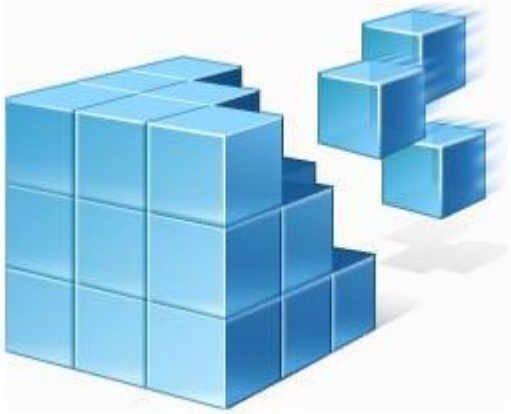
R 1.04

2023 - 2024

Introduction aux systèmes d'exploitation et à leur fonctionnement

TP N°2

Windows et son langage de commande



ANNE Jean-François

Le but de ce TD est de se familiariser avec Windows et sa base de registre.

Windows et son Registre

I. La base de registre Windows

Lire l'article « La base de registre Windows ou le registre Windows » :

- https://www.malekal.com/registre-windows/

Puis le complément :

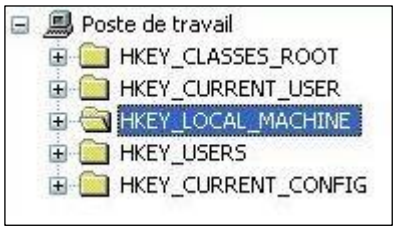
- https://www.zebulon.fr/dossiers/windows/57-base-de-registre.html

II. Premiers contacts avec la Base de Registres

Les données requises au SE Windows sont stockées dans une base de données appelée **Registre** ou **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 d'outils tel que **regedit** et **regedit32.** 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.

Pour les questions suivantes :Si vous n'avez pas les droits d'affichage du répertoire, cherchez les informations sur internet !

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 correspondant aux clés HKU. 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 peut induire 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 :

Affichez la clé **ifFaceName** utilisée par Notepad : HKEY_CURRENT_USER\Software\Microsoft\Notepad

- Que contient-elle ?

Sans quitter regedit,

- Ouvrez le BlocNote (Démarrer -> Programme -> Accessoire -> Bloc-Notes),
- Modifier la police dans le menu format.
- Fermez Notepad

et dans **regedit** appuyez sur la touche **F5.**

- Que vaut **ifFaceName** ?

Modification manuelle

- Donnez à la clé **ifFaceName** la valeur Courier

et vérifiez que cela a été pris en compte dans Notepad.

Fermez Regedit.

3. 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. Depuis Windows 2000, les variables d'environnement sont localisées dans la BDR.

a) 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.

b) Modifier les variables d'environnement

1) Affichez la fenêtre d'édition des variables d'environnement : sous Windows, tapez le mot clé «Variables» dans la barre de recherche.

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

2) Dans l'explorateur Windows, créez un répertoire **TP_SYS** sous votre compte Z:\ contenant un sous répertoire **BATCH**.

3) Modifier la variable d'environnement **PATH** avec la valeur en lui ajoutant le répertoire **Z:\TP_SYS\BATCH**.

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

III. 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 « Poste de travail » pour afficher l'arborescence du disque, on va taper la commande **DIR** dans la fenêtre de commandes.

A. Ouvrir une fenêtre de commande (on parle aussi, d'invite ou encore de prompt) :

Démarrer -> Exécuter -> **cmd** ou

Démarrer -> Invite de Commandes ou

Démarrer -> Tous les programmes -> Accessoires -> Invite de Commandes

Quel est le répertoire courant ?

B. Afficher et modifier les variables d'environnement

a) Affichez la liste des variables d'environnement avec la commande SET

b) Tapez **ECHO TMP.** Puis **ECHO %TMP%.** Quelle est la différence entre les deux syntaxes ? Que fait la commande **ECHO** ?

c) Affichez le contenu de la variable **PATH** et vérifier que votre répertoire **Z:\TP_SYS\BATCH** est bien dans le **PATH.**

C. Gérer l'arborescence des fichiers

a) 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).

b) Positionnez vous sous votre répertoire **Z:\TP_SYS** à l'aide de la commande **CD**.

c) Créez un sous-répertoire **TMP** au même niveau que **BATCH** avec la commande **MD**. Affichez l'arborescence du répertoire **TP_SYS** avec la commande **TREE**.

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

D. Ecrire et lire dans un fichier

a) 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 ?

b) Affichez le contenu du fichier commandes.txt en utilisant la commande **type** ou **more** et déduisez-en le rôle du symbole >.

c) Tapez la commande :

```
mkdir Tmp
echo Usage du prompt : > Tmp\aide_prompt.txt
```

d) Que fait cette dernière commande ?

e) Ajoutez l'aide de la commande **prompt** à la fin du fichier aide_prompt.txt

Attention : pour ajouter du texte à un fichier utilisez **>>**.

f) Vérifiez le résultat en tapant :

```
more Tmp\aide_prompt.txt
```

E. Copier / déplacer / supprimer des fichiers

Les jokers :

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

Exemple : ?ale.txt regroupe 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 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.

a) Déplacez le fichier commande.txt vers le répertoire **TMP** (**MOVE**)

b) Dans **TMP**, dupliquez le fichier aide_prompt.txt sous le nom aidep.txt (**COPY**)

c) Copiez tous les fichiers .txt contenus dans **TMP** dans le répertoire **BATCH**

d) Détruisez tous les fichiers du répertoire **TMP** (**DEL**) et du répertoire **BATCH**.

IV. Automatiser les traitements : les fichiers Batch

A. Script Batch

« Batch » signifie « lot ». 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 paramétrées et organisées à l'aide des instructions if, goto et des étiquettes.

B. 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. Ils sont des outils puissants et indispensables à tout utilisateur avancé.

1. Mon premier Batch :

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

```
Nous sommes le : date
Il est : heure
Le repertoire du systeme Windows est :
nom_du_repertoire

Il contient les fichiers systeme : liste du contenu
```

A l'aide des variables systèmes : date, time, WINDIR et HOMEDRIVE

Liste du contenu correspond à la liste des fichiers du répertoire C:\ qui ont l'attribut S (Système).

Attention, les accents sont codés différemment selon que vous êtes sous Windows (SciTE) 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 **Z :** et tapez Batch1.bat pour l'exécuter. Pourquoi ça ne fonctionne pas ? Enregistrez votre fichier sous **Z:\TP_SYS\BATCH** et retaper votre commande dans la console. Pourquoi ça marche ?

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

2. Mon second Batch : Tester l'existence de fichier et/ou de

répertoire

- Créer un fichier de commandes **testFic.bat** qui reprend l'exemple (ci-dessous) sur les expressions conditionnelles.

```
@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 present !
        ) else (
                    echo %1 est absent
          )
)
goto fin
:usage
echo usage : testFic nomfichier
:fin
```

- Affichez l'aide de la commande.
- Testez cette commande pour un fichier et un répertoire existants/non existants. Commentez le code.
- Que représente le `%1 ?`
- Que fait `if exist %1\NUL` ? Ce test ne fonctionne pas sur toutes les versions de Windows. Fonctionne-t-il ici ? Il existe des alternatives utilisant des commandes de traitement des répertoires et le test de la condition ERRORLEVEL (cf. encadré ci-dessous).

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 ≥ 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 ci-dessous pour afficher

les paramètres d'un script).

```
Rem affichage de tous les parametres
@echo off
if NOT "%1"=="" goto suite
echo Aucun parametre !
goto fin
:suite
if "%1"=="/?" goto usage
rem cas normal : au-moins un parametre
echo liste des parametres :
set nbp=0

:tantque
   if "%1"=="" goto ftq
   set /a "nbp=%nbp%+1"
   echo parametre %nbp%: %1
   set %nbp%=%1
   shift
   goto tantque
:ftq
echo il y a %nbp% parametres
goto fin
:usage
echo format de la commande : nbparam p1 p2 ... pn
66
:fin
```

3. Mon troisième Batch : Afficher la date du jour

- Ecrire un script dateDuJour.bat qui affiche :

  `Nous sommes le Xeme jour du Yeme mois de l'année 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 debut de la sous-chaine (0
est l'indice du premier caractere) lg est la
longueur de la sous-chaine
exemple : si la variable nomvar=tralala, alors
%nomvar:~1,3% désigne la chaine "ral"
```

4. 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
jticou: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 2nde et 3ème 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 2nde partie, **%j** pour l'obtention de la 3ème 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
```

5. 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 reconnaitre 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 `creelogin.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 créé
```

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

V. Webographie :

- https://miashs-www.u-ga.fr/prevert/SpecialiteIHS/Documents/TP1_systeme_DCISS1011.pdf
- https://mrproof.blogspot.com/2010/11/exercices-corriges-avec-le-registre-de.html
- http://idecibel.tuxfamily.org/courswinxp/Bases%20du%20registre.pdf
- https://www.toutwindows.com/registre.shtml
- https://www.malekal.com/registre-windows/
- https://miashs-www.u-ga.fr/~adamj/doclicence/TP3-systemes_MIASHS3-2017.pdf
- https://miashs-www.u-ga.fr/prevert/SpecialiteIHS/Documents/2-SystemeWindowsIC2A.pdf
- https://www.commentcamarche.net/contents/1093-variables-d-environnement
- https://www.courstechinfo.be/OS/FichierCmd.html
- https://windows.developpez.com/cours/ligne-commande/
- http://libertyboy.free.fr/computing/reference/envariables/index.php