# Paquetage d'applications

Lors du déploiement d'une application dans un parc informatique. Il est généralement préférable de préparer nos propres paquets d'installation pour contrôler la version déployée des logiciels ainsi que de pouvoir personnaliser leur configuration automatiquement suite à leur installation (*ex. page d'accueil d'un navigateur web, association d'extensions de fichier à l'application, activation d'une licence, etc.*).

Lors de la préparation de ces installations, il est bonne pratique de conserver une machine virtuelle avec la même configuration que les postes de travail de notre parc informatique avec un snapshot pour rapidement retourner à l'état de la machine précédant l'installation et tester l'installation avec de nouveaux paramètres.

# Processus conceptuel

## Préparation des fichiers

Il est généralement bonne pratique de s'assurer que tous les fichiers requis à l'installation de l'application et de ses pré-requis soient disponibles sans nécessiter une connexion à Internet. Ceci évitera des téléchargements inutiles ainsi que des tentatives de téléchargement de liens désuets ou erronés.

Regrouper et structurer ces fichiers dans un dossier avec un script d'installation facilitera la préparation d'un script en permettant l'utilisation de chemins relatifs (*emplacement des fichiers par rapport au script*) plutôt que l'utilisation de chemins absolus (*chemin complet incluant une lettre de lecteur*). Ceci permettra de déplacer le paquet d'installation (*ex. sur un lecteur réseau, un périphérique USB, etc.*) sans affecter son fonctionnement.

## Préparation d'un script d'installation

### Vérification des pré-requis

Avant de déclencher l'installation de l'application à déployer, il est impératif de s'assurer que le système soit compatible et que tous les pré-requis soient installés. À titre d'exemple, si une version des librairies Visual C++ redistribuables de Microsoft sont requises au fonctionnement d'un logiciel, celles-ci ne seront généralement pas incluses et leur installation manuelle au préalable sera requis. Un autre exemple de validation à effectuer avant l'installation d'un logiciel pourrait être de s'assurer qu'il ne soit pas déjà installé ou encore que la version à déployer du logiciel corresponde à l'architecture du processeur. Une autre validation qui devra être effectuée dans certains cas spécifiques pourrait être la quantité de mémoire vive installée sur un poste de travail.

### Installation silencieuse

Pour le bien-être des utilisateurs, il est préférable d'installer le logiciel de façon silencieuse (sans interaction avec l'interface de l'utilisateur) et sans impacter leur travail. S'il s'agit d'une mise à jour, il est donc préférable d'attendre que le logiciel soit en cours d'utilisation par l'utilisateur du poste de travail. Rares sont les applications qui ne peuvent être installées avec des arguments à la ligne de commande. Advenant un cas d'une application ne le permettant pas, il est possible de capturer l'ensemble des modifications que le programme d'installation applique au système et répliquer ces modifications ou de contrôler la fenêtre d'installation par macros à l'aide de logiciels tels que AutoHotKey ou AutoIT.

### Détection de l'installation

Suite à l'installation, il sera important de s'assurer que l'application s'est correctement installée avant de personnaliser sa configuration ainsi que de s'assurer, s'il s'agit d'une mise à jour, de ne pas écraser les paramètres que l'utilisateur aurait pu configurer. Si l'installation de l'application échoue sur certains systèmes, il sera important d'intervenir plutôt que de seulement fermer une demande d'installation de logiciel par un utilisateur sans s'assurer que la tâche se soit correctement réalisée.

### Journalisation de l'installation

Particulièrement lors de la préparation du paquet d'installation, il sera important de journaliser le processus d'installation pour être en mesure d'apporter les correctifs nécessaires en cas d'échec d'installation et de configuration.

# Exemples concrets

Voici quelques exemples concerts desquels vous pouvez vous inspirer. Sachez que toutes les applications déployables se configurent et se déploient différemment.

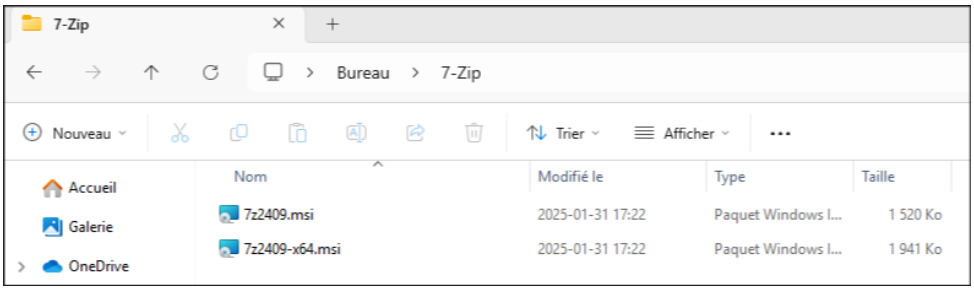
## 7-Zip

7-Zip est une des application parmis les plus simples à paqueter. Son logiciel d'installation est disponible sous forme de ".msi" qui est le format standardisé de déploiement d'application pour Windows. Étant un utilitaire pour la gestion d'archives compressées, il est bonne pratique de configurer l'association des extensions de fichiers d'archive une fois le logiciel installé. Quoi que sa configuration soit simple, il sera nécessaire d'ajuster le registre Windows pour appliquer cette configuration.

### Préparation des fichiers

7-Zip ne dépend d'aucun autre logiciel et nécessite peu de ressources. La seule validation à effectuer dans un cas comme celui-ci serait de déployer la version pour la bonne architecture de processeur. Commençons par regrouper les fichiers requis pour l'installation. Préparons nous à l'éventualité d'une machine x86 ainsi qu'une machine amd64/x64. Un fichier d'installation pour ARM est aussi disponible mais est seulement disponible en format ".exe". Gardons les choses simples pour l'instant.

Les fichiers d'installation peuvent être téléchargés ici.

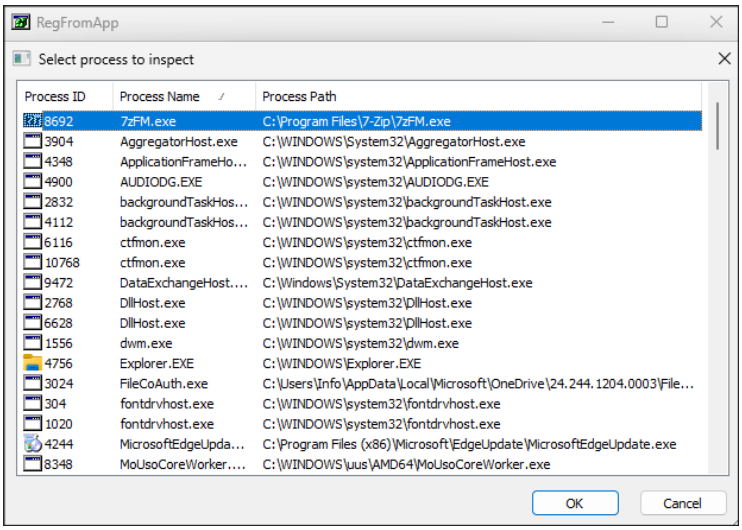



Sachant à l'avance que les modifications qui devront être apportées au système seront dans le registre. Installons l'application manuellement, configurons l'association des fichiers et préparons notre fichier de modification du registre par la même occasion. L'application "RegFromApp" de NirSoft est un bel outil pour détecter toute modification apportée au registre par une application. Elle permet de créer automatiquement un fichier de modification de registre en fonction de ce qui sera détecté.

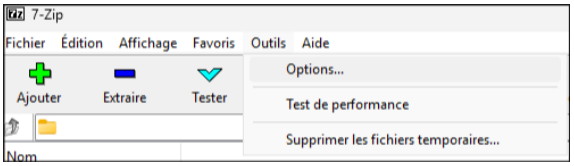
Cette application peut être téléchargée ici.

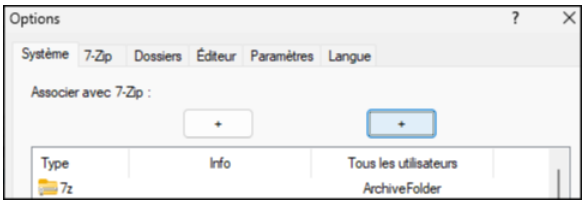
Exécutons 7-Zip ainsi que RegFromApp en tant qu'administrateur, associons les fichiers d'archive au logiciel et enregistrons les modifications qui seront appliquées au registre à l'air de RegFromApp.

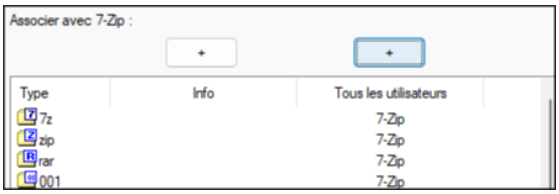
Sélection du processus 7-Zip



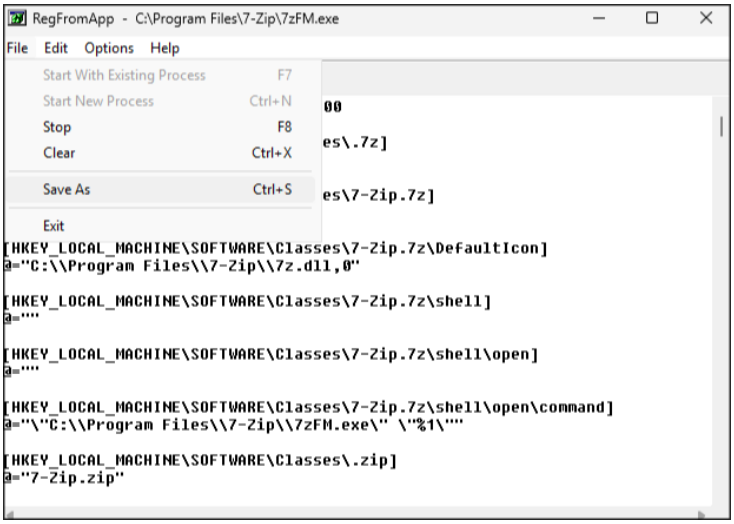

Configuration des associations de types de fichier



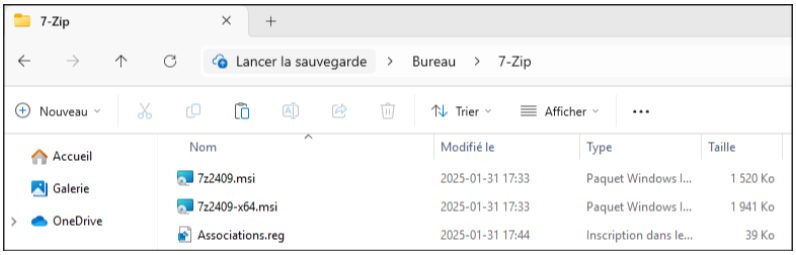



Cliquez sur "appliquer" et toutes les modifications devraient apparaître dans RegFromApp. Enregistrez ces modifications dans un fichier dans le même dossier que les fichiers d'installation MSI.




Nous devrions maintenant avoir tout ce qui sera nécessaire pour procéder à l'installation du logiciel.




Prenez note de l'emplacement de l'exécutable installé pour détecter son installation plus tard ( *C:\Program Files\7-zip\7zFM.exe dans ce cas*). Prenez soin de sauvegarder ce dossier à l'extérieur de votre machine virtuelle et restaurez son "snapshot" ou cliché instantané de son état précédant l'installation manuelle du logiciel et préparons maintenant l'installation automatisée.

## Préparation du script d'installation

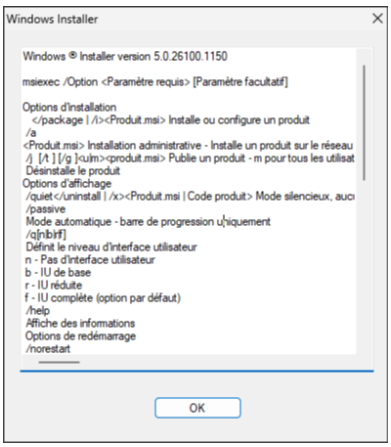
Puisque cette installation est simple et que le seul paramètre ajusté est l'association des fichiers, on ne se souciera pas ici de vérifier si une version précédente était installée ou si des paramètres

d'interface utilisateur étaient définis. On s'assurera uniquement d'installer la version correspondant à l'architecture du processeur de la machine.

Créons un script powershell (*setup.ps1*) au même endroit que les fichiers d'installation et éditons le script dans Powershell ISE exécuté en tant qu'administrateur. Il faudra tout d'abord localiser l'emplacement des fichiers d'installation. La variable "$PSScriptRoot" correspond à l'emplacement du script et peut être utilisé comme point de référence pour localiser le reste des fichiers. Débutont le script en sélectionnant le dossier du paquet d'installation comme dossier de travail avec la commande "Push-Location"

```
# Sélection du dossier de travail
Push-Location $PSScriptRoot
```

Puisque notre fichier d'installation est un ".msi", l'exécutable "msiexec.exe" situé dans le dossier "System32" et ses paramètres standardisés nous permetterons d'effectuer l'installation de façon automatisée silencieusement. Il est possible d'afficher l'ensemble des options de l'exécutable avec la commande "msiexec /?". Assurons-nous d'installer la version correspondant à l'architecture du processeur et démarrons l'installation.




Les options qui nous intéressent ici sont /i pour lancer l'installation /q[option d'affichage] et /l [fichier de journal]. Démarrons l'installation et laissons le script attendre la fin de l'installation du logiciel avec la commande "Start-Process" et son argument "-Wait".

```
# Déterminer le fichier d'installation à utiliser en fonction de l'architecture du processeur
switch ($env:PROCESSOR_ARCHITECTURE) {
  "AMD64" {$setupfile = "7z2409-x64.msi"}
  "x86" {$setupfile = "7z2409.msi"}
}

# Démarrage de l'installation si le nom de fichier est défini
if ($setupfile) {
  Start-Process "msiexec" -ArgumentList "/i $setupfile /qn /log install.log" -Wait
}
```

Assurons-nous maintenant que l'installation s'est bien déroulée et procédons à la configuration du système. Si aucun des fichiers d'installation correspondait à l'architecture du processeur, l'installation ne se sera pas déroulée et par conséquent, les modifications à apporter ne seront pas appliquées. On peut ici procéder avec un "if-else" mais si on utilise plutôt un "try-catch", il sera possible d'exécuter le code d'échec si l'échec se produit à l'importe quelle étape de la configuration.

```
# Début de la configuration
try {

  # Vérification de l'existence du fichier de l'application
  Get-Item "C:\Program Files\7-zip\7zFM.exe"

  # Importation des modifications au registre
  Start-Process "reg" -ArgumentList "import Associations.reg" -Wait

  # Information de l'état de l'installation
  Write-Host "Installation de 7-zip réussie"
  Sleep 5

} catch {

  # Information de l'état de l'installation
  Write-Host "Échec d'installation de 7-zip"
  Sleep 5

}
```

Suite à l'exécution de ces blocs de script, 7-Zip sera installé et assigné comme application par défaut pour la gestion de fichier d'archives.

Pour faciliter l'exécution de ce script en tant qu'administrateur sans nécessiter le changement de politique d'exécution de script et d'exécuter le script à partir d'un invite de commandes avec permissions élevées, il est possible à l'aide d'un script "batch" (*extension .bat*) comme le suivant de lancer l'exécution de tout ceci en tant qu'administrateur.

```
@echo off
cd /D %~dp0
powershell -c "Set-ExecutionPolicy Unrestricted -Force"
powershell -noexit "& " ".\setup.ps1"
exit
```

Le script complet ainsi que les fichiers d'installation sont disponibles ici.

## Google Chrome

Google Chrome est une application qui par défaut s'installe par utilisateur. Une version entreprise est disponible gratuitement ici pour déployer l'application à la largeur d'un système. Cette version est aussi accompagnée d'extensions de domaine ADMX permettant la configuration du navigateur par politique de groupe. Il est aussi possible de consulter le résultat de ces politiques de groupe pour appliquer les configurations manuellement dans le registre Windows advenant qu'une machine ne soit pas jointe à un domaine. Cette ressource permet de consulter l'ensemble des GPOs et leur résultat sur le registre.

Certains paramètres initiaux peuvent être déployés par un fichier JSON initial_preferences en le copiant dans le dossier d'installation de Chrome (ex. C:\Program Files\Chrome\Application\initial_preferences). Ce fichier est disponible dans le "bundle" téléchargeable de Chrome Entreprise mais peut être téléchargé ici.

Préparons le paquet d'installation pour tous les jeux d'instructions de processeur supportés (AMD64, x86, ARM64) avant son installation. Assurons-nous aussi que l'ordinateur soit doté d'au moins 8Gb de mémoire vive.

```
## Définir des variables de configuration

# Mises à jour automatiques
$EnableUpdates = 0

# La variable suivante peut être changée pour HKCU sans "Recommended" pour appliquer les paramètres seulement à l'utilisateur actuel ou Ã  HKLM sans "Recommended" pour enforcer ces paramÃ¨tres Ã  tous.
$RecommendedPath = "HKLM:\SOFTWARE\Policies\Google\Chrome\Recommended"
$LocalMachinePath = "HKLM:\SOFTWARE\Policies\Google\Chrome"
$DefaultUserPath = "HKU:\.DEFAULT\SOFTWARE\Policies\Google\Chrome"
$CurrentUserPath = "HKCU:\SOFTWARE\Policies\Google\Chrome"

# Configuration du navigateur
#$HomeURL = "https://theonion.com"
$BookmarksBar = 1
$ImportOnFirstStart = 0
$SyncDisabled = 1
$DisableHistory = 1
$PasswordManager = 0
$HideStoreIcon = 1
```

```
## Se délacer dans le dossier du script
Push-Location $PSScriptRoot

## Déterminer le fichier d'installation à utiliser en fonction de l'architecture du processeur
switch ($env:PROCESSOR_ARCHITECTURE) {
  "AMD64" {$SetupFile = "GoogleChromeStandaloneEnterprise-AMD64.msi"}
  "x86" {$SetupFile = "GoogleChromeStandaloneEnterprise-x86.msi"}
  "IA64" {$SetupFile = "GoogleChromeStandaloneEnterprise-IA64.msi"}
}

## Déterminer la quantité de mémoire vive
if ((Get-ComputerInfo).OsTotalVisibleMemorySize -gt 8000000) {
  # Démarrage de l'installation
  Start-Process "msiexec" -ArgumentList "/i $SetupFile /qn /log install.log" -Wait
} else {
  Write-Host "Mémoire vive insuffisante"
  Read-Host "Appuyez sur entrée pour quitter"
  exit
}

try {
    ## Détecter l'installation par code de produit et version MSI
    $MSICode = (Get-AppLockerFileInformation -Path $SetupFile | Select-Object -ExpandProperty
Publisher).BinaryName
    $MSIVersion = (Get-AppLockerFileInformation -Path $SetupFile | Select-Object -
ExpandProperty Publisher).BinaryVersion

    # La verison du MSI ne correspond pas à la version de Chrome mais seulement à la version
du paquet. On peut donc seulement se fier à la clé MSI.
    Test-Path "HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\$MSICode"

    ## Configuration de l'installation

    # Création forcée de la clé des paramètres recommandés pour la remplacer
    New-Item -Path $RecommendedPath -Force

    # Configuration des mises à jour automatiques
    New-ItemProperty -Path "HKLM:\Software\Policies\Google\Update" -Name "Install{8A69D345-
D564-463C-AFF1-A69D9E530F96}" -PropertyType "DWord" -Value $EnableUpdates -Force
```

```
    # Comportement au démarrage (fonctionne seulement si la machine est jointe à un domaine)
    if ($HomeURL) {
        New-ItemProperty -Path $LocalMachinePath -Name "RestoreOnStartup" -Value 4 -
PropertyType "DWord" -Force
        New-ItemProperty -Path $LocalMachinePath -Name "ShowHomeButton" -Value 1 -PropertyType
"DWord" -Force
        New-ItemProperty -Path $RecommendedPath -Name "HomepageLocation" -Value $HomeURL -
Force
        New-Item -Path "$LocalMachinePath\RestoreOnStartupURLs" -Force
        New-ItemProperty -Path "$LocalMachinePath\RestoreOnStartupURLs" -Name "1" -Value
$HomeURL -Force
    }

    # Importations au premier démarrage
    New-ItemProperty -Path $RecommendedPath -Name "ImportAutofillFormData" -Value
$ImportOnFirstStart -PropertyType "DWord" -Force
    New-ItemProperty -Path $RecommendedPath -Name "ImportBookmarks" -Value $ImportOnFirstStart
-PropertyType "DWord" -Force
    New-ItemProperty -Path $RecommendedPath -Name "ImportHistory" -Value $ImportOnFirstStart -
PropertyType "DWord" -Force
    New-ItemProperty -Path $RecommendedPath -Name "ImportSavedPasswords" -Value
$ImportOnFirstStart -PropertyType "DWord" -Force
    New-ItemProperty -Path $RecommendedPath -Name "ImportSearchEngine" -Value
$ImportOnFirstStart -PropertyType "DWord" -Force
    New-ItemProperty -Path $RecommendedPath -Name "ImportSearchEngine" -Value
$ImportOnFirstStart -PropertyType "DWord" -Force

    # Gestionnaire MDP
    New-ItemProperty -Path $RecommendedPath -Name "PasswordManagerEnabled" -Value
$PasswordManager -PropertyType "DWord" -Force

    # Favoris et barre de favoris
    New-ItemProperty -Path $RecommendedPath -Name "BookmarkBarEnabled" -Value
$BookmarksBarEnabled -PropertyType "DWord" -Force

    # Masquer l'icône du magasin
    New-ItemProperty -Path $LocalMachinePath -Name "HideWebStoreIcon" -Value $HideStoreIcon -
PropertyType "DWord" -Force
```

```
    # Synchronisation cloud
    New-ItemProperty -Path "HKLM:\SOFTWARE\Policies\Google\Chrome" -Name "SyncDisabled" -Value
$SyncDisabled -PropertyType "DWord" -Force

    # Sauvegarde d'historique
    New-ItemProperty -Path "HKLM:\SOFTWARE\Policies\Google\Chrome" -Name
"SavingBrowserHistoryDisabled" -Value $DisableHistory -PropertyType "DWord" -Force

} catch {
    Write-Host "Erreur d'installation"
    Read-Host
}
```

Cet exemple de paquet peut être téléchargé ici.

---

Revision #11
Created 2025-01-31 21:40:57 UTC by Alexandre Arsenault-Jetté
Updated 2025-02-05 03:30:39 UTC by Alexandre Arsenault-Jetté