

TP3 : Windows et son langage de commande

1 Objectifs : Familiarisation avec Windows

L'objectif de ce premier TP sera :

1. D'observer certaines caractéristiques du Système d'Exploitation (SE) Windows et notamment la solution implémentée pour gérer la configuration du système.
2. De se familiariser avec l'interpréteur et le langage de commande (hérité de MS-DOS) qui permet d'interagir avec le SE par le langage de commande.
3. De réaliser un premier fichier de commande Batch.

2 Installation et démarrage de la machine virtuelle Windows 7

2.1 - Installation

A l'aide du gestionnaire de fichiers Ubuntu, placez-vous sur le répertoire `/media/localdata` ; si un répertoire nommé **Windows 7 Pro - 32 bits** est présent, passez à l'étape démarrage de la machine virtuelle Windows, sinon depuis le disque commun (racine) copier le fichier **Windows 7 Pro-32 bits.zip** dans le répertoire `/media/localdata`, puis décompressez-le dans ce répertoire. Une fois l'opération de décompression terminée vous pouvez lancer la machine virtuelle Windows.

2.2 – Démarrage de la machine virtuelle Windows

Lancer la machine virtuelle Windows par un double clic sur le fichier **Windows 7 Pro-x86.vmx** ; une fenêtre apparaît vous demandant si vous avez déplacé ou copié la machine virtuelle, cliquez sur « I copied it ». Ne pas installer les add-ons VMWare. Si c'est le premier démarrage de ma machine virtuelle, Windows installe du matériel et il faut redémarrer Windows.

3 La base de registre

3.1 La base de registre

Les données requises à Windows sont stockées dans une base de données appelée **Base de Registre** (BDR). La BRD centralise les informations nécessaires au fonctionnement du système et des programmes. Une altération de la BDR peut entraîner le dysfonctionnement d'une application ou du système. Le contenu du registre est stocké dans des fichiers systèmes et peut-être édité et modifié par l'intermédiaire du logiciel **regedit**. La BDR est organisée en clés et sous-clés hiérarchisées. Chaque clé ou sous clé peut contenir des valeurs caractérisées par un nom, un type parmi ceux prédéfinis et la donnée proprement dite.

1) Afficher et observer la base de registre

a. Dans le menu Démarrer → Programme → Exécuter, tapez la commande **regedit**.



HKEY_LOCAL_MACHINE (HKLM) et HKEY_USERS (HKU) sont les deux branches principales de la BDR. Les autres branches sont des liens vers les sous-répertoires de ces deux clés.

b. **HKLM** est stocké dans `C:\WINDOWS\System32\Config`. Affichez le contenu de ce répertoire et déterminer les fichiers associés aux différentes sous-clés de HKLM.

c. Dans le menu démarrer → rechercher, faites une recherche des fichiers `Ntuser.dat` qui correspondent aux clés `HKEY_USERS`. Où sont situés ces fichiers ?

d. Recherchez la clé **ComputerName** (menu Edition → rechercher). Que contient-elle ?

La base de registre est utilisée par la plupart des applications Windows notamment pour la gestion des configurations. Ainsi, l'installation ou la suppression d'une application engendre souvent la modification de la BDR. Il existe des outils permettant de « nettoyer » la BDR et de supprimer les entrées inutilisées. La base de registre peut aussi être modifiée manuellement dans la limite des droits de l'utilisateur.

2) Modifier la Base de Registre

Modification par une application :

Lancez Notepad (Démarrer → Programme → Accessoire → Bloc-Notes) et changez la police de caractères (menu format) puis refermez Notepad. Ensuite, affichez la clé **ifFaceName** utilisée par Notepad : **HKEY_CURRENT_USER\Software\Microsoft\Notepad**

Que contient-elle ? Sans quitter regedit, ouvrez à nouveau Notepad, modifiez la police dans le menu format. **Fermez Notepad** et dans **regedit** appuyez sur la touche **F5** (rafraichissement des valeurs). Que vaut **ifFaceName** ?

Modification manuelle donnez à la clé **ifFaceName** la valeur **Courier** et vérifiez que cela a été pris en compte dans Notepad.

3.2 Les variables d'environnement

Une variable d'environnement est une variable globale du système pouvant être lue ou modifiées par une application ou un programme. C'est un moyen simple de centraliser une information et de configurer le système. Par exemple, la variable TMP va contenir le répertoire des fichiers temporaires, la variable COMPUTERNAME, le nom de l'ordinateur, la variable USERNAME, le nom de l'utilisateur courant... Notons l'importance de la variable PATH. Elle comprend la liste des répertoires contenant les programmes que le système peut exécuter. Les variables d'environnement sont localisées dans la BDR.

3) Localiser les variables d'environnement dans la BDR

Etudiez le contenu des clés : **HKEY_CURRENT_USER\Environment**
et **HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Environment**.

4) Modifier les variables d'environnement

a. Affichez la fenêtre d'édition des variables d'environnement : clic droit sur « **Ordinateur** » (menu Démarrer) → « **Propriétés** » → « **Paramètres systèmes avancés** » → « **Variables d'environnement** »

La partie supérieure de la fenêtre correspond aux variables utilisateur et la partie inférieure, aux variables système que vous ne pouvez pas modifier sauf si vous êtes administrateur de la machine.

b. Ajoutez la variable d'environnement utilisateur **PATH** en lui associant le répertoire **C:\TP_SYS\BATCH** (i = votre n° de groupe de TP).

c. Dans l'explorateur Windows, créez un répertoire **C:\TP_SYS** contenant un sous-répertoire **BATCH**.

Ainsi, vous avez ajouté votre répertoire **C:\TP_SYS\BATCH** au **PATH** déjà défini. De cette manière, le système pourra exécuter les programmes enregistrés dans ce répertoire.

4 De l'interface graphique au langage de commande

Toutes les opérations réalisées par le biais de l'interface graphique (lancement d'application, gestion de l'arborescence de fichiers, modification de fichiers etc...) peuvent se faire en ligne de commande. Ainsi, au lieu de cliquer sur « Explorateur de fichiers » pour afficher l'arborescence du disque, on va utiliser la commande **dir** dans la fenêtre de commandes.

5) Ouvrir une fenêtre de commande

Démarrer → Exécuter → **cmd**

ou

Démarrer → Tous les programmes → Accessoires → Invite de Commandes

Quel est le répertoire courant ?

6) Afficher et modifier les variables d'environnement

- Affichez la liste des variables d'environnement avec la commande **set**
- Tapez **echo TMP**, puis **echo %TMP%**. Quelle est la différence entre les deux syntaxes ? Que fait la commande **echo** ?
- Affichez le contenu de la variable **PATH** et vérifiez que votre répertoire **C:\TP_SYS\BATCH** est bien dans le **PATH**.

7) Gérer l'arborescence des fichiers

- Tapez **C:** pour vous positionner sur le disque C, puis éventuellement **cd ** pour vous positionner à la racine du disque **C**. Affichez le contenu du répertoire **C:\WINDOWS\System32\Config** avec la commande **dir**. Restreignez l'affichage aux fichiers cachés en utilisant la bonne option de la commande **dir** (**help dir** pour avoir la liste des options de la commande).
- Positionnez vous sous votre répertoire **C:\TP_SYS** à l'aide de la commande **cd**.
- Créez un sous-répertoire **TMP** au même niveau que **BATCH** avec la commande **mkdir**. Affichez l'arborescence du répertoire **TP_SYS** avec la commande **tree**.

Pour la suite du TP, votre répertoire courant doit être **C:\TP_SYS**

8) Ecrire et lire dans un fichier

- Affichez le contenu du répertoire avec la commande **DIR**. Tapez la commande **help** puis: **help > commandes.txt**. De nouveau, affichez le contenu du répertoire. Que constatez-vous ?
- Affichez le contenu du fichier **commandes.txt** en utilisant la commande **type** ou **more** et rappelez-vous le rôle du symbole **>** (voir le cours).
- Tapez la commande **echo Usage du prompt : > tmp\aide_prompt.txt**.
Que fait cette commande ?
- Ajoutez l'aide de la commande prompt à la fin du fichier **aide_prompt.txt**
Rappel : pour ajouter du texte à un fichier utilisez **>>** (voir le cours).
- Vérifiez le résultat en tapant **more Tmp\aide_prompt.txt**

9) Copier / déplacer / supprimer des fichiers

Rappel : les jokers

? : remplace un seul caractère du nom ou de l'extension d'un fichier.

Exemple : ?ale.txt désigne tous les fichiers comme cale.txt dale.txt hale.txt etc.

* : remplace un ensemble de caractères

Exemple : b*.txt regroupe tous les fichiers dont le nom commence par b et dont l'extension est txt comme ba.txt ballon.txt boule.txt etc.

Toutes les opérations suivantes sont à faire à l'aide de commandes textuelles (shell Windows) et non avec l'explorateur de fichiers.

- Déplacez le fichier **commande.txt** vers le répertoire **TMP** (**move**)
- Dans **TMP**, dupliquez le fichier **aide_prompt.txt** sous le nom **aidep.txt** (**copy**)
- Copiez tous les fichiers de suffixe **.txt** contenus dans **TMP** dans le répertoire **BATCH**
- Détruisez tous les fichiers du répertoire **TMP** (**del**) et du répertoire **BATCH**.

5 Automatiser les traitements : les fichiers Batch

Script Batch

Les scripts Batch sont des programmes qui permettent d'automatiser certaines tâches répétitives telles que l'administration ou les sauvegardes de fichiers. Ce sont des fichiers texte avec l'extension .bat ou .cmd. Ils contiennent les mêmes commandes que celles exécutées en ligne de commande (une commande par ligne). Ils peuvent être paramétrés et organisés à l'aide des instructions if, goto et des étiquettes.

Exécuter un script Batch

Pour exécuter un script Batch, il suffit de taper le nom de ce fichier dans la fenêtre de commandes ; la séquence de commandes qu'il contient est alors exécutée par l'interpréteur du langage de commande. L'interprète exécute les commandes de manière séquentielle, dans l'ordre où elles apparaissent dans le fichier. Il n'est pas nécessaire de taper l'extension .bat du fichier.

Les scripts permettent d'automatiser un certain nombre de traitements. Ce sont des outils puissants et indispensables à tout utilisateur avancé.

10) Mon premier Batch :

Lancez l'éditeur de texte **Notepad**. Créez un fichier nommé **Batch1.bat**, dans le répertoire **C:\TP_SYSi**. Dans ce fichier écrivez les lignes de commandes permettant d'afficher :

```
Nous sommes le : date
Il est : heure
Le répertoire du système Windows est : nom_du_répertoire
Il contient les fichiers système :
liste du contenu
```

liste du contenu correspond à la liste des fichiers du répertoire système de Windows qui ont l'attribut S (Système). Le répertoire système de Windows est défini dans la variable **windir**

Rappel : la date est disponible dans la variable **date**, et l'heure dans la variable **time**

Rappel : La valeur d'une variable **v** s'écrit **%v%**

❗ *Attention, les accents sont codés différemment selon que vous êtes sous Windows (Notepad) ou dans l'interprète de commande. Le plus simple est de ne pas afficher de message comprenant des caractères accentués.*

Enregistrez votre fichier. Dans la fenêtre de commande, positionnez-vous sous **C:** et tapez **Batch1.bat** pour l'exécuter. Pourquoi ça ne fonctionne pas ? Enregistrez votre fichier dans le répertoire **C:\TP_SYSi\BATCH** et retapez votre commande dans la console. Pourquoi la commande fonctionne-t-elle cette fois ?

Modifiez votre **Batch1.bat** en ajoutant **echo off** au début. Que se passe-t-il ? Remplacez **echo off** par **@echo off**. Quel changement cela induit-il ?

11) Mon second Batch : Tester l'existence de fichier et/ou de répertoire

- Créer un fichier de commandes **testFic.bat** qui contient l'exemple ci-contre ; il s'agit d'une commande testant l'existence d'un fichier ou d'un répertoire
- Affichez l'aide de la commande. Testez cette commande pour un fichier et un répertoire existants/non existants.
- Que représente le **%1** ?
- Que fait **if exist %1\NUL** ?

```
@echo off
if "%1"==" " goto usage
if "%1"=="?" goto usage
if exist %1\NUL (
    echo le repertoire %1 existe !
) else (
    if exist %1 (
        echo le fichier %1 est présent
    ) else echo %1 est absent
)
goto fin
:usage
echo usage : testFic nomfichier
:fin
```

ERRORLEVEL : Toutes les commandes renvoient une valeur de retour pour indiquer leur succès (0) ou leur échec (≥ 1). Cette valeur est accessible par une condition de la forme **ERRORLEVEL n** qui signifie « le code renvoyé par la commande précédente est $\geq n$ ». Pour tester si une commande s'est bien déroulée, on fait le test suivant :

```
IF ERRORLEVEL 1 goto erreur
```

Cette instruction a pour effet de se brancher à l'étiquette **erreur** si la commande précédente a renvoyé un code ≥ 1 .

- Modifiez votre script en utilisant la commande **dir /ad**
Cette commande échoue si le fichier spécifié n'est pas un répertoire ; elle permet donc de distinguer un fichier d'un répertoire.
- Modifiez le script **testFic.bat** de façon à ce qu'il permette de tester l'existence de plusieurs fichiers ou répertoires (nombre indéfini de paramètres) en utilisant la commande **SHIFT** et une itération (voir l'exemple donné en cours pour afficher les paramètres d'un script).
- Ecrire un script **dateDuJour.bat** qui affiche :

Nous sommes le Xeme jour du Yeme mois de l'annee Z.

Pour extraire une sous-chaine d'une variable, on utilise la syntaxe suivante :

%nomvar:~d,lg%

nomvar est une variable d'environnement

d est l'indice de début de la sous-chaine (**0** est l'indice du premier caractère)

lg est la longueur de la sous-chaine

exemple : si la variable **nomvar=tralala**, alors **%nomvar:~1,3%** désigne la chaine "ral"

Lecture de fichiers, analyses de chaînes, calculs (si vous avez le temps)

On dispose d'un fichier nommé **annuaire.txt** contenant une ligne pour chaque utilisateur avec le format suivant :

```
prenom nom:bureau nnn:telephone
```

Exemple de fichier **annuaire.txt** :

```
#annuaire. Format des enregistrements:
#prenom nom:Bureau xxx:tel
gerard manvussat:bureau 118:04.76.54.32.32
ivan skivolle:bureau 116:04.76.54.32.33
vincent tim:bureau 116:04.76.54.32.33
debby scott:bureau 120:04.76.54.32.10
justin ticou:bureau 121:04.76.54.32.11
```

En utilisant la commande **FOR**, on veut écrire la commande **creelogin.bat** qui, à partir d'**annuaire.txt**, crée le fichier **annuaire_login.txt** en rajoutant le login de chaque utilisateur en début de ligne. Le login est formé de la première lettre du prénom et des 7 premières lettres du nom.

Exemple de fichier **annuaire_login.txt** :

```
#annuaire_login. Format des enregistrements:
#login:prenom nom:Bureau xxx:tel
gmanvuss:gerard manvussat:bureau 118:04.76.54.32.32
iskivoll:ivan skivolle:bureau 116:04.76.54.32.33
vtim:vincent tim:bureau 116:04.76.54.32.33
dscott:debby scott:bureau 120:04.76.54.32.10
jtizou:justin ticou:bureau 121:04.76.54.32.11
```

La commande FOR permet d'appliquer une commande ou une suite de commandes

- soit à un (ou plusieurs) fichiers,
- soit à une chaîne de caractères :

```
FOR /F ["options"] %variable IN (ensemble-fichiers) DO commande [paramètres]
FOR /F ["options"] %variable IN ("chaîne") DO commande [paramètres]
ou
FOR /F ["options"] %variable IN (ensemble-fichiers) DO (
commande [paramètres]
...
)
```

ensemble-fichiers est un ou plusieurs noms de fichiers. Chaque fichier est ouvert, lu et traité avant de passer au fichier suivant de ensemble-fichiers. Le traitement consiste à lire dans le fichier, le découper en lignes individuelles de texte puis analyser chaque ligne en zéro ou plusieurs parties. Le corps de la boucle FOR est ensuite appelé avec la ou les valeurs de variables prenant la valeur de la ou des parties trouvées. Par défaut, /F transmet la première partie séparée par un espace dans chaque ligne de chaque fichier. Les lignes vides sont ignorées. Vous pouvez outrepasser le comportement d'analyse par défaut en spécifiant le paramètre optionnel "options". Il s'agit d'une chaîne entre guillemets contenant un ou plusieurs mots-clés spécifiant diverses options d'analyse.

Les mots-clés sont :

eol=c spécifie un caractère de commentaire de fin de ligne (un seul)
skip=n spécifie le nombre de lignes à ignorer en début de fichier.
delims=xxx spécifie un ensemble de caractères délimiteurs. Ceci remplace l'ensemble de délimiteurs par défaut qui sont l'espace et la tabulation.

tokens=x,y,m-n spécifie les parties de chaque ligne devant être transmises au corps de FOR à chaque itération. Ceci causera l'allocation de noms de variables supplémentaires. La forme m-n est une étendue spécifiant les parties allant de m à n.

Par exemple, la commande :

```
FOR /F "eol=; tokens=2,3 delims=, " %i in (monfich.txt) do echo %i %j
```

analyse chaque ligne de monfich.txt, en ignorant les lignes commençant par un point-virgule, en transmettant les 2^{de} et 3^{ème} parties de chaque ligne au corps de FOR, les parties étant délimitées par des virgules et/ou espaces. Notez que le corps de FOR référence %i pour l'obtention de la 2^{de} partie, %j pour l'obtention de la 3^{ème} partie.

Si le fichier monfich.txt contient les lignes suivantes :

```
;exemple de fichier
Bonjour Georges;Dupond,vous allez bien ?
Oui Marcel Durand je vais bien
```

La commande ci-dessus affiche :

```
Georges;Dupont vous
Marcel Durand
```

Lorsque la commande for s'applique à une chaîne et non à un ensemble de fichiers, cette chaîne est traitée de la même manière qu'une ligne de fichier.

exemple :

```
FOR /F "tokens=2,3 delims=, " %i in ("Bonjour Georges;Dupond,vous allez bien ?") do ( echo %i %j )
```

affiche : Georges;Dupont vous

Attention : dans un fichier de commande, les variables de la commande FOR s'écrivent avec %% à la place de %. Ainsi la commande donnée en premier exemple s'écrirait de la manière suivante :

```
FOR /F "eol=; tokens=2,3 delims=, " %%i in (monfich.txt) do @echo %%i %%j
```

Option EnableDelayedExpansion

Avant la commande FOR, il faut placer la commande suivante dans le fichier de commande :

```
setlocal EnableDelayedExpansion
```

Cette option permet à la boucle for de reconnaître qu'une variable d'environnement prend différentes valeurs au cours du for. Sans cette option, si on utilise des variables d'environnement dans la boucle FOR, la variable aura toujours la même valeur car FOR itère en considérant par défaut que la valeur de la variable a été fixée et ne changera pas. La variable dont la valeur change dans l'itération est notée **!variable!** au lieu de **%variable%**

Voici un exemple de réalisation de la commande `creellogin.bat`

```
@echo off
rem on supprime annuaire.txt s'il existe
if exist annuaire_login.txt del annuaire_login.txt
setlocal EnableDelayedExpansion
FOR /f "eol=# tokens=1-5 delims=: " %%a in (annuaire.txt) DO (
set prenom=%%a
set nom=%%b
echo !prenom:~0,1!!nom!:%%a %%b:%%c %%d:%%e >>annuaire_login.txt
)
echo le fichier annuaire_login.txt est cree
```

Modifiez cette commande de sorte que `annuaire.txt` et `annuaire_login.txt` soient des paramètres de la commande `creellogin.bat`