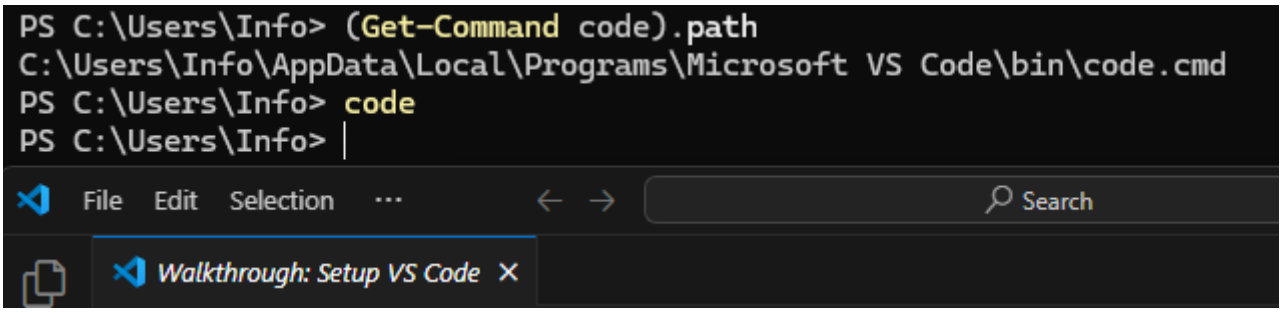
La commande "winget search [application]" vous permettrait de vérifier si une application est disponible sur leur répertoire. Dans l'exemple suivant,

```
PS C:\WINDOWS\system32> winget search VSCode
Nom                                ID                                  Version
Correspondance              Source
-------------------------------------------------------------------------------------------
-------------------------
Microsoft Visual Studio Code       Microsoft.VisualStudioCode          1.96.4       Moniker:
vscode             winget
Codium                             Alex313031.Codium                   1.89.1.24132 Tag:
vscode                 winget
TheiaBlueprint                     EclipseFoundation.TheiaBlueprint    1.44.0       Tag:
vscode                 winget
TheiaIDE                           EclipseFoundation.TheiaIDE          1.54.0       Tag:
vscode                 winget
Huawei QuickApp IDE                Huawei.QuickAppIde                  14.0.1       Tag:
vscode                 winget
DevPod                             LoftLabs.DevPod                     0.6.9        Tag:
vscode                 winget
VSCodium                           VSCodium.VSCodium                   1.96.4.25017 Tag:
```

```
vscode                  winget
VSCodium Insiders                   VSCodium.VSCodium.Insiders          1.97.0.24352 Tag:
vscode                  winget
vscli                               michidk.vscli                       1.1.0        Tag:
vscode                  winget
Upgit                               pluveto.Upgit                       0.2.18       Tag:
vscode                  winget
MrCode                              zokugun.MrCode                      1.82.0.23253 Tag:
vscode                  winget
Microsoft Visual Studio Code CLI    Microsoft.VisualStudioCode.CLI      1.96.4       Moniker:
vscode-cli          winget
Microsoft Visual Studio Code Insi… Microsoft.VisualStudioCode.Insiders 1.97.0       Moniker:
vscode-insiders     winget
Microsoft Visual Studio Code Insi… Microsoft.VisualStudioCode.Insider… 1.97.0       Moniker:
vscode-insiders-cli winget
markdown-oxide                      FelixZeller.markdown-oxide          0.23.1       Tag:
vscode-language-support winget
```

Connaissant maintenant l'ID de l'application recherchée, nous pouvons maintenant l'installer!

```
PS C:\WINDOWS\system32> winget install Microsoft.VisualStudioCode
Trouvé Microsoft Visual Studio Code [Microsoft.VisualStudioCode] Version 1.96.4
La licence d’utilisation de cette application vous est octroyée par son propriétaire.
Microsoft n’est pas responsable des paquets tiers et n’accorde pas de licences à ceux-ci.
Téléchargement en cours
https://vscode.download.prss.microsoft.com/dbazure/download/stable/cd4ee3b1c348a13bafd8f9ad806
0705f6d4b9cba/VSCodeUserSetup-x64-1.96.4.exe
  ██████████████████████████████  100 MB /  100 MB
Le code de hachage de l’installation a été vérifié avec succès
Démarrage du package d’installation... Merci de patienter.
Installé correctement

# Redémarrage du shell pour recharger les variables d'environnement
PS C:\Users\Info> (Get-Command code).path
C:\Users\Info\AppData\Local\Programs\Microsoft VS Code\bin\code.cmd
```



La commande "winget list" permet de voir l'ensemble des logiciels et indiquera lesquelles ont été installées et sont gérées par winget.

```
PS C:\Users\Info> winget list
Nom                                            ID
Version             Source
-------------------------------------------------------------------------------------------
------------------------
VMware Tools                                   ARP\Machine\X64\{6070BE95-B84D-40FE-8ABD-C70…
12.4.5.23787635
Microsoft Edge                                 Microsoft.Edge
132.0.2957.115      winget
Microsoft Visual C++ 2015-2022 Redistributab… Microsoft.VCRedist.2015+.x64
14.42.34433.0       winget
Microsoft Visual C++ 2015-2022 Redistributab… Microsoft.VCRedist.2015+.x86
14.42.34433.0       winget
Microsoft OneDrive                             Microsoft.OneDrive
24.244.1204.0003    winget
Microsoft Visual Studio Code (User)            Microsoft.VisualStudioCode
1.96.4              winget
Microsoft Clipchamp                            MSIX\Clipchamp.Clipchamp_3.1.11920.0_neutral…
3.1.11920.0
Microsoft Teams                                Microsoft.Teams
24335.208.3315.1951 winget
AV1 Video Extension                            MSIX\Microsoft.AV1VideoExtension_1.3.4.0_x64…
1.3.4.0
AVC Encoder Video Extension                    MSIX\Microsoft.AVCEncoderVideoExtension_1.1.…
1.1.3.0
Améliorations de la compatibilité des applic… MSIX\Microsoft.ApplicationCompatibilityEnhan…
1.2411.16.0
Actualités                                     MSIX\Microsoft.BingNews_4.55.62231.0_x64__8w…
4.55.62231.0
Recherche Web de Microsoft Bing                MSIX\Microsoft.BingSearch_1.1.3.0_x64__8weky…
```

```
1.1.3.0
MSN Météo                                       MSIX\Microsoft.BingWeather_4.54.63007.0_x64_…
4.54.63007.0
Copilot                                         MSIX\Microsoft.Copilot_1.24122.48.0_x64__8we…
1.24122.48.0
Installeur d'applications                       Microsoft.AppInstaller
1.24.25200.0        winget
```