Projet SaveYourDatas

122 – Documentation de projet

Kolly Florian

Projet du 6.09.2016 au 28.09.2016

Table des matières

1 Introduction 4

2 Structogramme 5

3 Conception 6

3.1 Choix du code 6

3.1.1 Quel moyen pour détecter le téléphone 6

3.1.2 Utilisation de ADB 6

3.1.3 ADB avec PowerShell 6

3.1.4 Ecrire le résultat dans un fichier texte 6

3.1.5 Interface 6

3.1.5.1 Choix des checkbox 6

3.1.5.2 Choix de l’affichage de l’historique 7

3.1.5.3 Choix du bouton aide 7

3.2 Tests technologiques 7

3.2.1 Les possibilités d’ADB 7

3.2.2 Lancer ADB depuis PowerShell 7

3.2.3 Ecrire dans le log Windows 7

3.2.4 Créer une tâche planifiée 7

3.2.5 Gestion des processus 7

4 Réalisation 8

4.1 Fonctionnalités du script 8

4.1.1 Fonctionnalité : transfert 8

4.1.1.1 Installation automatique d’ADB 8

4.1.1.2 Trouver et copier tous les fichiers d’un type spécifique 8

4.1.2 Fonctionnalité : Choix des transferts 8

4.1.2.1 Sélection par des Check Box 8

4.1.2.2 Détecter quels médias envoyés 9

4.1.3 Bouton d’aide 9

4.2 Fonctionnalités obligatoires 9

4.2.1 Ecriture : Fichier personnalisé 9

4.2.1.1 Création ou contrôle de l’existence du fichier 9

4.2.1.2 Ecriture dans le fichier 10

4.2.2 Lecture : Fichier personnalisé 10

4.2.3 Ecriture : Windows Log 10

4.2.3.1 Création d’un nouveau journal si nécessaire 10

4.2.3.2 Ecriture dans le log 10

4.2.4 Lecture : Windows Log 10

4.2.5 Gestion : Processus 11

4.2.5.1 Arrêter les processus s’ils existent déjà 11

4.2.6 Gestion : Tâches planifiées 11

4.2.6.1 Tâches planifiées : Création 11

4.2.6.2 Tâches planifiées : Suppression 11

4.2.6.3 Gestion : Case cochée au lancement si planification 11

4.2.6.4 Gestion : Administrateur pour tâches planifiées 11

4.2.7 Interface 12

4.2.8 Fonction : général 12

5 Tests 13

5.1 WhiteBox 13

5.2 BlackBox 13

5.3 Environnementaux 14

6 Problèmes et solutions 15

6.1 ADB ne sauvegarde que la carte SD | résolu 15

6.2 Le chemin des fichiers à copier n’existe pas | résolu 15

6.3 2x MsgBox création de tâches non administrateur | résolu 15

6.4 Pas de système de détection du téléphone 15

7 Conclusion 16

# Introduction

Voici la documentation de projet contenant les éléments de conception, réalisation, tests et problèmes et solutions. Il permet d’avoir un suivi sur le travail réalisé et un aperçu de ce que peut et va faire le script. La partie réalisation contient des bouts des codes ainsi que leur explication, tandis que la partie conception contient les tests avant le commencement et l’aperçu des possibilités et de la réalisation du script.

# Structogramme



# Conception

## Choix du code

### Quel moyen pour détecter le téléphone

|  |  |  |  |
| --- | --- | --- | --- |
| Moyen | Description | Problème(s) | Utilisation |
| Win32\_USBControllerDevice | Permet de détecter le téléphone directement sans complément | Le téléphone est de base connectée comme MTP, qui ne possède pas de lettre de lecteur dans Windows, donc il n’est pas possible de copier des fichiers | Non |
| FTP sur le téléphone | Permet de connecter le téléphone par réseau en créant un serveur sur le téléphone | Ne respecte pas le cahier des charges | Non |
| ADB | Permet de « virtualiser » le téléphone afin d’y accèder facilement | Commandes inconnues, apprentissage supplémentaire, installation doit être automatique | Oui |

### Utilisation de ADB

ADB (Android Debug Bridge) permet d’accéder à de nouvelles commandes en lien avec le téléphone. Il est cependant nécessaire que le téléphone soit en administrateur et permettent à ADB d’y accéder. Les commandes ADB peuvent être lancées depuis PowerShell.

### ADB avec PowerShell

Pour permettre l’utilisation d’ADB avec PowerShell, il suffit de Set-Location dans le fichier ou se trouve l’exécutable ADB et de lancer les commandes en utilisant ADB.

### Ecrire le résultat dans un fichier texte

|  |  |  |
| --- | --- | --- |
| Moyen | Description | Utilisation |
| Opérateur >> | Comme pour batch | Oui, puisque je n’ajoute pas beaucoup de données |
| Out-file | Permet d’écrire un tableau | Non, pas de tableau |
| Export-csv | Exporte en CSV pour Excel | Non, pas de tableau |
| .Net StreamWriter | Ecrit dans un fichier texte en le lisant | Non, bien que plus rapide, pas nécessaire pour ces données |

### Interface

#### Choix des checkbox

* Plus intuitif pour l’utilisateur
* Permet le choix de plusieurs Check Box
* Facile à détecter dans le code

#### Choix de l’affichage de l’historique

* Permet de savoir rapidement quels fichiers ont été transférés
* Permet de savoir le résultat du transfert pour chaque fichier

#### Choix du bouton aide

Puisque le script nécessite des compétences avancées pour le paramétrage nécessaire du téléphone avant le lancement du script, l’ajout d’un bouton d’aide permet aux utilisateurs de découvrir l’ensemble des actions à réaliser avant de pouvoir exécuter le script. Il s’agit d’un aperçu, nommant les actions à réaliser. L’utilisateur peu averti devra faire ses propres recherches.

## Tests technologiques

### Les possibilités d’ADB

ADB permet de bien gérer les transferts entre le téléphone et l’ordinateur. Il va donc me permettre de réaliser les transferts correctement. Il est possible de transférer du PC vers le téléphone, et vice-versa. Le point fort d’ADB est de pouvoir lancer des commandes Linux via le Shell intégré. (adb shell)

### Lancer ADB depuis PowerShell

Pour lancer ADB depuis PowerShell, il faut indiquer le chemin où se trouve ADB et Set-Location sur ce chemin. Je peux ensuite utiliser ADB en faisant .\adb.exe suivi des commandes souhaitées.

### Ecrire dans le log Windows

Il existe des CmdLets spécifiquement créés pour pouvoir écrire dans le log de Windows. Elles commencent par un verbe et sont suivies de « EventLog ». L’utilisation de ces dernières va donc permettre de contrôler l’existence d’un planificateur, et d’en créer un si nécessaire, avant d’écrire dans le log.

### Créer une tâche planifiée

Pour les tâches planifiées, il existe aussi des CmdLets spécifiques. Ils contiennent ScheduledTask dans leur nom. Attention tout de fois à contrôler si la tâche existe déjà avant de la créer.

### Gestion des processus

Afin de réaliser la gestion des processus, il est nécessaire de connaître la variable $PID. Cette dernière permet d’obtenir l’ID du processus du script lié. Il suffit donc de contrôler s’il existe déjà et si oui, l’arrêter.

# Réalisation

## Fonctionnalités du script

### Fonctionnalité : transfert

#### Installation automatique d’ADB

Pour l’installation automatique d’ADB, il faut (dans l’ordre)

* Chercher et télécharger depuis Internet

$source = <http://adbshell.com/upload/adb.zip>  
$destination = "C:\users\admin\Desktop\adb2\adb.zip"  
Invoke-WebRequest $source -OutFile $destination

* Déziper le fichier

$shell = new-object -com shell.application  
$zip = $shell.NameSpace(“C:\users\admin\Desktop\adb2\adb.zip”)  
foreach($item in $zip.items())  
{  
 $shell.Namespace(“C:\users\admin\Desktop\adb2”).copyhere($item)  
}

* Supprimer le zip

Remove-Item C:\Users\admin\Desktop\adb2\adb.zip

#### Trouver et copier tous les fichiers d’un type spécifique

Afin de trouver et copier tous les fichiers d’un certain type, il faut créer une boucle foreach qui traverse la liste récursive où l’objet contient l’extension souhaitée. Il suffit ensuite de copier les fichiers dans la boucle.

foreach ($file in .\adb.exe shell find | Where-Object {$\_ -match…\*

\*cf 6.2

##### Copie des fichiers

Pour la copie des fichiers, on utilise la fonctionnalité pull offerte par ADB en mettant comme source la variable $file contenant tout le chemin du fichier.

.\adb.exe pull $file C:\Users\admin\Desktop\médias\...

### Fonctionnalité : Choix des transferts

#### Sélection par des Check Box

* Lancement des fonctions de transfert si les CheckBox sont cochées

#### Détecter quels médias envoyés

Afin de ne pas lancer le script si ce n’est pas nécessaire, on contrôle avant tout si au moins une case est cochée. Si ce n’est pas le cas, on demande é l’utilisateur de le faire. Ensuite, on regarde pour chaque case et lance la fonction voulue si nécessaire. Finalement, on indique à l’utilisateur que le script a fini les transferts.

if($ChckMusics.Checked.Equals($True) -or $ChckImages.Checked.Equals($True) -or $ChckVideos.Checked.Equals($True)){  
 chargementADB  
 if($ChckMusics.Checked.Equals($True)){  
 transfertMusics  
 }  
 if($ChckImages.Checked.Equals($True)){  
 transfertImages  
 }  
 if($ChckVideos.Checked.Equals($True)){  
 transfertVideos  
 }  
 [System.Windows.Forms.MessageBox]::Show("Les transferts souhaités ont été réalisés!")  
 }else{  
 [System.Windows.Forms.MessageBox]::Show("Veuillez choisir une catégorie à transférer!")  
}

### Bouton d’aide

* Affiche une msgBox décrivant les étapes à réaliser

[System.Windows.Forms.MessageBox]::Show("Pour que ce script fonctionne correctement, …")

## Fonctionnalités obligatoires

### Ecriture : Fichier personnalisé

#### Création ou contrôle de l’existence du fichier

* Si le fichier existe, OK
* Si le fichier n’existe pas, crée-le

if(Test-Path C:\Users\admin\Desktop\test.txt){  
 "Le fichier existe déjà"  
}else{  
 New-Item C:\Users\admin\Desktop\test.txt -ItemType File  
}

* Crée aussi le fichier si le bouton historique est cliqué, afin d’éviter les bugs si le fichier n’existe pas

if(!(Test-Path C:\Users\admin\Desktop\log.txt)){  
 New-Item C:\Users\admin\Desktop\log.txt -ItemType File  
}

#### Ecriture dans le fichier

* Ecrit dans le fichier le nom de $file et la date, tout cela à la ligne (`n).

"`n$dateCourante : Le transfert de $file a été fait" >> C:\Users\admin\Desktop\test.txt

### Lecture : Fichier personnalisé

La lecture du fichier personnalisé se fait à travers l’historique de l’interface. Celui-ci va lire ligne par ligne les informations et va s’arrêter à la fin. La partie de lecture du code est tirée du code par Roman Kuzmin trouvé sur StackOverFlow.

$reader = [System.IO.File]::OpenText("C:\users\admin\DeskTop\log.txt")  
try {  
 for() {  
 $line = $reader.ReadLine()  
 if ($line -eq $null -and !($line -contains ($dateJMA))){  
 break  
 }  
 # process the line  
 $txtHistorique.AppendText($line)  
 $txtHistorique.AppendText("`r")  
 }  
}  
 finally {  
 $reader.Close()  
}

### Ecriture : Windows Log

#### Création d’un nouveau journal si nécessaire

$nomScript est une variable déclarée plus haut dans le code. Il s’agit en quelque sorte d’une constante.

$nomScript = "Backup phone"

* Si le log n’existe pas, le créer

$logExiste = Get-EventLog -LogName Application | Where {$\_.Source -eq $nomScript}   
if (! $logExiste) {  
 New-EventLog -LogName Application -Source $nomScript  
}

#### Ecriture dans le log

* Ecrire dans le log à condition que le journal soit créé ou qu’il soit déjà existant

Write-EventLog -LogName Application -Source $nomScript -EntryType Information -EventId 1 -Message "L'application a bien lancé la sauvegarde pour les $origine"

### Lecture : Windows Log

* Permet de voir si le planificateur existe déjà

$logExiste = Get-EventLog -LogName Application | Where {$\_.Source -eq $nomScript}   
if (! $logExiste) {  
 …  
}

### Gestion : Processus

#### Arrêter les processus s’ils existent déjà

Get-Process | where {$\_.Name -eq "adb"} | Stop-Process  
foreach($processus in Get-Process | where {$\_.Name -eq "powershell" -and $\_.Id -ne $PID}){  
 Stop-Process $processus  
}

* Lancer la fonction lors du chargement de l’interface

### Gestion : Tâches planifiées

#### Tâches planifiées : Création

* Si le planificateur n’existe pas, alors le créer

$tache = New-ScheduledTaskAction -Execute 'C:\Users\admin\Desktop\projet.ps1'  
$dateLancement = New-ScheduledTaskTrigger -Weekly -DaysOfWeek Saturday -At 9:30am  
$planificateur = Get-ScheduledTask | select TaskName | ? {$\_.TaskName -eq $nomScript}  
  
#Si le planificateur n'existe pas  
if($planificateur -eq $null){  
 Write-Host $planificateur  
 Register-ScheduledTask -Action $tache -Trigger $dateLancement -TaskName $nomScript -Description "Sauvegarde automatique des médias du téléphone"  
}

#### Tâches planifiées : Suppression

* Confirm :$false permet d’empêcher l’apparition d’une fenêtre de confirmation

Unregister-ScheduledTask -TaskName $nomScript -Confirm:$false

#### Gestion : Case cochée au lancement si planification

* Permet d’automatiquement cocher la case s’il y a planification dès le lancement

if(Get-ScheduledTask | select TaskName | ? {$\_.TaskName -eq $nomScript}){  
 $ChckTaches.Checked = $True  
}

#### Gestion : Administrateur pour tâches planifiées

* Force à lancer en administrateur pour pouvoir créer ou supprimer des tâches planifiées

if (-NOT  
([Security.Principal.WindowsPrincipal][Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuiltInRole] "Administrator")){   
 [System.Windows.Forms.MessageBox]::Show("Pour utiliser cette fonctionnalité, veuillez lancer ce script en administrateur!")  
 $ChckTaches.Checked = $false  
 …  
}

### Interface

L’interface est générée par PrimalForms. Elle contient cependant des handlers permettant de détecter une action.

En voici la liste :

|  |  |  |
| --- | --- | --- |
| handler | Quand ? | Quoi ? |
| $handler\_btnAide\_Click | Clique sur le bouton d’aide | Affiche une msgBox |
| $handler\_ChckTache\_Click | Clique sur la case de création de tâches planifiées | Crée ou supprime la tâche planifiée |
| $handler\_saveYourDatas\_Load | Au chargement de l’interface | Gère les processus et coche la case tâches si nécessaire |
| $handler\_btnHistorique\_Click | Clique sur le bouton historique | Affiche l’historique |
| $handler\_btnTransfert\_Click | Clique sur le bouton transfert | Lance les transferts souhaités |

### Fonction : général

Voici la liste des fonctions créées, avec leur but simplifié et où elles sont utilisées :

|  |  |  |
| --- | --- | --- |
| Fonction | Action | Utilisation |
| installationADB | Installe automatiquement ADB | chargementADB (fonction) |
| chargementADB | Charge ADB et change la location | $handler\_btnTransfert\_Click |
| creerTachePlanifiee | Crée une tâche planifiée | $handler\_ChckTache\_Click |
| supprimerTachePlanifiee | Supprime une tâche planifiée | $handler\_ChckTache\_Click |
| gestionProcessus | Gère les processus | $handler\_saveYourDatas\_Load |
| ecrireResultat | Ecrit dans le fichier personnalisé | transfertMusics, transfertImages et transfertVideos (fonctions) |
| ecrireWindowsLog | Ecrit dans le Windows Log | transfertMusics, transfertImages et transfertVideos (fonctions) |
| transfertMusics | Transfère les musiques | $handler\_btnTransfert\_Click |
| transfertImages | Transfère les images | $handler\_btnTransfert\_Click |
| transfertVideos | Transfère les vidéos | $handler\_btnTransfert\_Click |

# Tests

## WhiteBox

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| N° | Nom | Description | Résultat souhaité | Résultat obtenu | Erreur |
| 1 | Les fichiers sont tous détectés par la boucle foreach | Tous les fichiers d’un certain type doivent être détectés par la boucle foreach | OK | OK | N’arrive pas à détecter certain fichier dans des dossiers spéciaux |
| 2 | Les processus sont correctement gérés | Si l’utilisateur lance deux fois le script, le premier doit être arrêté | OK | OK | - |
| 3 | Le planificateur du Windows Log est créé s’il n’existe pas encore | Le planificateur qui va gérer l’ajout de ligne dans le Windows Log doit être créé, sinon il y a une erreur | OK | OK | - |

## BlackBox

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| N° | Nom | Description | Résultat souhaité | Résultat obtenu | Erreur |
| 1 | Transfert les médias sélectionnés | Le script doit pouvoir faire transférer chaque type de média séparément, puis tous ensemble sans erreur | OK | OK | Erreur liée à ADB s’affiche. Cette dernière ne pose pas de problème. |
| 2 | Affiche msgBox si pas de sélection | Si aucun type de média n’est sélectionné, une msgBox doit apparaître et le programme ne doit pas se lancer | OK | OK | - |
| 3 | Ajoute tâche planifiées si clique sur le bouton | Lors du clic sur le bouton de planification, une tâche planifiée doit être créée | OK | OK | - |
| 4 | Affiche aide si clique sur le bouton | Lors du clic sur le bouton d’aide, une msgBox doit s’afficher contenant quelques lignes d’aide | OK | OK | - |

## Environnementaux

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| N° | Nom | Description | Résultat souhaité | Résultat obtenu | Erreur |
| 1 | Windows 10 | Le script doit fonctionner sous Windows 10 | OK | OK | - |
| 2 | Windows 8 | Le script doit fonctionner sous Windows 8 | OK | KO | ADB ne peut pas détecter de device. |
| 3 | Windows 7 | Le script doit fonctionner sous Windows 7 | Pas testé | Pas testé | Pas testé |

# Problèmes et solutions

Les problèmes sont décrits dans l’ordre d’apparition. Certains problèmes peuvent donc être sujet à des fonctionnalités, partie de code ou élément non implémenté ou modifié.

## ADB ne sauvegarde que la carte SD | résolu

Lorsque je réalise un pull sur le stockage en général, ADB ne me permet que de sauvegarder la carte SD, et non le stockage en général (mémoire interne). En utilisant une boucle foreach récursive (-R), le problème est réglé.

## Le chemin des fichiers à copier n’existe pas | résolu

Lors de la copie des fichiers, le chemin des fichiers n’est pas trouvé par PowerShell, ce qui empêche la copie des fichiers sur le PC. Pour résoudre le problème, il faut modifier la boucle (cf. 5.1.2.2) en mettant find au lieu de ls –R.

## 2x MsgBox création de tâches non administrateur | résolu

Si le script n’est pas lancé en admin, il ne faut pas pouvoir créer ou supprimer les tâches planifiées, car cela créerait une exception. Pour cela, j’ai instauré un système qui demande à l’utilisateur de lancer en administrateur et qui ne modifie pas les tâches. Le système modifie alors la valeur de la case à false. A ce moment, puisque la case va être modifiée deux fois, l’handler contrôlant s’il y changement va se lancer deux fois et donc afficher deux fois le message. Pour éviter cela, au lieu de mettre un handler détectant un changement, il faut mettre un handler détectant un clic.

## Pas de système de détection du téléphone

Puisque qu’un téléphone ne possède pas de lettre de lecteur, je n’ai pas trouver le moyen de contrôler si un téléphone est connecté ou non, ce qui peut renvoyer à une erreur ADB : device not found.

# Conclusion

Le script fonctionne avec les fonctionnalités obligatoires, mais le manque de temps m’a empêché de réaliser toutes les fonctionnalités souhaitées pour le script. La détection des fichiers pose encore des problèmes, mais le transfert fonctionne correctement. L’interface est plus fonctionnelle que belle, mais elle contient les outils nécessaires pour la compréhension.