



# **SYSTEMES D'EXPLOITATION**

## **EXEMPLE DE WINDOWS**

## Présentation de l'Enseignant- Formateur :

**M. Ndiamé CAMARA**

Ingénieur en Informatique industrielle

Administrateur Systèmes



## Séquence 5 : PRÉSENTATION DE WINDOWS

- Se familiariser avec l'environnement du système d'exploitation Windows
- Sélectionner le SE adéquat par rapport aux besoins

**DUREE: 4 heures**

## **Prérequis :**

- Initiation à l'informatique
- Historique des ordinateurs

## **Consignes pour l'apprenant :**

- Télécharger le document de présentation du cours
- Faire l'évaluation formative
- Animer le forum de partage
- Effectuer les projets de classe

Evolution

Caractéristiques

Architecture de la console Windows

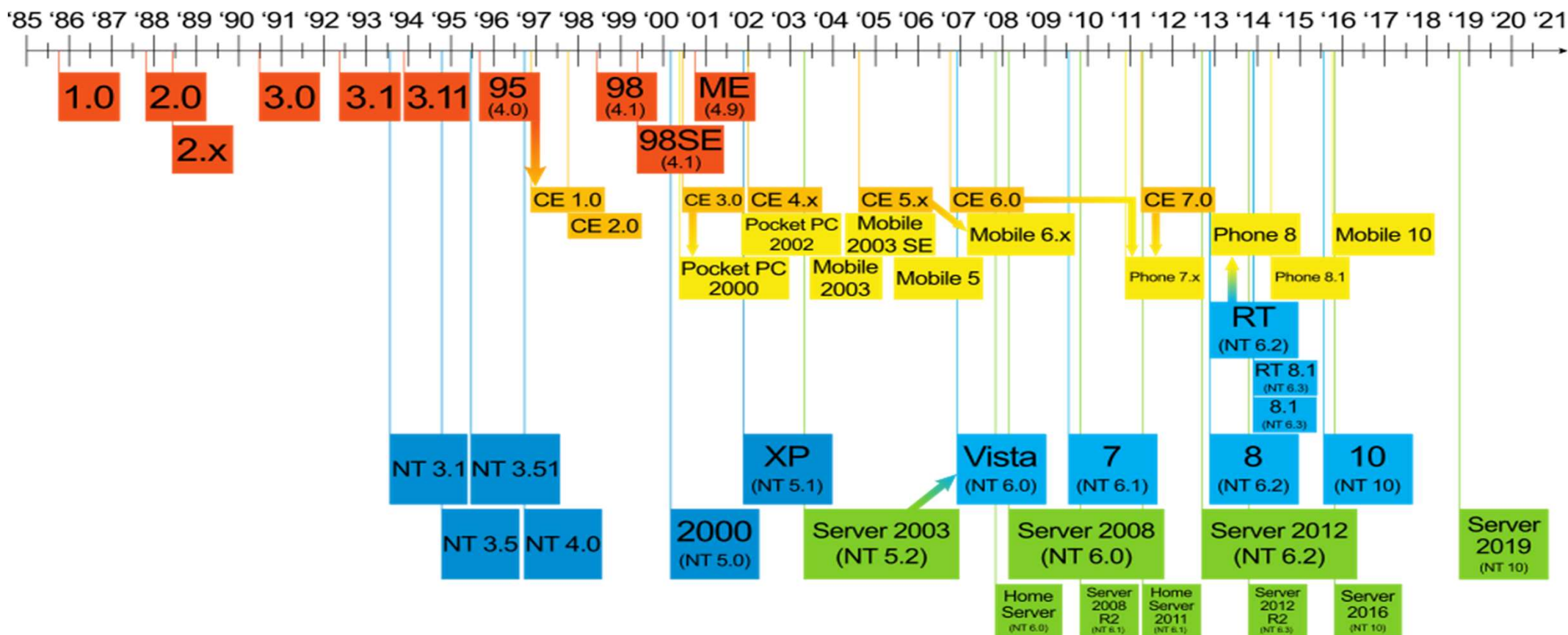
API Windows (Application Programming Interface)

La base des registres

Introduction à Windows POWERSHELL

# PRESENTATION DE WINDOWS

## Evolution





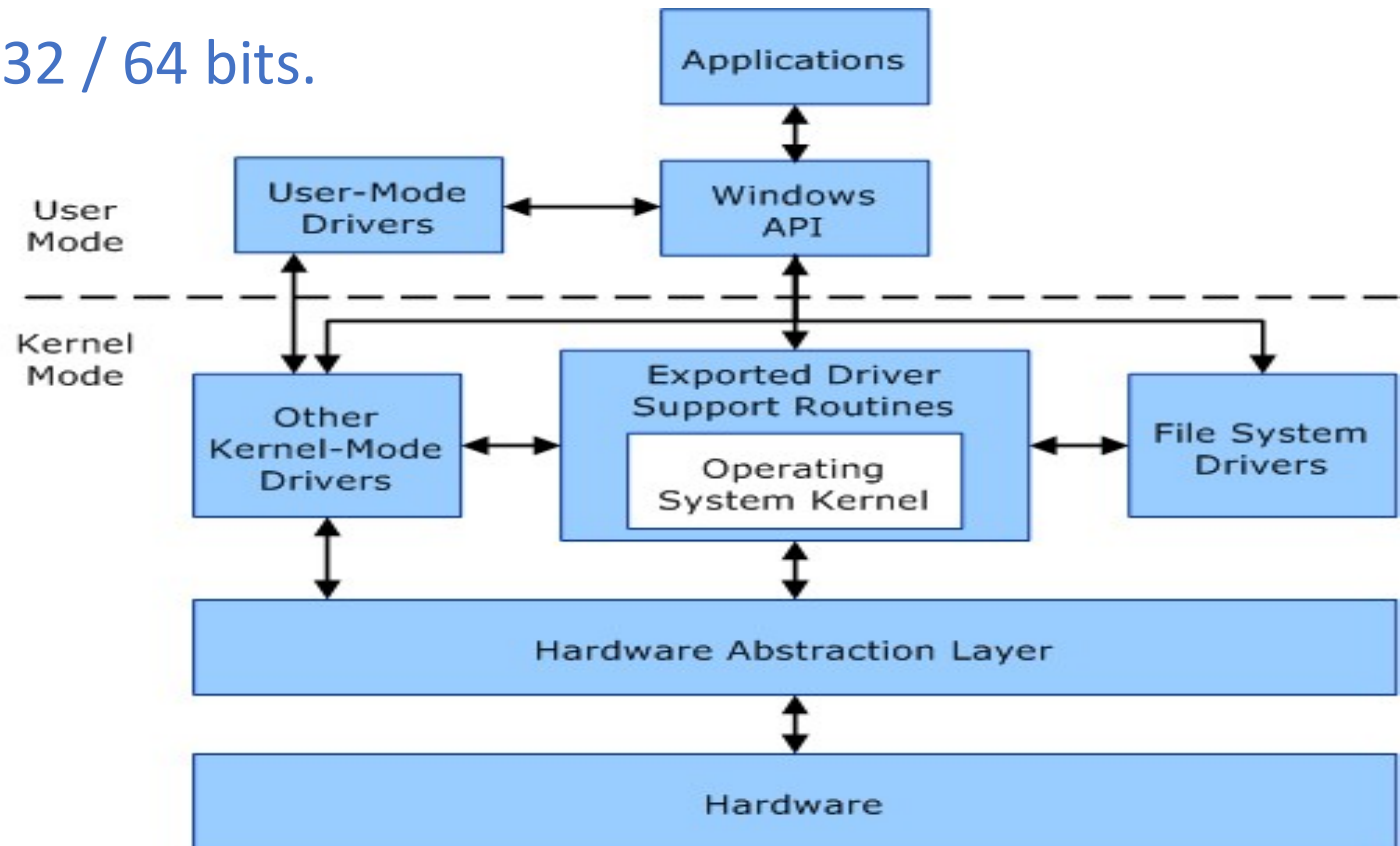
# PRESENTATION DE WINDOWS

## Evolution



## Caractéristiques

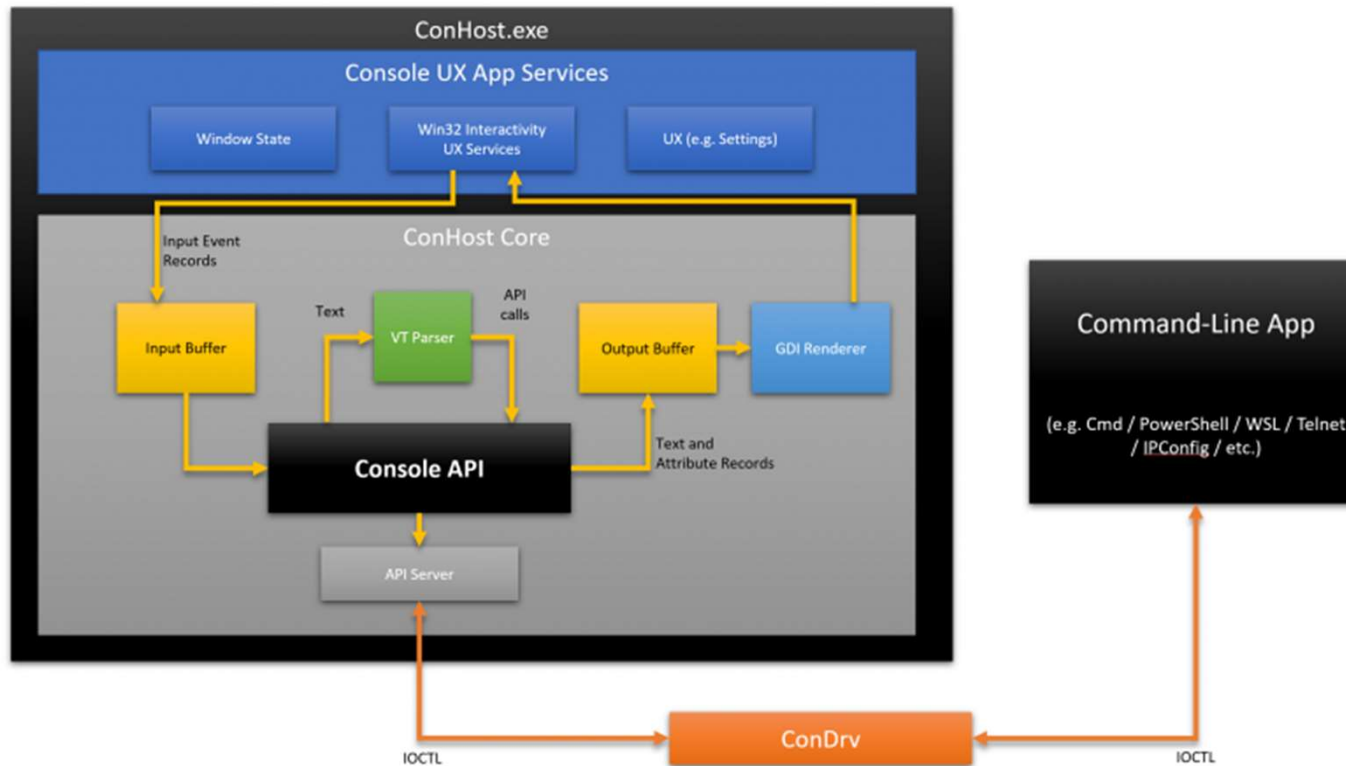
- Système d'exploitation 32 / 64 bits.
- Noyau : KERNEL
- Système de fichier : NTFS
- RAM conseillée :  
> 4 Go
- Espace disque  
conseillé :  
> 25 Go





## Architecture de la console Windows

Console Architecture in Windows 10 Spring 2018 Update (1803)

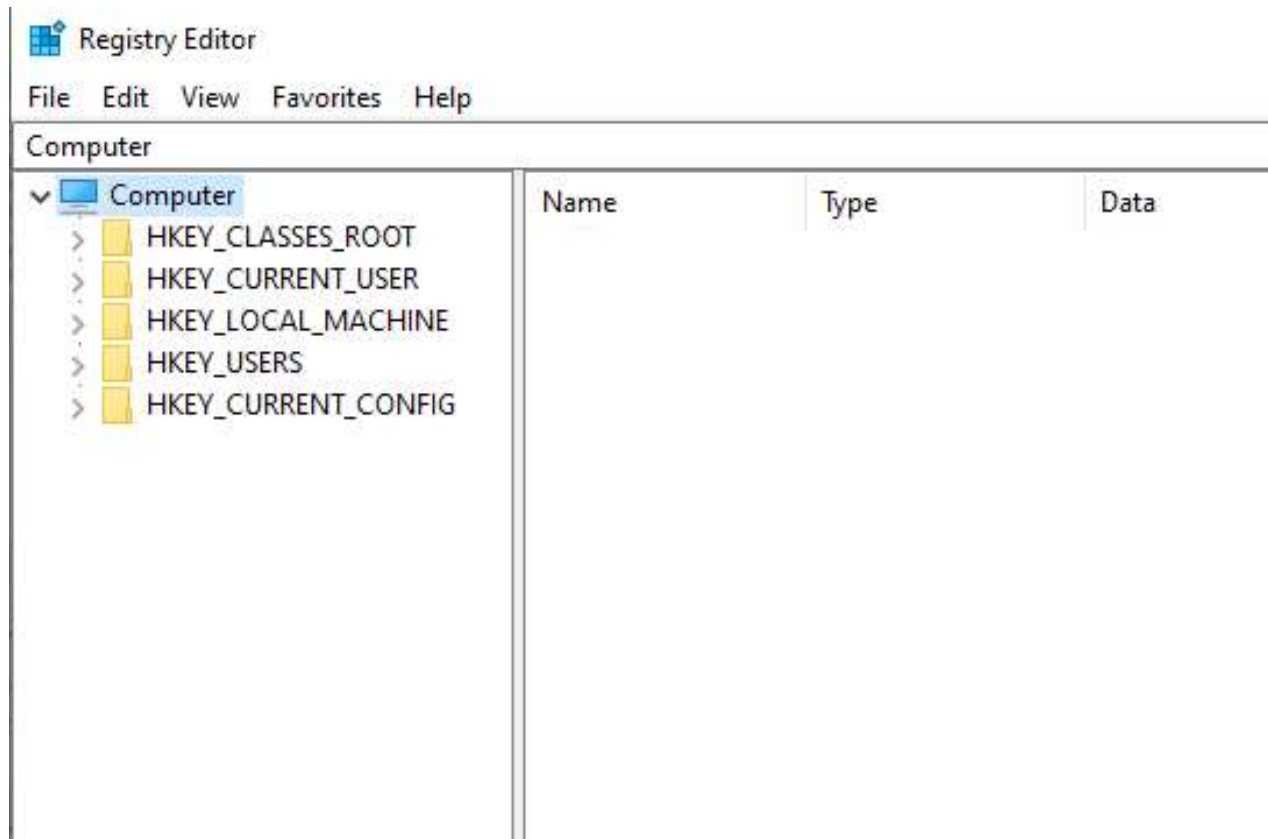


- CMD
- POWERSHELL
- WSL

## API Windows (Application Programming Interface)

- Services de base : (kernel32.dll)
- Interface graphique : (gdi32.dll)
- Interface utilisateur : (user32.dll, comctl32.dll...)
- Boîtes de dialogue communes : (comdlg32.dll)
- Bibliothèque de contrôles communs : barres d'outils, de statuts...
- Shell Windows : interface permettant d'accéder aux fonctions avancées
- Services réseau (NetBIOS, Winsock, RPC)

## La base des registres



# Introduction à Windows POWERSHELL



# POWERSHELL

# Introduction à Windows POWERSHELL

---

**POURQUOI POWERSHELL ?**

**ACCEDER A POWERSHELL**

**LES APPLETS DE COMMANDES**

**QUELQUES APPLETS DE COMMANDES UTILES**

**AUTORISER LE SCRIPTING POWERSHELL**

## Pourquoi POWERSHELL

- Les commandes natives Windows (CMD) n'offrent pas autant de possibilités que les SHELL de UNIX ou LINUX (bash par exemple)
- Windows offre une première amélioration avec le WMI (Windows Management Instrumentation) : technologie de Microsoft, dont le but est de prendre en charge des différents environnements opérationnels de Windows.
- pour en savoir plus sur WMI :

<https://docs.microsoft.com/en-us/windows/win32/wmisdk/wmi-start-page>

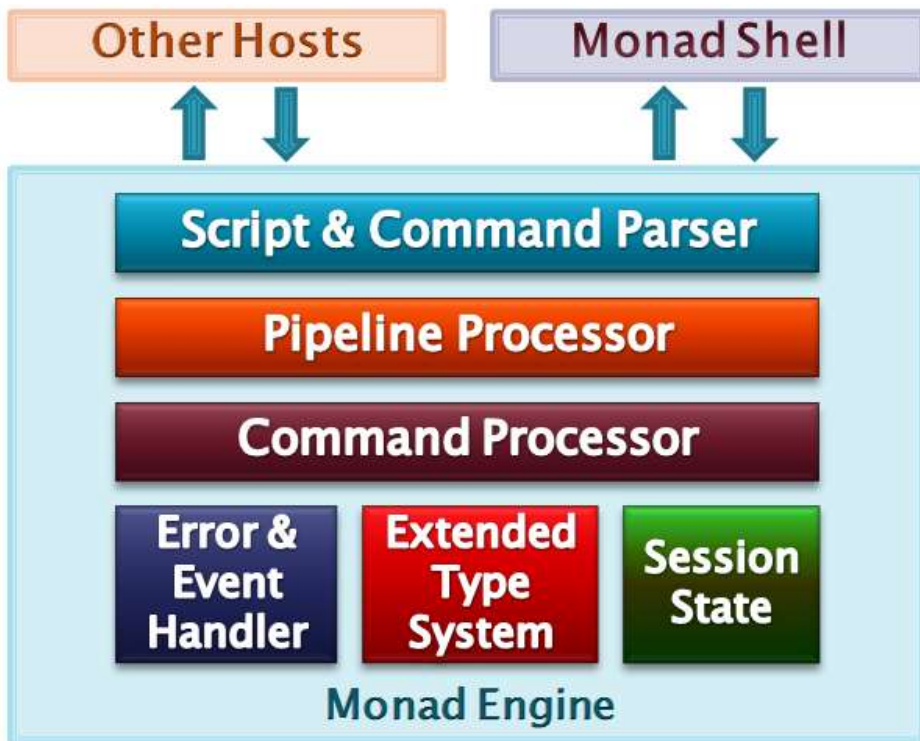
- Les commandes POWERSHELL apportent nombre d'améliorations permettant d'accéder à des fonctions bas niveau et d'automatiser les tâches d'administration système



## Introduction à Windows POWERSHELL

- Depuis 2006, Microsoft propose une nouvelle interface en ligne de commande basée sur le Framework .NET.
- Permet aux administrateurs systèmes d'effectuer la plupart des tâches complexes en programmant des scripts avec Visual Basic, C# ou PERL.
- Programmation plus simple, sans aucune compilation.
- Cependant c'est lorsque l'on utilise les objets .NET que l'on exploite au maximum la puissance de Powershell.
- permet l'exécution de script très poussé orienté objet

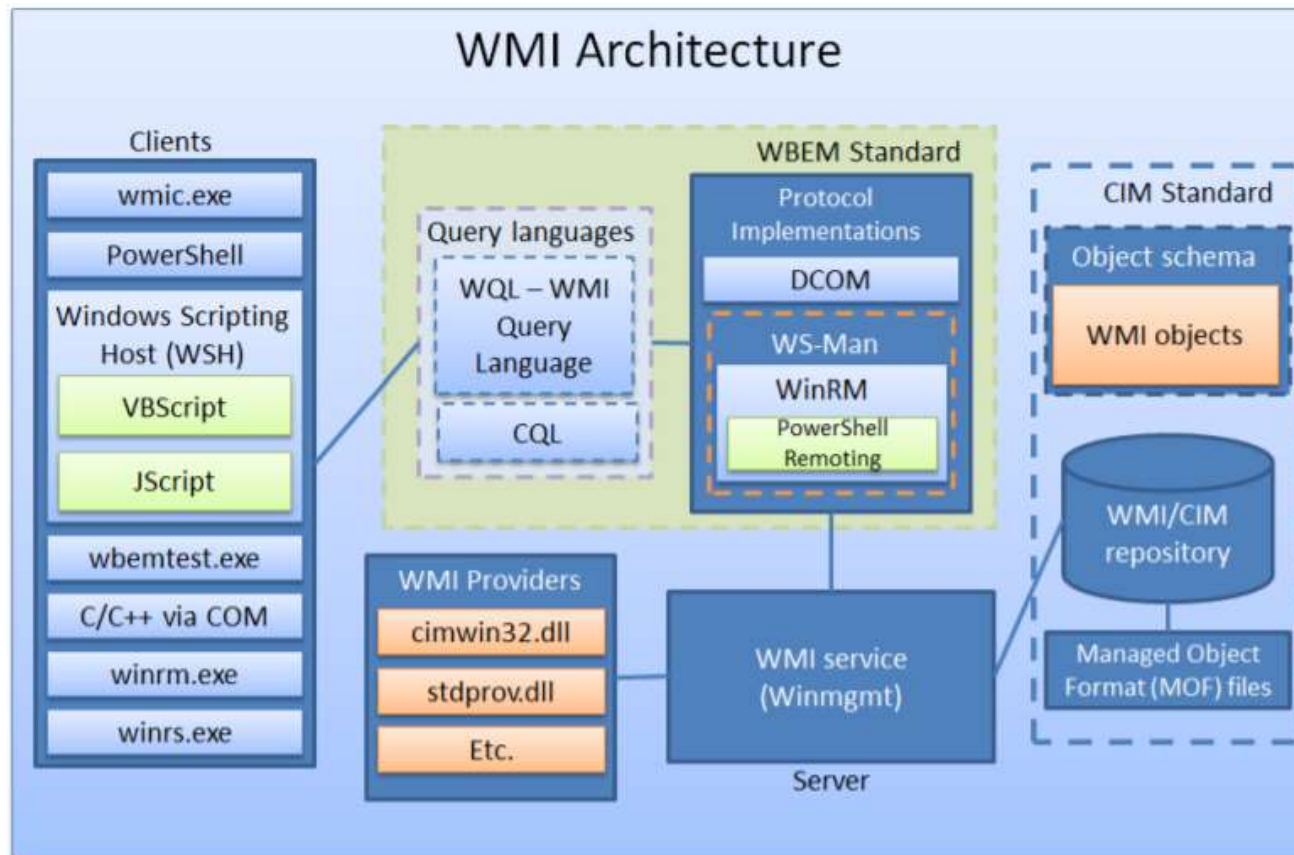
# Introduction à Windows POWERSHELL



- **Script/Parser** : Traite les constructions du langage telles que les scripts, les prédicats, les conditions, etc.
- **Pipeline Processor** : Gère les communications entre les cmdlets via des pipes.
- **Command Processor** : Gère l'exécution des cmdlets, l'enregistrement et les métadonnées associées.
- **Session State** : Gère les ensembles de données utilisés par un cmdlet lors de son exécution.
- **Error et Event Handler** : Gère les exceptions et les événements.
- **Extended Type System** : Fournit une interface commune d'accès aux propriétés, méthodes, etc.

# Introduction à Windows POWERSHELL

## Interaction entre POWERSHELL et WMI



<https://docs.microsoft.com/en-us/windows/win32/wmisdk/wmi-start-page>

<https://pandorafms.com/blog/fr/quest-ce-que-wmi/>

# Introduction à Windows POWERSHELL

- Un **Cmdlet** (prononcé "Commande-let") est une commande qui se présente sous la forme d'une instance de classe .NET. Ce n'est donc pas un simple exécutable.
- Les Cmdlets peuvent être réalisés avec n'importe quel langage .NET ou à l'aide du langage de script PowerShell. Pour afficher les Cmdlets disponibles, il faut utiliser la commande **Get-Command**.
- Le nommage d'un Cmdlet respecte certaines règles afin que les utilisateurs puissent les mémoriser ou les retrouver facilement.
- Syntaxe :

**Verbe-Nom [-parametre0] arg0] [-parametre1 arg1]**

## ACCEDER A POWERSHELL

- Deux méthodes pour accéder à powershell :
  1. Cliquez sur Démarrer > Exécuter.
  2. Saisissez « PowerShell ISE » dans la boîte de dialogue Exécuter et cliquez sur OK.

## OU BIEN

1. Cliquez sur Démarrer > Exécuter ou Windows + R.
2. Saisissez « Powershell ISE », puis cliquez sur OK pour ouvrir une invite de commande.

## LES APPLETS DE COMMANDES

- Les commandes native de Windows fonctionnent !
- Exemples : Ping, IPConfig, calc, notepad, mspaint

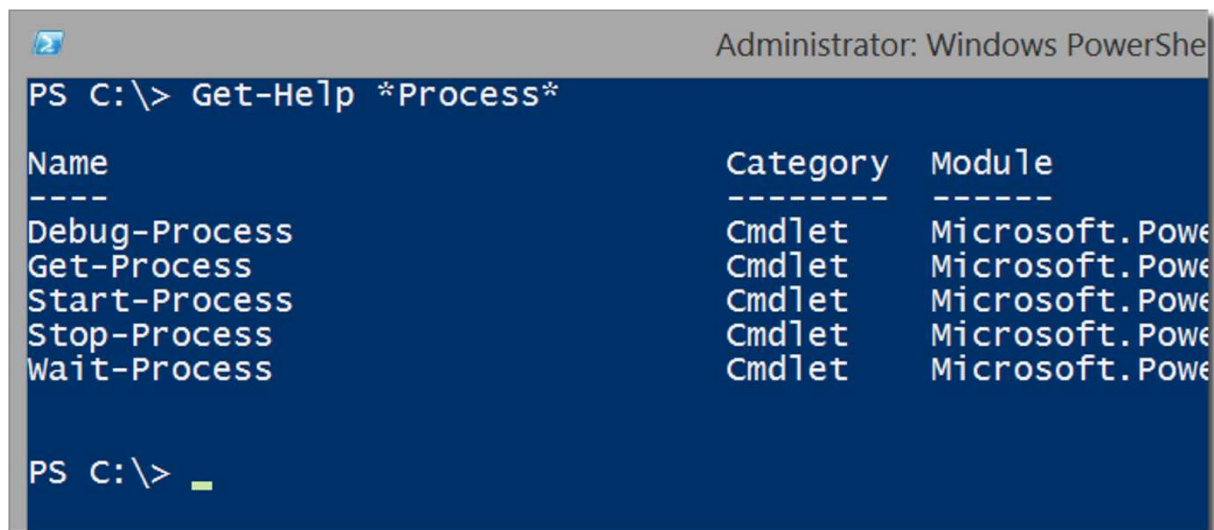
Equivalences commandes CMD et POWERSHELL

- cls = Clear-Host
- cd = Set-Location
- dir, ls = Get-Childitem
- type, cat = Get-Content
- Copy, cp = Copy-item



## LES APPLETS DE COMMANDES

- Obtenir de l'aide : commande « get-help »

A screenshot of a Windows PowerShell console window titled "Administrator: Windows PowerShell". The prompt is "PS C:\>". The command "Get-Help \*Process\*" has been entered, and the output is a table with three columns: Name, Category, and Module. The table lists five commands: Debug-Process, Get-Process, Start-Process, Stop-Process, and Wait-Process, all categorized as Cmdlet and associated with the Microsoft.PowerShell module.

```
PS C:\> Get-Help *Process*

Name                Category  Module
----                -
Debug-Process       Cmdlet    Microsoft.PowerShell
Get-Process         Cmdlet    Microsoft.PowerShell
Start-Process       Cmdlet    Microsoft.PowerShell
Stop-Process        Cmdlet    Microsoft.PowerShell
Wait-Process        Cmdlet    Microsoft.PowerShell

PS C:\> _
```

## QUELQUES APPLETS DE COMMANDES UTILES

Exécuter la commande suivante :

```
Get-Process > C:\ISEPDD\getprocess.log
```

Exemple de commande pour interrompre l'exécution d'un processus :

```
stop-process -Name "winword"      Ou bien :
```

```
stop-process -FilePath "C:\Program Files (x86)\Microsoft Office\root\Office16\WINWORD.EXE"
```

Connaître la version de Powershell

```
Get-Host | Select-Object version
```

## QUELQUES APPLETS DE COMMANDES UTILES

Commande de création de dossier avec la commande *new-item*

```
New-Item -Path C:\ -name temp1 -ItemType Directory
```

Pour créer un sous-dossier « exos » dans le répertoire « temp1 »

```
New-Item -Path C:\temp1 -name exos -ItemType Directory
```

Commande de création de fichier : (toujours avec *new-item*)

```
New-Item -Path 'C:\temp1\exos\exemple.txt' -ItemType file (Fichier vide)
```

Connaître la taille des volumes de mémoire de masse

```
Get-CimInstance -Class Win32_LogicalDisk | Select-Object -Property Name,FreeSpace
```

## LE SCRIPTING POWERSHELL

- Autoriser le scripting Powershell en saisissant la commande  
« **Set-ExecutionPolicy** RemoteSigned »

Autres paramètres :

Restricted

Unrestricted

AllSigned

RemoteSigned

Bypass

Undefined

## QUELQUES SCRIPTS POWERSHELL

Exemple de script utilisant la synthèse vocale,  
*Saisir les lignes de commandes ci-dessous puis sauvegarder le tout sous un nom de fichier avec extension **ps1** (**script1.ps1** par exemple)*

```
Add-Type -AssemblyName System.speech  
$speak = New-Object System.Speech.Synthesis.SpeechSynthesizer  
$speak.Speak("Je donne ma langue au chat")
```

Remplacer la phrase en vert par une chaîne de caractères qui fera l'objet d'un traitement de synthèse vocale

Ouvrir le script à partir de Powershell ISE puis l'exécuter

## QUELQUES SCRIPTS POWERSHELL

Autre exemple de script :

*Saisir les lignes de commandes ci-dessous puis sauvegarder le tout sous un nom de fichier avec extension **ps1** (**script2.ps1** par exemple)*

```
New-Item -Path C:\ -name temp1 -ItemType Directory  
New-Item -Path C:\temp1 -name exos -ItemType Directory  
New-Item -Path 'C:\temp1\exos\exemple.txt' -ItemType file
```

Ouvrir le script à partir de Powershell ISE puis l'exécuter



## QUELQUES SCRIPTS POWERSHELL

Script 3 :

Mission impossible.ps1

Script 4 :

Marche imperiale.ps1

# Introduction à Windows POWERSHELL

## POWERSHELL SUR UNE PLATEFORME ANDROID

<https://www.youtube.com/watch?v=Ucl-TIMSWik>

## RESSOURCES POWERSHELL

<https://docs.microsoft.com/en-us/powershell/scripting/windows-powershell/install/installing-windows-powershell?view=powershell-7.1>

A BIENTOT