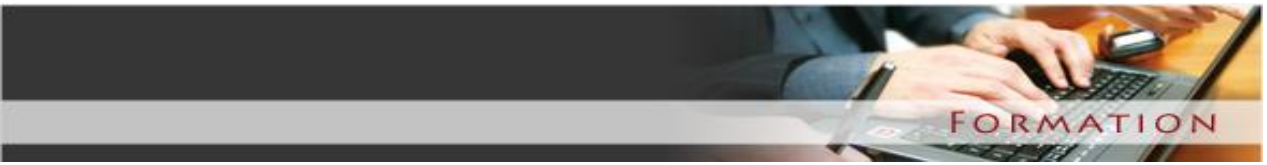


SCRIPT BATCH

Ligne de Commande Windows

PLAN:

- Scripting Windows … Pour quoi faire ?
- Connaitre les différentes commandes
- Créer un premier script Batch
- Le langage
    - *Les variables*
    - *Les fonctions*
    - *L'affichage*
- FAQ/TroubleShooting

Scripting Windows … Pour quoi faire ?

Scripting Windows … Pour quoi faire ?

Qu’est-ce qu’un script ?

-Un script est un fichier au format texte comprenant un ensemble de commandes écrites dans un langage interprété s’exécutant sur un système d’exploitation.

Mauvaise réputation ? … retour en arrière? … anachronisme?

- rime avec Programmation, complexité, manque d’intuitivité …

Les limites des interfaces graphiques ?

- Outils d'administration par défaut font l'affaire pour certaines tâches « basiques » à petite échelle.

- Les interfaces graphiques ne couvrent pas l'ensemble des cas de figure qui se présentent aux administrateurs.

Les cas courants ou le scripting facilite la vie ?

- Automatiser les tâches répétitives.

- Accéder à des fonctionnalités du système qui ne sont pas directement accessibles via les interfaces graphiques (certes nombreuses).

- *Automatiser des fonctionnalités cachées …*

Améliorer la productivité côté administration et support technique.

- Gestion des comptes utilisateurs, machines, groupes de sécurité ;
- Gestion Réseau …
- Maintenance des services ;
- Audit de l’existant ;
- Gestion globale d’Active Directory ;
- Reporting pour les serveurs et postes de travail.
- …

Améliorer la productivité côté poste de travail

- Gestion des scripts de connexions ;
- Automatisation d'applications bureautique ;
- Maintenance du système d'exploitation ;
- Personnalisation de l'interface ;
- Aide pour effectuer des tâches complexes pour utilisateurs courants.

Le temps gagné par l'utilisation du scripting peut servir à améliorer la qualité de service.

CONNAITRE LES DIFFERENTES COMMANDES

Commandes de bases :

CD : Permet de se déplacer d'un répertoire à un autre. *(Exemple: c> cd dossier)*

CD \ : Permet d'accéder à la racine d'un lecteur. *(Exemple: c> cd \ )*

DIR : Liste le contenu du répertoire courant. *(Exemple: c> dir)*

MKDIR : Création d'un dossier. *(Exemple: c>mkdir dossier)*

RMDIR : Effacer un dossier. *(Exemple: c>rmdir dossier)*

COPY : Copie des fichiers. *(Exemple: c> copy bibi.txt c:\toto.txt)*

XCOPY : Copie des fichiers et des répertoires. *(Exemple: c> xcopy bibi.txt c:\toto.txt)*

DEL : Effacer un fichier *(Exemple: c>del c:\temp.txt)*

REN : Renommer des fichier *(Exemple: ren toto.txt tata.txt)*

MOVE : Déplace un fichier. *(Exemple: c> move c:\temp.txt d:*)

EDIT : Lance un éditeur de texte sous MS-DOS.*(Exemple: c> edit toto.txt)*

MORE : Visualiser le contenu d'un fichier texte *(Exemple: c> more toto.txt)*

FORMAT : Permet d'effacer le contenu d'un lecteur. *(Exemple: c> format a:)*

CLS : Efface l'écran actuel. *(Exemple: c> cls)*

FIND : Recherche dans un fichier la ligne contenant une valeur.

CMD : Ouvre la fenêtre de commande DOS. *(Exemple: c> cmd*)

ECHO : Affiche un message. *(Exemple: c> echo salut)*

ECHO. : Permet de sauter une ligne. *(Exemple: c>echo.*)

SORT : Permet de trier une liste en fonction d'un critère. *(Exemple: c> dir | sort reverse*)

PRINT : Imprime le fichier spécifié. *(Exemple: c>print toto.txt)*

EXIT : Ferme la fenêtre MSDOS. *(Exemple: c> exit)*

TYPE : Affiche un fichier texte. *(Exemple: c> type list.txt*) même fonction que more

FC : Comparaison de fichiers.

ATTRIB : Modifie les attributs d'un fichier. *(Exemple: c> attrib c:\test +a*)

CACLS : Modifie les droits utilisateurs sur un fichier.

ASSOC : Affiche ou modifie les applications associées aux extensions de fichiers.

CHDIR : Affiche l'arborescence actuelle. *(Exemple: c:\temp> chdir)*

Commandes réseaux :

PING : Effectue un test de connectivité sur une machine distante à utiliser avec une adresse IP. *(exemple: c> ping google.fr)*

IPCONFIG : Permet de voir votre configuration réseau, adresse IP, DNS, serveur DHCP... *(exemple: c> ipconfig ,ipconfig /all)*

NESTAT : Affiche leurs connexions active sur votre machine, port et protocole. *(exemple: c> NETSTAT)*

NET USE : Connecte un lecteur réseau. *(exemple: c>net use z: \\nompc\nompartage)*

TRACERT : Affiche les adresses de toutes les passerelles pour accédez à une destination *(exemple: c> tracert google.fr)*

ARP : Permet de voir et de modifier la table ARP, correspondance MAC <-> IP des machines connectées sur le réseau.

TELNET : Telnet sur une autre machine *(exemple: c> telnet 192.168.0.1)*

NET SEND : Envoi un message sur une machine sur le réseau.*(exemple: c> net send %computername% coucou)*

NSLOOKUP : Permet de faire une résolution DNS, l'exemple donne les IP de google *(exemple: c> nslookup google.fr)*

FTP : Lance un module FTP permettant de faire des transfert de fichiers. *(exemple: c> ftp 01net.com)*

REXEC : Exécute des commandes sur des hôtes distants exécutant le service REXEC. Rexec authentifie l'utilisateur sur l'hôte distant avant d'exécuter la commande spécifiée.

TFTP : transfère les fichiers depuis et vers un ordinateur distant exécutant le service TFTP.

RUNAS : Permet d'éxécuter une commande avec un compte utilisateur différent.

NET TIME Permet de pouvoir synchroniser l'horloge avec un serveur.

MODE : Permet d'afficher des informations concerant les ports COM

NETSH : Permet de configurer des interfaces réseaux.

ROUTE : Permet de gérer la table de routage de la machine locale *(exemple: c> route print*)

NBTSTAT : Affiche les statistiques du protocole TCP/IP actuelles

Commandes utilitaires Windows :

DEFRAG : Permet de défragmenter un lecteur *(exemple: c> defrag c:)*

CHKDSK : Permet de lancer un scandisk sur un disque dur

NET START : Démarre un service windows *(exemple: c> net start sharedaccess)*

CLEANMGR : Permet de faire un netoyage des disques durs.

CONVERT : Convertit des volumes FAT en volumes NTFS

SCHTASKS : Permet sous Windows à un administrateur de créer, supprimer, effectuer des requêtes, modifier, exécuter et mettre fin à des tâches planifiées sur un système local ou distant.

TASKKILL : Permet sous Windows de mettre fin à une ou plusieurs tâches ou processus.

TASKLIST : Affiche la liste des applications et tâches ou processus associés actuellement activés sur un système à distance sous Windows

SHUTDOWN : Arrête ou redémarre un ordinateur local ou distant.

BOOTCFG : Configure, interroge ou modifie les paramètres du fichier boot.ini.

DISKPART : Gère des disques, des partitions ou des volumes.

SYSTEMINFO : Permet d'obtenir une foule d'information sur votre équipement.

TYPEPERF : Affiche les données du compteur de performances dans la fenêtre de commande ou dans un format de fichier journal pris en charge.

DRIVERQUERY : Recherche une liste de pilotes et de propriétés de pilotes.

MEM : Affiche des informations concernant les zones de mémoire allouées, les zones de mémoire libre et les programmes actuellement chargés en mémoire dans le sous-système MS-DOS.

LABEL : Les symboles ^ et & peuvent être utilisés dans les noms de volume.

AT : Planifie l'exécution de commandes

Commandes dédiées aux scripts :

PAUSE : Permet de stopper le traitement en cours en demandant à l'utilisateur de taper une touche pour continuer.

IF : Instore une structure conditionnelle

FOR : Permet de créer des boucles

SET : Définition d'une variable %variable% pour la rapeller

GOTO : Permet d'aller à une étiquette définit dans le script via ":".

SET VARNAME= : Crée une variable.

SETLOCAL : Crée une variable locale.

TIMEOUT : Permet de créer une temporisation avec un réglage en secondes.

Commandes pour obtenir de l'aide sur les commandes :

HELP : Permet d'obtenir la liste des commandes existantes.

HELP nomcmd : Permet d'obtenir une aide sur la commande spécifiée.

NomCmd /? : Permet d'obtenir de l'aide sur la commande spécifiée.

CRÉER UN PREMIER SCRIPT BATCH

Editeur de texte :

- Bloc note Windows (notepad)
- Notepad ++
- Pspad
- PowerBatch

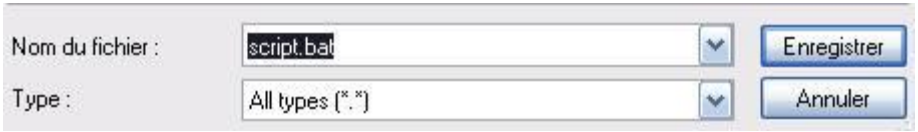
L’utilisation de logiciels utilisant la coloration syntaxique est un plus dans le cadre de la création de scripts complexes.

Création d'un fichier Batch :

```
rem la directive rem permet de commenter une ligne au sein
rem d'un fichier batch

rem la commande echo off permet de désactiver l'affichage
rem  des commandes
echo off

rem la commande cls de mettre à blanc l'affichage effaçant
rem toutes les lignes précédentes
cls

rem la commande echo texte permet d'afficher du texte dans
rem la fenêtre dos
echo Voila mon premier batch

rem la commande pause permet d'arrêter le script en
rem demandant une action de l'utilisateur
pause
```

Enregistrement du Batch :

Une fois le fichier conçu il faut l'enregistrer avec une extension spécifique « .bat », en sélectionnant le Type « All Types(*.*) », pour l'éditeur bloc note Windows.



Le fichier ainsi créé apparaît comme ceci sous Windows.

Il est également possible d'enregistrer le fichier sous l'extension « .cmd » ou encore grâce à un utilitaire de le compiler en « .com » ou « .exe »

Exécution du Batch :

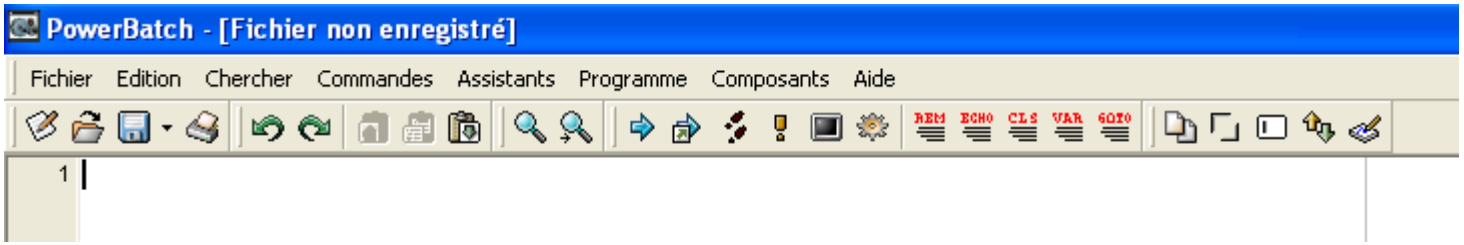
Pour l'exécuter double cliquez simplement sur le fichier. Vous pouvez sinon l'exécuter dans une tâche planifiée de programmer le lancement tous les jours par exemple.

Editeur Powerbatch :

« PowerBatch » sert à créer facilement des fichiers batch (*.bat).

Cette application permet d'inclure sans notions de programmation (à l'aide d'assistants) les commandes batch les plus fréquentes afin d'automatiser certaines tâches ou de créer aisément de petits programmes exécutables.

Il est également possible de compiler les fichiers batch produits pour les transformer en applications DOS/Windows (*.com), en proposant un support graphique à l'utilitaire Bat2Exec.



Le Langage

LES VARIABLES :

Une variable permet de stocker une donnée indiquée… à tout moment dans le script on pourra faire appel à elle.

```
rem désactive l'affichage des commandes
echo off
rem remise à blanc de l'écran
cls
rem définition de la valeur de la variable
set variable=1
rem affiche du texte en rappelant la variable grâce aux %
echo la valeur de la variable est %variable%
rem arrêt
pause
```

Résultat:

Demande de valeur à l'utilisateur :

Il semble intéressant d'interagir avec l'utilisateur en demandant une variable. Cela est possible grâce à l'utilisation du paramètre "/p", passer à la commande set, comme nous le montre cet exemple:

```
rem désactive l'affichage des commandes
echo off
rem remise à blanc de l'écran
cls
rem le /p permet de demander le retour de la variable
set /p prenom= Quel est votre prenom :
rem remise à blanc
cls
rem affiche le texte avec la variable rentrée
echo Ca va %prenom%, tu as un joli prenom :-)
rem arrêt
pause
```

Les variables d’environnement :

Windows possède un certain nombre de variables déjà renseignées concernant votre système.

```
rem désactive l'affichage des commandes
echo off
rem remise à blanc de l'écran
cls
rem liste des variables
echo Salut %USERNAME%,nous sommes le %DATE%
echo il est %Time% déjà!,
echo %RANDOM% est un chiffre aléatoire.
echo Ton PC se nomme %COMPUTERNAME%,
echo il posséde %NUMBER_OF_PROCESSORS% processeur,
echo c'est une architecture %PROCESSOR_IDENTIFIER%
rem arrêt
pause
```

Découpage d’une variable :

L'intérêt de découper une variable est d'extraire certaines parties afin de les retraiter.

Principe:

Prenons la variable d'environnement %ProgramFiles%

echo %ProgramFiles% donne C:\Program Files

Désormais nous souhaitons conserver uniquement le lecteur "c:\"

echo %ProgramFiles:~0,3%

On observe que l'ajout du ":~" indique le découpage, les valeurs suivantes "0,3" indiquent comment découper. En effet le 0 indique que le début de la sélection est le 0eme caractère, le 3 indique qu'elle s'arrête au 3eme.

Autre solution possible, par suppression de caractères:

echo %ProgramFiles:~,-13%

Conservons désormais le nom du dossier "Program Files" uniquement

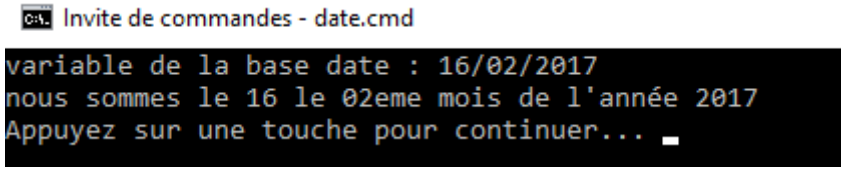
echo %ProgramFiles:~3,13%

echo %ProgramFiles:~3%

Exemple :

```
rem désactive l'affichage des commandes
@echo off
rem remise à blanc de l'écran
cls
echo variable de base date: %date%
rem découpage %date:~0,2%
rem 1er chiffre numéro du caractère de début de la sélection
rem 2eme chiffre nombre de caractères après le début
echo Nous sommes le %date:~0,2% le %date:~3,2%eme mois de l'ann,e %date:~6,4%
rem arrêt
pause
```

Résultat :



Calcul simple avec des variables :

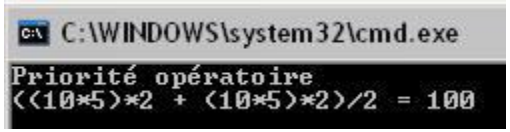
Grâce aux batch il est possible de faire des opérations simples telle qu'une addition, soustraction, multiplication ou encore division. Cela est possible avec la commande set /a.

```
@echo off
echo Addition
set /a add = 5+5
echo 5 + 5 = %add%
echo.
echo Soustraction
set /a sous = 10-5
echo 10 - 5 = %sous%
echo.
```

```
echo Division
set /a div = 10/2
echo 10 / 2 = %div%
echo.
echo Multiplication
set /a mult = 10*2
echo 10 * 2 = %mult%
pause
```

Il est possible de donner des priorités aux opérations grâce aux parenthèses

```
@echo off

echo Priorit, op,ratoire
set /a pri = ((10*5)*2 + (10*5)*2)/2
echo ((10*5)*2 + (10*5)*2)/2 = %pri%
```



LES FONCTIONS :

Les étiquettes :

La fonction étiquette permet d'intervenir sur le séquencement, typiquement votre script va exécuter les commandes les une à la suite des autres. Pour casser tout ça les étiquettes permettent d'accéder directement à un endroit du script.

rem désactive l'affichage des commandes
echo *off*
rem remise à blanc de l'écran
cls
rem définition du point de retour
*:**boucle***
rem affiche le texte salut
echo *salut*
rem indique de retourner à la :boucle
goto ***boucle***

Ce script va afficher salut à l'infini, grâce à la boucle formée avec le saut qui fait un retour sur la directive *:boucle*.

Les conditions avec la fonction IF / ELSE :

La fonction If permet de définir une condition, ainsi il est possible de définir une comparaison qui fera accepter ou non la condition.

Structures de la commande:

IF "chaîne1" comparant "chaîne2" (action)
IF "chaîne1" comparant "chaîne2" (action1) else (action2)

Exemple:

IF toto EQU toto (echo la tete a toto)
IF toto EQU tata (echo la tete a toto) else (echo différent)

N.B: L'ajout du paramètre /i permet d'ignorer la casse.

Les comparants utilisables sont :

EQU - égal à
NEQ - différent de
LSS - inférieur à
LEQ - inférieur ou égal à
GTR - supérieur à
GEQ - supérieur ou égal à

N.B : Il est possible de comparer des chaînes de caractères ou bien des chiffres.

Exemple :

Avec des SI ont pourrait mettre Bordeaux en bouteille, démontrons que cela est possible (utilisation des SI afin de créer une condition pour sortir de la boucle).

```
echo off
cls
rem défini le point de retour
:boucle
rem définit une variable incrémenté de 1 à chaque passage
set /a count = count + 1
rem affiche la variable à chaque passage
echo %count%
rem SI %count% est égal à 10 alors aller au saut :fin
if %count%==10 goto fin
goto boucle
:fin
rem Insertion d'une une variable
echo Grace a des SI on mis %count% fois Bordeaux en bouteille
pause
```

Tests sur des fichiers :

Test sur l'existence d'un fichier :

```
rem désactive l'affichage des commandes
@echo off
rem remise à blanc de l'écran
cls
rem test la présence du fichier temp.bat alors afficher
rem existe sinon afficher existe pas.
if exist temp.bat (echo existe) else echo existe pas
pause
```

Test sur la non-existence d'un fichier :

```
rem désactive l'affichage des commandes
echo off
rem remise à blanc de l'écran
cls
rem test la présence du fichier temp.bat alors
rem afficher existe sinon afficher existe pas.
if not exist temp.bat (echo existe pas ) else echo existe
pause
```

La fonction CHOICE :

Comme son nom l'indique elle permet de faire un choix est d'être redirigé vers un saut..

Attention la fonction CHOICE n'est pas disponible sur le système de base de Windows XP, pour résoudre ce problème il suffit d'installer Powerbatch il installe la commande en même temps que son programme.

Présente sur les versions supérieures…

```
Echo off
:debut
cls

Echo Question: Quelle est la capitale de la suŠde ?
Echo .
Echo R,ponse a: Olso
Echo R,ponse b: Stokholm
Echo R,ponse c: Reykjavik
Echo q: Quitter
rem la commande choice utilise une liste de paramètres ici abcq qui font
rem  référence à 4 niveau d'erreur a fait référence au 1
CHOICE /C:abcq Faites votre choix

IF %ERRORLEVEL%==1 goto a
IF %ERRORLEVEL%==2 goto b
IF %ERRORLEVEL%==3 goto c
IF %ERRORLEVEL%==4 goto q
```

```
:a
cls
echo Perdu ! la capitale de la suŠde n'est pas Olso mais Stokholm
GOTO Fin

:b
cls
echo Gagn,e ! la capitale de la suŠde est bien Stokholm
GOTO Fin

:c
cls
echo Perdu ! la capitale de la suŠde n'est pas Reykjavik mais Stokholm
GOTO Fin

:Fin
pause
goto debut
:q
```

La boucle FOR :

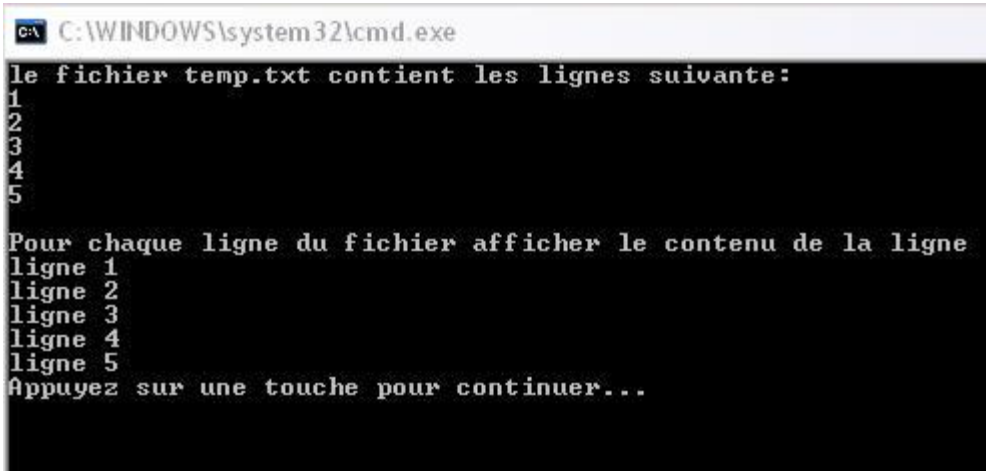
Cette fonction for permet de pouvoir parcourir un ensemble de lignes dans un fichier, et d'exécuter des actions pour l'élément parcouru.

Structure simple:

FOR /f %%variable IN (ensemble) DO commande

N.B: On peut noter que les variables utilisées ici, ne sont pas de la forme %var%, mais de la forme %%a.

```
@echo off
del temp.txt
rem création d'un fichier
echo 1 >> temp.txt
echo 2 >> temp.txt
echo 3 >> temp.txt
echo 4 >> temp.txt
echo 5 >> temp.txt

rem visualisation du contenu
echo le fichier temp.txt contient les lignes suivantes:
more temp.txt

echo.

echo Pour chaque ligne du fichier afficher son contenu
FOR /f %%i IN (temp.txt) DO (
echo ligne %%i
)
pause
```



Structure « complexe » :

```
@FOR /f "tokens=2 delims=," %%a in (c:\data.txt) do (
@echo %%a

)
pause
```

Dans cette exemple nous ajoutons deux paramètres:
Tokens: Ce paramètre indique que la variable « %%a » prendra la deuxième colonne du fichier c:\data.txt.

Delims: Ce paramètre indique que le délimiteur est la virgule, ce séparateur délimite en fait les colonnes du fichier c:\data.txt.
Pour pouvoir interroger chaque colonne d'un fichier il faut modifier le paramètres "token". Si *"tokens=1,2"* alors un *"echo %%a"* ramera la valeur de la première colonne et un *"echo %%b"* ramènera la valeur de la deuxième colonne.

La fonction FIND … recherche dans un fichier, tableau :

Grâce aux batchs il est possible à l'intérieur d'un fichier texte de faire une recherche, à la fois sur une ligne et sur une colonne.
Tout d'abord il est préférable d'utiliser des fichiers possédant des séparateurs simples, comme par exemple les fichiers « .cvs » avec le délimiteur virgule.

Fichier: c:\data.csv
1,pierre,martin
2,antoine,dupont
3,marcel,roger
4,thomas,froger
5,marie,simon
6,lea,robert
7,ines,bertrand
8,kenza,fournier
9,laure,richard

Sélection d'une ligne:
find "4" c:\data.csv
pause

Grâce à au find nous avons sélectionné la ligne numéro 4.

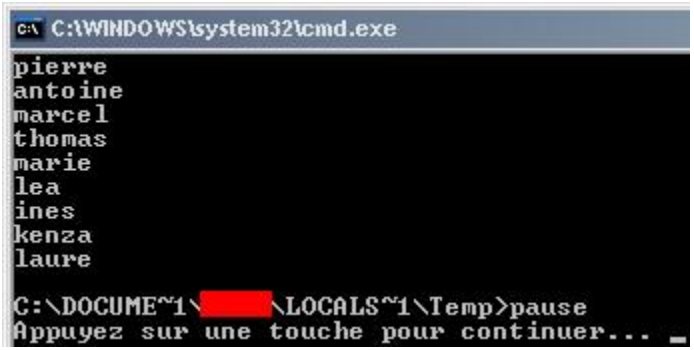
Sélection d’une colonne :

A l'aide, de la fonction FOR il est possible de sélectionner une colonne au sein d'un fichier avec délimiteur.

```
@FOR /f "tokens=2 delims=," %%i in (c:\data.csv) do @echo %%i
pause
```

N.B : Il est possible avec la fonction FOR d'exécuter plusieurs commandes dans le do, en intégrant des parenthèses après le DO.

Avec ce batch nous pouvons afficher la 2ème colonne du fichier dont les séparateurs sont des ","



Combinaison des deux :

```
@echo off
rem définition de la variable var qui définie la place recherchée
set /p var=Quelle place cherchez-vous (1-9)?
rem enregistre dans le fichier tmp.txt la ligne contenant le numéro indiqué
find "%var%" c:/data.csv >> tmp.txt
rem dans le fichier tmp.txt sélection de la colonne 2 à 3, la colonne 2 %%i
rem la colonne 3 %%j
FOR /f "tokens=2-3 delims=," %%i in (tmp.txt) do @echo le num,ro %var% est
%%i %%j
del tmp.txt
pause
```

Création d’un compteur :

La création d'un compteur peut être très utile dans certaines situations, par exemple afin de temporiser. Ce dernier aura pour effet de retarder certaines commandes.

Il s'agit de créer une boucle à l'aide d'une étiquette incrémentant une variable ici %count%

```
rem temporistation
:boucle
set /a count = count + 1
if %count%==1000 goto finboucle
goto boucle
:finboucle
```

En jouant sur le seuil de déclenchement, ici "1000" nous pouvons faire varier la durée de la temporisation.

Les paramètres :

Ce sont en fait des variables que l'utilisateur peut introduire à l'exécution du script. Mais voyons cela avec un exemple simple.

```
@echo off
echo J'ai pass, le paramŠtre : %1
pause
```

Ce micro script permet grâce à la variable %1 de récupérer le premier mot passé en paramètre au script. Comme vous l'aurez deviné %2 est le deuxième paramètre au script…

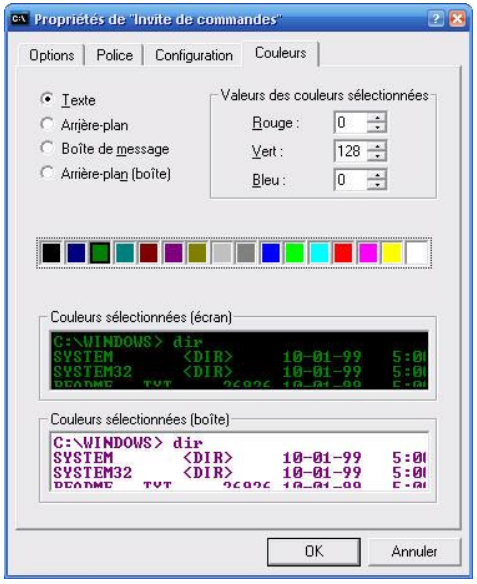
AFFICHAGE

Affichage :

Afin de rendre plus agréable votre script à l'utilisateur, il est possible de personnaliser légèrement l'affichage.

Couleurs texte et arrière plan sous MS-DOS :

Pour commencer il est possible de modifier les couleurs, vous avez sûrement remarquer qu'il était possible de modifier les couleurs grâce à un clic droit dans la barre de titre d'une fenêtre MS-DOS.



COLOR [attr]

attr Spécifie les attributs de couleurs de l'apparence de la console

Les attributs de couleurs sont spécifiés par DEUX chiffres hexadécimaux -- le premier correspond à l'arrière plan, le second au premier plan. Chaque chiffre peut prendre n'importe quelle de ces valeurs :

0 = Noir	8 = Gris
1 = Bleu foncé	9 = Bleu clair
2 = Vert	A = Vert clair
3 = Bleu-gris	B = Cyan
4 = Marron	C = Rouge
5 = Pourpre	D = Rose
6 = Kaki	E = Jaune
7 = Gris clair	F = Blanc

Si aucun argument n’est donné, cette commande restaure les couleurs

Si aucun argument n'est donné, cette commande restaure les couleurs sélectionnées au moment où CMD.EXE a été ouvert.

Cette valeur vient soit de la fenêtre de la console, du commutateur en ligne de commande /T, ou de la valeur DefaultColor du registre.

La commande COLOR met ERRORLEVEL à 1 si vous tentez de l'exécuter avec la même couleur pour l'arrière et le premier plan.

Exemple : "COLOR fc" affiche du rouge sur du blanc

Exemple :

```
@echo off
color 0A
echo Mon script est en couleur!
pause
```

Les cadres :

Il est possible de créer des faux cadres afin d'avoir une interface plus agréable. Pour cette section je vous recommande vivement Powerbatch qui possède un assistant pour crée ces cadres.

Pour crée ces derniers il faut utiliser des caractère spéciaux, ainsi:

```
@echo ÉÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ»
@echo ºQuel joli cadre !º
@echo ÈÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍÍ¼
pause
```

Nom de la fenêtre MS-DOS :

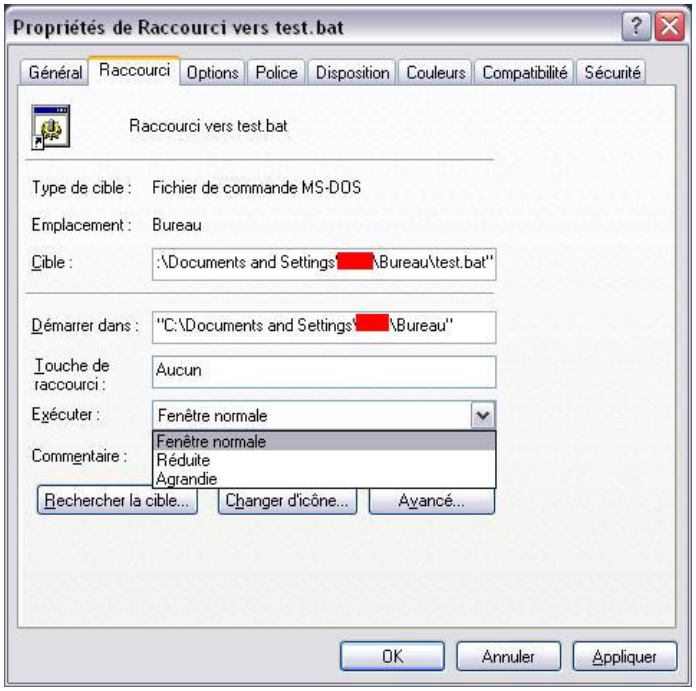
Grâce à la commande "title" il est possible de remplacer le joli « Invite de commande » par ce que vous voulez.

*@**title** Script !*
pause

Exécution en fenêtre réduite :

Lorsque vous exécuter un batch automatiquement au démarrage de l'ordinateur ou encore en tâche planifiée, il est souvent utile de masquer l'exécution du batch pour le rendre transparent aux yeux des utilisateurs.
Il faut tout d'abord crée un raccourci vers le fichier batch (clic droit)
Ensuite il faut aller voir les propriétés du raccourci

Enfin vous trouverez trois modes d'exécution, Fenêtre normale, Réduite, Agrandie, sélectionnez « Réduite ».



Voila le batch sera désormais visible uniquement dans la barre des tâches, vous pouvez mettre le raccourci dans Démarrer -> Tous les programmes -> Démarrage pour une exécution au démarrage de l'ordinateur.

Effacer l’écran :

L'effacement de l'écran permet de remettre à blanc toutes les lignes de la fenêtre DOS, il faut utiliser la commande « cls »

exemple:

*@**echo** ecran 1*
cls
*@**echo** ecran 2*
pause

Exemple script affichage :

```
@title Mon Script !
@set t=0
:boucle
cls
@color %t%
@echo É͍͍͍͍͍͍͍͍͍͍͍͍͍͍͍͍»
@echo ºQuel joli cadre !º
@echo È͍͍͍͍͍͍͍͍͍͍͍͍͍͍͍͍¼
@set /a t = t + 1
if %t%==99 goto fin
goto boucle
:fin
pause
```

FAQ / TROUBLESHOOTING

Les chemins comportant un espace, ne fonctionnent pas :

MS-DOS ne gère pas les noms comportant plus de 8 caractères, et ceux comportant des espaces.
Solution 1: Les guillemets
cd c:"program files"
Solution 2: Découper le nom
cd c:progra~1

Les accents ne s'affichent pas dans mes batchs, pourquoi ? :

En effet les batchs utilisent les caractères « unicode », il faudra donc, pour utiliser les accents ou autres caractères spéciaux, utiliser un logiciel tiers. Edit de windows remplit cette mission, mais Powerbatch sera plus confortable.

Exemple de conversion automatique:
echo €a utilise des caractŠres accentu, .

Comment caché les lignes de commandes lors de l'exécution ? :

Pour ne pas afficher les lignes de commandes, il y a deux solutions:

Solution 1: mettre cette ligne en début du batch.
@echo off

Solution 2: placé un @ devant toutes les lignes de commandes.
@echo salut

Comment faire pour créer un log de mon batch? :

Pour crée un log, d'un batch il suffit de faire une redirection du résultat de la commande vers un fichier texte.
Exemple:
xcopy c:\temp.* c:\temp1 >>*
c:\log\%DATE:~6,4%%DATE:~3,2%%DATE:~0,2%.txt

Comment enregistrer une variable dans une boucle for ? :

Dans un batch il arrive que l'on ait à définir une variable dans une boucle for, cependant il est impossible en « DOS » d'affecter une variable dans une boucle for:
Exemple de problème:

```
for /f %%a in (c:\list.cvs) do (
set var=%%a
echo %var%
)
```

Solution: Utiliser l'expansion retardée

```
setlocal enableDelayedExpansion
for /f %%a in (c:\list.cvs) do (
set var=%%a
echo !var!
)
endlocal
```

Notons qu'avec cette méthode le rappel de la variable ne se fait pas avec des %var% mais avec !var!

Autre solution, faire appel à un autre batch via la commande call est y placer le contenu du DO

Comment télécharger un fichier via un FTP avec un batch? :

Windows intègre déjà un outil pour se connecter à un site ftp, via la commande ftp, ensuite le principe et de créer un fichier de réponse avec les paramètres de connexion.

Exemple:

```
echo open monftp.fr  > ftp.ftp
echo USER nomutilisateur motdepasse >> ftp.ftp
echo cd repertoire >> ftp.ftp
echo mget nomfichier >> ftp.ftp
echo bye >> ftp.ftp
ftp -inv -s:ftp.ftp
```

Ainsi via se batch on se connecte au ftp, on se déplace dans le répertoire "répertoire" on télécharge le fichier "nomfichier" puis on quitte.

Comment mettre le résultat d'une commande dans une variable ? :

Pour ce faire il suffit de mettre la commande entre simple quote('commande') dans une commande for:
Par exemple si l'on veut récupérer le chemin courant :
for /f %%d in ('chdir') do set chemin=%%d
echo %chemin%
ou bien la date
for /f %%d in ('date /t') do set madate=%%d
echo %madate%

Afficher des informations sur un fichier :

Vous pouvez afficher la date et l\'heure de création dans un fichier, pour cela il faut utiliser le paramètre %~t
Dans un nouveau batch copier coller ceci:
@echo off

for %%a in (\"C:\\test.txt\") do echo %%~ta

pause
Constatez le résultat.

Vous pouvez afficher la taille du fichier (en octets) pour cela il faut utiliser le paramètre %~z

Dans un nouveau batch copier coller ceci:
@echo off

for %%a in (\"C:\\test.txt\") do echo %%~za

pause
Constatez le résultat.

Vous pouvez afficher l\'extension d\'un fichier (en octets) pour cela il faut utiliser le paramètre %~x

Dans un nouveau batch copier coller ceci:
@echo off

for %%a in (\"C:\\test.txt\") do echo %%~xa

pause
Constatez le résultat.

Vous pouvez afficher a ou les attributs d\'un fichier pour cela il faut utiliser le paramètre %~a

Dans un nouveau batch copier coller ceci:

@echo off

for %%a in (\"C:\\test.txt\") do echo %%~aa

pause

Constatez le résultat.

De plus, la substitution de références de variables FOR a été améliorée. Vous pouvez maintenant utiliser la syntaxe optionnelle suivante :

%~I - étend %I en supprimant les guillemets (")
%~fI - étend %I en nom de chemin d'accès reconnu
%~dI - étend %I en lettre de lecteur uniquement
%~pI - étend %I en chemin d'accès uniquement
%~nI - étend %I en nom de fichier uniquement
%~xI - étend %I en extension de fichier uniquement
%~sI - chemin étendu contenant uniquement des noms courts
%~aI - étend %I en attributs du fichier
%~tI - étend %I en date/heure du fichier
%~zI - étend %I en taille du fichier
%~$PATH:I - parcourt les répertoires de la variable d'environnement PATH et étend %I en nom du premier fichier reconnu trouvé. Si le nom de la variable d'environnement n'est pas défini ou que le fichier n'est pas trouvé par la recherche, alors ce modificateur étend en chaîne vide

Vous pouvez combiner les modificateurs pour obtenir des résultats composés :

%~dpI - étend %I en lettre de lecteur et chemin d'accès uniquement
%~nxI - étend %I en nom de fichier et extension uniquement
%~fsI - étend %I en nom de chemin complet avec noms courts uniquement
%~dp$PATH:i - parcourt les répertoires listés dans la variable d'environnement PATH à la recherche de %I et étend en lettre de lecteur du premier trouvé.
%~ftzaI - étend %I en DIR comme ligne en sortie

Dans les exemples ci-dessus %I et PATH peuvent être remplacés par d'autres valeurs valides. La syntaxe %~ se termine par un nom de variable FOR valide. Le choix de noms de variables en majuscules comme %I facilite la lecture et empêche toute confusion avec les modificateurs qui ne tiennent pas compte de la casse.