

26 janvier 2023

Bulgheroni Ruben (cp-20RUB)

ceff INDUSTRIE

PIINF-3-TSYS-13

RAPPORT PROJET POWERSHELL

CHOCOLATEY

Table des matières

[1. Introduction 2](#_Toc124402504)

[2. Elément 1 3](#_Toc124402505)

[3. Elément 2 3](#_Toc124402506)

[4. Elément 3 5](#_Toc124402507)

[5. Conclusion 5](#_Toc124402508)

[6. Cahier des charges 5](#_Toc124402509)

[7. Source 5](#_Toc124402510)

# Introduction

**Dans ce rapport il y a :**

* Une introduction
* 3 Eléments de mon projet
* Une conclusion
* Le cahier des charges
* Les sources

**Présentation de mon projet :**

Mon projet doit permettre d’installer des logiciels automatiquement avec le langage de programmation PowerShell.

Pour ça je me suis aidé d’un outil qui se nomme Chocolatey. Cet outil permet d’installer des applications (comme Notepad ++) en ligne de commande.

**Développement du projet :**

Concrètement, mon projet fonctionne avec le langage de programmation PowerShell, et dans le projet PowerShell il y a 4 scripts qui fonctionnent différemment.  
  
Il y a le script « InstallChoco.ps1 » qui donne l’installation du programme de base qui est Chocolatey.

Le script « ProjetChoco.ps1 » dans ce script là il y a un menu avec 4 options qui permet d’avoir un script avec un plus large utilisation.

Le script « Popup.ps1 » qui fait une demande pour exécuter le scripte ou non.

Le script « PopupAdmin.ps1 » qui fait une Popup qui dit à l’utilisateur de passer en mode Administrateur.

Je vais présenter plus tard dans le rapport 3 éléments essentielle de mes scripts.

**But de mon projet :**

Le but final est d’avoir un script « utilitaire » qui permet de faciliter l’installation des logiciels et qui soit simple à comprendre.  
  
Retrouver le projet avec la dernière mise a jour sur mon **GITHUB :**

<https://github.com/RubenBulgheroni/Projet-CHOCO>

# Elément 1

Le premier élément va être sur l’utilisation de Chocolatey et son implémentation à PowerShell.

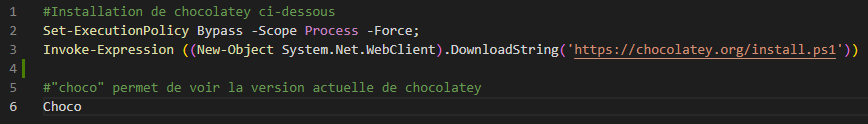
Pour ça j’ai dû faire un script qui se nomme InstallChoco.ps1.

**Dans ce script j’ai dû mettre les commandes suivantes :**

Set-ExecutionPolicy Bypass -Scope Process -Force;

Invoke-Expression ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.o §34rg/install.ps1'))

*#Ces commandes-là permette l’installation de Chocolatey sur PowerShell*



Ce code PowerShell sert à installer l'outil de gestion de paquet Chocolatey en utilisant PowerShell. Il y a deux commandes principales :

La première commande "Set-ExecutionPolicy Bypass -Scope Process -Force" permet de contourner la politique d'exécution de PowerShell pour éviter tout blocage de sécurité lors de l'exécution de scripts.

La deuxième commande "Invoke-Expression ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))" télécharge et exécute le script d'installation de Chocolatey à partir du site Web chocolatey.org.

Donc, l'ensemble de ces deux commandes permet de télécharger et installer Chocolatey en utilisant PowerShell, et cela en contournant la politique d'exécution pour éviter toutes les restrictions de sécurité.

Pour vérifier que Chocolatey a bien été installé il faut faire la commande suivante :

Choco

C’est tout pour l’élément 1.

# Elément 2

(Partie mise en page) Pour l’élément 2 j’ai voulu faire un script « principale » qui permet de faciliter l’installation des logiciels via un menu.

Dans ce menu il y a des options qui permettent par exemple de mettre à jour tous ces logiciels ou d’installer une pré configuration.

Pour commencer à faire le menu, j’ai dû utiliser la commande Write-Host pour noter les différentes options et l’opérateur de commande -ForegroundColor pour mettre le texte en couleur. (Partie mise en page)

**Exemple :**  
  


(Partie technique) Une fois la mise en page terminé il faut faire les options qu’il y a dans le menu.

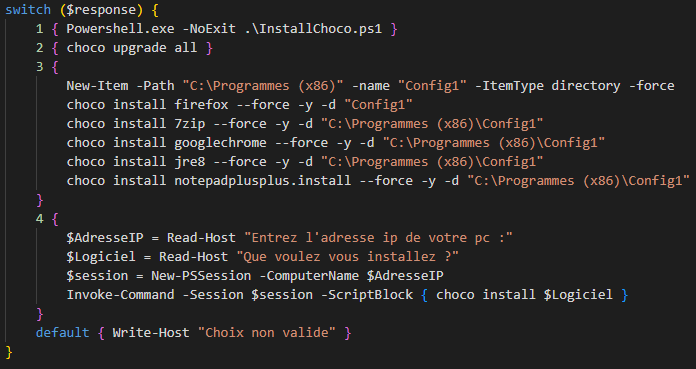
Pour ça j’ai dû crée une base de code qui est comme ça :

$réponse = Read-Host "Entrez le numéro de l’option"

Switch ($reponse) {  
1 {}  
2 {}  
3 {}  
default {Write-Host "Choix non valide"}  
}

Dans la base de code ci-dessus je mets dans 1 {} ce qui doit se passer quand je rentre le chiffre 1 et c’est pareil pour le 2 {} 3 {} et s’il faut que j’ajoute une option 4 il faut mettre ceci 4 {} à la suite des numéros (1,2,3,4). (Partie technique)

**Voici à quoi ressemble le scripte « menu » avec les options :**



# Elément 3

Dans cet élément je vais parler de la vérification du mode Administrateur.  
  
J’ai fait un code qui affiche un popup qui met que l’utilisateur n’a pas lancé le script en mode administrateur, et quand il clic sur « ok » ça le fait quitter le lancement du script.

Ce popup permet de lancer toutes les options de mon script correctement sans bug.

**Le script se présente de la manière suivante :**

if (-NOT ([Security.Principal.WindowsPrincipal][Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuiltInRole] "Administrator"))

{

$wsh = New-Object -ComObject WScript.Shell

$result = $wsh.Popup("Vous devez exécuter ce script en tant qu'administrateur. Appuyez sur OK pour fermer ce script.",0,"Attention",0x1)

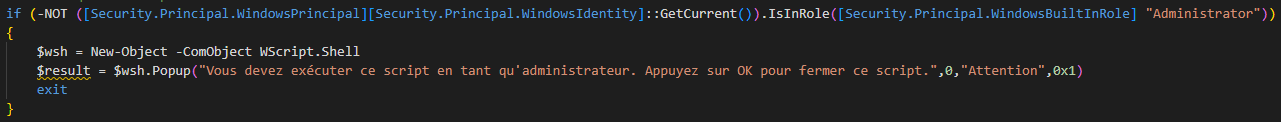
exit

}

La partie rouge permet de vérifier si l'utilisateur courant exécutant le script PowerShell a des privilèges d'administrateur. S'il n'en a pas, il renvoie une erreur indiquant que les privilèges d'administrateur sont nécessaires pour exécuter le script.

Pour la partie bleu le script crée une fenêtre qui demande à l'utilisateur d'exécuter le script en tant qu'administrateur. Si l'utilisateur appuie sur « OK », le script se ferme.

Je précise que se code se met au début du script PowerShell.

**Capture d’écran du script PowerShell** **:**   
  


# Conclusion

En résumé, le projet présenté dans ce rapport vise à développer un script PowerShell qui automatise l'installation de logiciels en utilisant l'outil Chocolatey.

Ce script comprend 4 sous-scripts qui permettent une utilisation plus large, comme un menu d'options, des demandes d'exécution et des demandes pour passer en mode administrateur.

Le but final est d'avoir un script facile à utiliser et à comprendre pour faciliter l'installation de logiciels desktop.

Je pourrais améliorer dans ce script le 4ième sous-script qui permet d’installer un logiciel a distance sur un autre pc.

# Cahier des charges

**Les objectifs du script :**

*Le script doit permettre d’installer ou de mettre à jour des applications facilement avec chocolatey.   
  
 Le script doit permettre de faire du déploiement d’application sur plusieurs pcs.  
  
L’application doit aussi contenir des « preset » d’installation pour pouvoir installer d’un coup un nombre de logiciel très rapidement.*

**Le type de script :**

*Le script va servir à aider les utilisateurs à installer leurs logiciels en peu de temps donc je dirais que c’est plutôt un type (utilitaire).*

**Compatibilité :**

*Le script vise à être compatible pour tous les systèmes Windows et Linux,  
 et doit pouvoir se lancer sans faire d’erreur.*

**Périmètre du projet :**

*Le script vise à être en Français dans un premier temps car c’est une langue que je connais bien, dans une autre mise à jour (s’il y en a une) il faudrait envisager de mettre une autre langue.*

**Contraintes techniques :**

*La seule contrainte c’est qu’il faut PowerShell d’installer et obligatoirement avoir soit un Linux ou soit un Windows pour pouvoir faire fonctionner le programme.*

**Le planning :**

*Le planning est très simple c’est toute les semaines le mardi donc ça fait par journée 8h00, pour travailler sur le projet.*

# Source

ChatGPT: <https://chat.openai.com/chat>

Google : <https://www.google.ch/?hl=rm>

Sapien : <https://www.sapien.com/forums/viewforum.php?f=18>

YouTube : <https://www.youtube.com/>

Chocolatey : <https://community.chocolatey.org/>