Automatiser les tâches d'administration - Scripting pour Windows

Introduction

Nous avons vu, à travers l'utilisation du mode commande de Windows et particulièrement à travers la construction de fichiers Batch, l'intérêt de pouvoir faire appel à des lots de commandes dans le cadre de **l'automatisation de tâches répétitives**.

De nouveaux langages de script sont apparus dans les dernières versions de Windows (la notion de 'scripting' est apparu très tôt dans le monde Unix/Linux) et apportent des fonctionnalités beaucoup puissantes que leurs prédécesseurs.

Un script est un fichier texte (comme un lot de commandes .BAT) mais qui ne va plus contenir de simples commandes, mais un véritable programme écrit dans un langage de script. Un certain nombre d'outils et bibliothèques vont être associés au langage afin de couvrir l'administration de l'ensemble d'un poste de travail ou d'un serveur.

L'objectif de ce support est :

- de présenter les différents outils associés au *scripting* sous Windows
- de donner quelques exemples de scripts permettant d'aborder les possibilité de chacun des outils
- de lister quelques ressources permettant d'être plus efficace dans l'écriture de scripts

Table des matières

1. Les outils

VBScript
C'est le langage de scripting dérivé de Visual Basic. Contrairement aux langages de programmation d'applications qui nécessitent une phase de compilation avant exécution, les langages de scripting sont interprétés ; un interpréteur est un programme qui va lire chaque ligne du programme source et l'exécuter. Il est interprété par le moteur de script VBscript.

Windows Script Host (WSH)
C'est l'outil qui va être capable de prendre en charge l'exécution d'un script exprimé dans un des langages de scripts (VBScript, JScript, etc.). WSH détermine d'abord le langage utilisé, puis tente d'exécuter le script pour vérifier les erreurs de syntaxe (il appelle le moteur de script en fonction du langage), et enfin exécute le script jusqu'à la fin ou une éventuelle erreur.
WSH propose deux modes d'exécution :

- Wscript : en mode fenêtré (Windows) ;
- Cscript : en mode console.

WSH propose également un certain nombre d'objets qui permettent d'étendre les fonctionnalités de VBScript. Un fichier texte .WSF peut lancer l'exécution de plusieurs scripts .VBS ou autres.

Script Runtime
Librairie (scrrun.dll) permettant de manipuler le système de fichiers ; propose 3 objets manipulables en scripting

- FileSystem Object : gérer les objets du système de fichiers
- Dictionary Object : gestion de données sous forme de dictionnaire
- Script Encoder Object : chiffrage des scripts

Windows Management Instrumentation (WMI)
WMI présente une vaste bibliothèque permettant l'accès aux composants matériels et logiciels d'un ordinateur, ainsi qu'aux fonctions d'administrations, gestion des fichiers journaux, réseaux et de gestion des performances (toute ressource gérable est décrite dans une classe).
WMI est l'implémentation Microsoft de WBEM(Web-Based Enterprise Management), une norme de partage de ressources au sein d'un réseau et de CIM(Common Information Model), un modèle de représentation des objets d'un système et de la manière d'obtenir des informations de ses composants.

L'accès à WMI peut être réalisé en ligne de commandes (WMIC) ou en mode console (WMIMNGT.MSC).

La création de scripts utilisant les classes d'objets de WMI (WMI Scripting Library) est riche et des outils (par exemple Scriptomatic Tool, éditeur VBSEdit) permettent d'en simplifier l'écriture.

Attention : les classes disponibles peuvent varier en fonction des versions de systèmes d'exploitation Windows (XP, XP PRO, 2000, 2003, sercices packs)

Active Directory Service Interface (ADSI)
ADSI est une API fondée sur COM (Component Object Model, modèle de composants serveurs qui fournissent des services aux applications) va permettre d'agir sur l'annuaire Active Directory et de gérer les différents objets de l'AD : utilisateurs, ordinateurs, groupes, etc.

2. Les instructions du langage VBScript

Dans l'exemple qui suit, seules sont utilisés des éléments standard du langage VBScript

- **Déclaration de constantes et variables** : CONST, DIM
- **Fonctions de chaînes :** UCASE, RIGHT, concaténation avec &, retour à la ligne en utilisant les codes ASCII 10 et 13 (CR et LF)
- **Fonctions de dates :** DATE(), DAY, MONTH, YEAR
- **Structures de contrôles :** boucle déterminée : FOR NEXT – boucle non déterminée : DO WHILE LOOP et DO LOOP WHILE – test simple : IF THEN ELSE END IF – test multiple : SELECT CASE END SELECT
- **Gestion des erreurs d'exécution :** ON ERROR RESUME NEXT, objet ERR
- **Procédures et fonctions :** SUB, FUNCTION

Cf. exemple01.vbs : syntaxe de base de VBscript.
Lancement en invite de commande :
Soit :

```
> Wscript exemple01.vbs
```

Soit :

```
> Cscript exemple01.vbs
```

3. L'objet WSCRIPT de WSH (1)

Dans l'exemple qui suit, sont ajoutées des objets offerts par WSH – WSCRIPT (objet défini dans le contexte d'exécution, donc pas besoin de le créer) :

- **Affichage de messages en fonction du contexte (sortie standard)** : WSCRIPT.ECHO
- **Paramètres passés au script :** WSCRIPT.ARGUMENTS, WSCRIPT.ARGUMENTS.NAMED, etc.
- **Quitter un script :** WSCRIPT.QUIT
- **Marquer une pause dans l'exécution :** WSCRIPT.SLEEP
- **Informations générales relatives à l'exécution du script :** WSCRIPT.NAME, WSCRIPT.PATH, WSCRIPT.SCRIPTNAME, etc.

Cf. exemple02.vbs : récupération des paramètres
Lancement en invite de commande, sans paramètre :
Soit :

```
> Wscript exemple02.vbs
```

Soit :

```
> Cscript exemple02.vbs
```

Lancement en invite de commande, avec des paramètres :
Soit :

```
> Wscript exemple02.vbs /nom1:par1 par2 par3 /nom4:par4 par5
```

Soit :

```
> Cscript exemple02.vbs /nom1:par1 par2 par3 /nom4:par4 par5
```

4. Les objets instanciés de WSH (2)

Dans l'exemple qui suit, sont ajoutées des classes d'objets offertes par WSH – WSCRIPT qu'il est nécessaire s'instancier avant de pouvoir utiliser les propriétés et méthodes des objets.
On trouve les classes suivantes :

- Wscript.Shell : donne accès à l'invite de commande
- Wscript.Network : permet d'avoir accès aux paramètres du réseau local
- Wscript.Controller : permet le contrôle de l'exécution de scripts distants

- **Utilisation de l'objet Shell** :
  - Utiliser les commandes du mode commande : méthode Run
  - Exécuter le contenue d'une variable : méthode Exec
  - Consigner des messages dans les journaux d'évènement : méthode LogEvent
  - Accéder aux variables systèmes : méthode ExpandEnvironmentString, propriété Environment associée à Process, User, System
  - Accéder à la base de registre : méthode RegRead, RegWrite, RegDelete
  - Boites de dialogue : méthode Popup
  - Envoyer des frappes clavier au système : méthodes, Run, AppActivate, SendKeys
- **Utilisation de l'objet NetWork :**
  - Paramètres de connexion : propriétés UserName, UserDomain, ComputerName
  - Gérer les mappages de lecteurs réseau : méthodes MapNetWorkDrive, RemoveNetWorkDrive, EnumNetworkDrives
  - Gérer les mappages d'imprimantes réseau : méthodes AddPrinterConnection, SetDefaultPrinter, RemovePrinterConnection, EnumPrinterConnections
- **Utilisation de l'objet Controller :**
  - Exécuter un script sur une machine distante : méthodes Execute
-

Cf. exemple03.vbs : objet Network
Cf. exemple04.vbs : objet Shell

5. Runtime

Le composant Runtime propose essentiellement l'accès au système de fichiers à travers 2 classes d'objets :

- Scripting.FileSystemObject : donne accès aux objets du système de fichiers : disques, repertoires, fichiers
- Scripting.Dictionary : donne accès à un système de gestion temporaire d'informations sous la forme d'un dictionnaire (association couple clef-valeur)

- **Utilisation de l'objet FileSystemObject**:
  - Informations sur les dossiers : méthode GetFolder, FolderExists, objet Folder, propriétés Path, Name, ShortPath, ShortName
  - Informations sur les fichiers : méthode GetFile, FileExists, objet File, propriétés FileVersion, Type, Attributes

    - Gérer des dossiers : méthode CreateFolder, CopyFolder, MoveFolder, DeleteFolder, objet Folder
    - Gérer des fichiers : méthode CopyFile, MoveFile, DeleteFile, objet File
- **Utilisation de l'objet Dictionary**
    - Gérer temporairement une liste d'informations (tableau associatif) : méthodes Add, Remove, RemoveAll, Exists, Count, Items, Keys

Cf. exemple05.vbs : liste des lecteurs

6. WMI

VMI propose plusieurs centaines de classes d'objets (en fonction du système d'exploitation) permettant d'interroger les composants logiciels et matériels d'un ordinateur et ainsi d'offrir des capacités d'administration importantes.
La séquence générale permettant l'utilisation d'une classe d'objets est la suivante :

- Récupération d'une connexion au service à partir de winmgmts:\\pc1 (:protocole:\\machine)
- A partir de ce service, instanciation des classes d'objets disponibles ou requête de filtrage sur des données (des journaux par exemple, ou des propriétés des objets)

Et parmi les classes, on trouve par exemple :

- **Win32_service**
    - Connaître la liste des services disponibles et leurs propriétés
- **Win32_BIOS**
    - Retourner des informations sur le BIOS
- **Win32_BootConfiguration**
    - Retourner des informations sur le démarrage du système

Etc. cf. scripts produits par l'outil Scriptomatic

Cf. exemple06.vbs : extraite des données sur les journaux système

7. ADSI

La gestion de l'Active Directory (AD) grâce à ADSI va consister en :

- la création d'objet en lien avec le protocole LDAP, puis en l'utilisation de méthodes de type Put, Setinfo, Delete, etc.
- la recherche d'objets dans l'AD en utilisant une connexion ADO (ActiveX Data Object) et OLE DB (Object Linking and Embeding DataBases, accès uniforme à tout type de données) sur l'annuaire et en spécifiant des critères de sélection :
    - base de recherche : chemin LDAP pour démarrer la recherche dans l'arborescence
    - types d'objets recherchés (group, user, compiter, OrganisationalUnit)
    - attributs à retourner
    - champs de recherche (onelevel, subtree)

Une console ADSI est fournie sur les SE Windows 2000 et 2003 (administration des serveurs)

8 Fichiers WSF : jobs et packages

Les fichiers WSF offrent la possibilité de définir des fichiers de commandes sous forme d'un document XML où le code VBS (ou JS ou autre) est complété par des balises qui précisent sa description, en particulier une définition plus précises des paramètres.

```
<package>
<job id="TestArguments">
  <runtime>
    <named name="p1" helpstring="premier parametre" required="true" />
    <named name="p2" helpstring="premier parametre" required="true" />
    <unnamed name="fichier" helpstring="nom des fichiers" many="true"
required="true" />
    <description>Exemple de fichier WSF</description>
    <example>macommande.wsf /p1:val1 /p2:val2 fic1 fic2</example>
  </runtime>
  <resource id="msgok">Tout s'est bien deroulé</resource>
  <script language="VBscript">
Option Explicit
If wscript.arguments.count = 0 then
     Wscript.arguments.showusage
     Wscript.quit
End if
With wscript.arguments.named
     If .exists("p1") then wscript.echo "valeur arg1 = " & .item("p1")
     If .exists("p2") then wscript.echo "valeur arg2 = " & .item("p2")
End with

Dim arg
For each arg in wscript.arguments.unnamed
     Wscript.echo "non nommé : " & arg
next
Wscript.echo getResource ("msgok")
  </script>
</job>
</package>
```

Figure 1 : exemple d'exécution sans arguments

La notion de package permet de regrouper au sein d'un même fichier plusieurs 'job' et de lancer spécifiquement l'un d'entre eux en précisant sont nom en paramètre :
`Macommande.wsf //Job:TestArguments /p1:toto /p2:tata fichA fichB fichC`
(dans l'exemple proposé, la notion de package est optionnelle : ici un seul job est défini dans le fichier).

9. Evolution prévue avec PowerShell

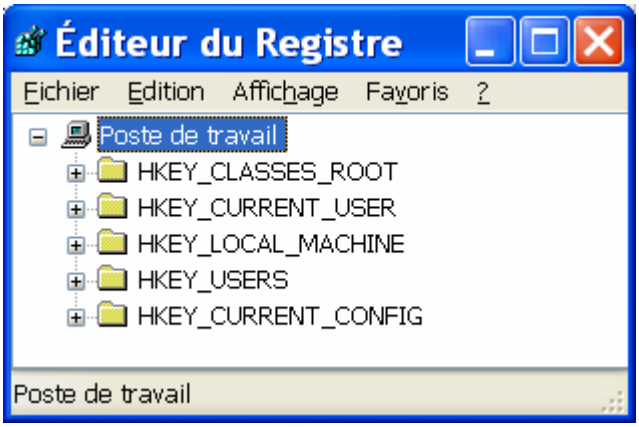
Dans sa nouvelle version de système d'exploitation Vista, nom de code Longhorn, Microsoft va proposer un nouveau shell (interpréteur de commandes), PowerShell, qui va se rapprocher des shells Unix/Linux (shell, Bash, etc.).

Annexe A – Base de registres

La base de registre (ou registre Windows) est une base de données utilisée par le système d'exploitation Windows et les applications.
Elle contient les données de configuration du système d'exploitation et des autres logiciels installés.
Ces informations sont stockées sous une forme hiérarchique, avec des « feuilles » terminales qui contiennent les valeurs des clefs de registre (à lire ou modifier ou créer).

Il est possible d'accéder à la base de registre grâce à la commande REGEDIT, et d'y modifier des valeurs **EN CONNAISSANCE DE CAUSE**.
ATTENTION AUX MANIPULATIONS MALENCONTREUSES SUR LA BASE DE REGISTRE.

La structure de la base de registre est hiérarchique et on trouve au premier niveau les entrées suivantes :



- HKEY_CLASSES_ROOT : contient les informations relatives aux applications enregistrées (extensions de fichiers et applications liées, etc.)
- HKEY_LOCAL_MACHINE : contient les informations générales à tous les utilisateurs (matériel, sécurité, logiciels, système)
- HKEY_USERS : contient des informations relatives à chaque utilisateur (l'utilisateur courant
- HKEY_CURRENT_USER : sous-ensemble de HKEY_USERS qui contient des informations relatives à l'utilisateur courant
- HKEY_CURRENT_CONFIG : sous ensemble de HKEY_LOCAL_MACHINE qui contient des informations de configuration logiciel et système de base

L'ancêtre de la base de registre correspond aux fichiers .INI des systèmes Windows précédents.

Annexe B – Active Directory

C'est le service d'annuaire des nouveaux systèmes Windows (les systèmes Unix/Linux peuvent utiliser un service similaire : annuaires LDAP, Lightweight Directory Access Protocol);

Il centralise le recensement et la gestion des objets du système : utilisateurs, ordinateurs, sécurité, etc.

Le protocole LDAP permet l'interrogation des annuaires.

Un exemple de hiérarchie dans l'annuaire LDAP :

- DNS du domaine lycee.npdc.fr (domaine component : dc=lycee, dc=npdc, dc=fr)
  - Arras (organizational unit : ou=arras)
    - GuyMollet (ou=gyumollet)
      - Utilisateurs (ou=utilisateurs)
        - Fred (objet, common name : cn=fred)

Adresse de Fred dans l'annuaire LDAP (son identifiant unique, DN, Distinguished Name) :
LDAP://cn=fred, ou=utilisateurs, ou=guymollet, ou=arras, dc=lycee, dc=npdc, dc=fr

Active Directory intègre un système de sécurité : l'utilisateur se logue une fois, et un processus de d'authentification Kerberos est effectué en arrière-plan, à chaque demande d'accès aux ressources.

Annexe C – 'Regular expressions', expressions rationnelles ou expressions régulières

La recherche d'informations textuelles dans des « bases de données » nécessitent l'utilisation de critères de sélection puissant : les comparaisons basiques (critères d'égalité, contient, débute par et se termine par) ne sont souvent plus suffisantes.

La notion de 'Regular Expression, RegExp' correspond à une notation qui utilise une combinaison de caractères spéciaux, dont les caractères jokers, pour effectuer une comparaison d'une chaîne de caractère avec un modèle ou motif (anglais pattern).
L'utilisation des caractères jokers '*' et ' ?' en ligne de commande est un exemple de motif très simple.

Par exemple :

- ''gr[oia]s'' → va permettre de trouver les correspondances avec gros, gris, gras
- ''([\w|\-]+)@([\w|\-]+)\.(\w+)'' → permet de vérifier qu'une chaîne correspond au modèle d'adresse email : caractère spécial w = lettres ou chiffres, ou bien caractère -, plusieurs fois (au moins une), puis @, puis idem, puis ., puis lettres ou chiffres.