

VIRGA

GUIDE DE RÉFÉRENCE

MS-DOS

TOUTES VERSIONS SOUS WINDOWS
(DE 98 À XP)

© Editions OEM (Groupe Eyrolles), 2005
ISBN 2-212-11470-2

OEM

EYROLLES

Un fichier batch contient une série de commande DOS. La plupart de ces commandes peuvent être exécutées manuellement, à l'invite de commandes. En les plaçant dans un fichier batch et en l'exécutant, on s'assure que chaque commande est exécutée, dans l'ordre dans lequel elle apparaît dans le fichier batch.

De plus, les fichiers batch ne sont pas toujours formés que d'une simple suite de commandes ; on peut y inclure des structures de programmation. Nous verrons que le shell supporte notamment les tests *si-alors-sinon* (*if then else*), les boucles (*for*) et des variables.

L'intérêt des scripts

Les fichiers batch, encore appelés « scripts », présentent de nombreux avantages.

- ▶ L'exécution simple d'une commande plus complexe : en s'assurant qu'une longue commande ne comporte pas de fautes (par exemple dans l'ordre des paramètres donnés), on diminue les risques d'obtenir des messages d'erreur.
- ▶ La répétition des commandes : une tâche répétitive, fastidieuse à entrer de multiples fois au clavier, peut être automatisée en plaçant les commandes utilisées dans un fichier batch puis en appelant ce fichier batch.
- ▶ L'automatisation de certaines procédures : une manœuvre manuelle pénible à effectuer par l'utilisateur, tel une sauvegarde quotidienne des données modifiées, peut être placée dans un fichier batch appelé automatiquement.

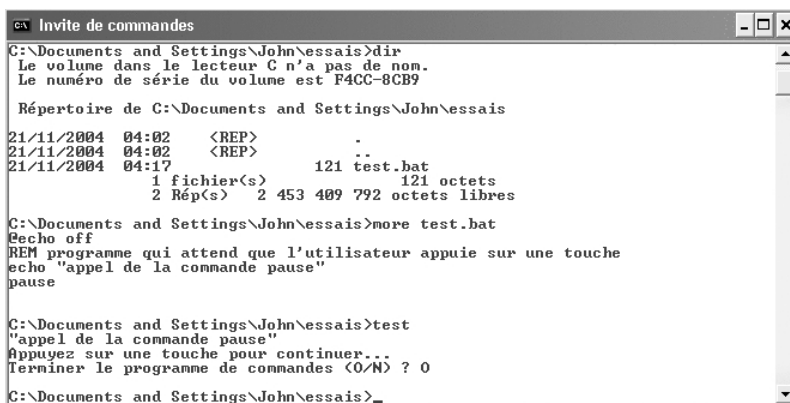


Il existe un équivalent Unix des fichiers de type batch du MS-DOS, qu'on appelle les fichiers de script (encore appelés « shell scripts »). Les fichiers batch et les « shell scripts » présentent de nombreuses similarités mais les fichiers batch sont légèrement plus simples à écrire et un peu plus limités dans leurs fonctionnalités.

L'exécution et l'interruption d'un fichier batch

Les fichiers batch sont exécutables. Il est donc également possible de les interrompre. Par exemple, si un fichier batch demande une confirmation avant d'effectuer une opération délicate, vous pouvez changer d'avis et décider de ne pas continuer son exécution.

Pour exécuter un fichier batch, il n'est pas nécessaire de taper l'extension *.bat* à l'invite du DOS : le début du nom du fichier suffit. Ainsi, pour exécuter un fichier s'appellant *encoder.bat*, on tape simplement « encoder » et



```
C:\Documents and Settings\John\essais>dir
Le volume dans le lecteur C n'a pas de nom.
Le numéro de série du volume est F4CC-8CB9

Répertoire de C:\Documents and Settings\John\essais
21/11/2004  04:02    <REP>          .
21/11/2004  04:02    <REP>          ..
21/11/2004  04:17                121 test.bat
               1 fichier(s)                121 octets
               2 Rép(s)    2 453 409 792 octets libres

C:\Documents and Settings\John\essais>more test.bat
Echo off
REM programme qui attend que l'utilisateur appuie sur une touche
echo "appel de la commande pause"
pause

C:\Documents and Settings\John\essais>test
"appel de la commande pause"
Appuyez sur une touche pour continuer...
Terminer le programme de commandes (O/N) ? O
C:\Documents and Settings\John\essais>_
```

La combinaison de touches <CTRL>+<C> permet d'interrompre un fichier batch.

puis on appuie sur la touche <ENTRÉE>. Les différentes commandes apparaissant dans le fichier batch sont alors successivement exécutées. Sauf, bien sûr, si le fichier batch est interrompu.

Pour interrompre l'exécution d'un fichier batch, on appuie simultanément sur les touches <CTRL>+<C> : on presse sur la touche <CTRL> et, tout en la maintenant enfoncée, on appuie sur la touche <C>. Le message « Terminer le programme de commandes (O/N) ? » s'affiche à l'écran et vous pouvez alors interrompre le fichier batch en appuyant sur la touche <O>.

- Si on précise le nom complet d'un fichier .bat, tel « encoder.bat », sur la ligne de commande, le fichier s'exécute aussi. Cependant, c'est une perte de temps.
- Un fichier batch peut également contenir, en plus de commandes, des appels à d'autres fichiers batch : il est ainsi possible de combiner autant de programmes qu'on le désire.

```

degenerate.bat - Bloc-notes
Fichier Edition Format Affichage ?

IF NOT EXIST "%JAVA_EXE%" goto error
SET MAIN_CLASS_NAME=com.intellij.internalUtilities.degenerator.Main
::
:: -----
:: There are two possible values of IDEA_POPUP_WEIGHT property: "heavy"
:: If you have VM
:: set this prope
:: configurations
SET IDEA_POPUP_WE
:: -----
:: You may specif
09/03/2004 22:50      284 poweroff-vm-default.bat
09/03/2004 22:50      284 poweron-vm-default.bat
02/11/2004 08:45      344 resume-vm-default.bat
02/11/2004 08:45      345 suspend-vm-default.bat
SET JVM_ARGS=%ID
4 fichier(s)          1 257 octets
:: -----
SET OLD_PATH=%PAT
SET PATH=%IDEA_HO
Répertoire de C:\WINDOWS\system32\MsDtcTrace
04/10/2001 11:42      19 429 msdtcctr.bat
1 fichier(s)          19 429 octets
Total des fichiers listés :
7 fichier(s)          23 449 octets
0 Rép(s)             2 451 181 568 octets libres
C:\>cd IntelliJ-IDEA-4.5\bin
C:\IntelliJ-IDEA-4.5\bin>notepad degenerate.bat
C:\IntelliJ-IDEA-4.5\bin>
  
```

L'éditeur Notepad avec un fichier .bat.



Le format d'un fichier batch

Un fichier batch est un fichier au format texte. Vous pouvez donc utiliser n'importe quel éditeur de texte pour créer ou modifier ces fichiers :

- ▶ la commande *edit* des premières versions du DOS ;
- ▶ le logiciel Notepad de Windows ;
- ▶ un traitement de texte capable d'exporter au format « texte ».

Sur la capture d'écran à la page précédente, vous pouvez constater qu'on trouve un fichier nommé *degenerate.bat* dans le répertoire *bin* d'un programme nommé IntelliJ (il s'agit d'un programme présent sur le système de l'auteur). On utilise ensuite le DOS pour lancer l'éditeur de texte Notepad, en lui demandant d'ouvrir directement le fichier *degenerate.bat*.

Bien que cela soit possible, nous ne recommandons pas d'utiliser un traitement de texte, tel Word, pour éditer les fichiers batch. Ceux-ci sont beaucoup trop simples pour qu'il soit nécessaire de recourir à des applications aussi gourmandes en ressources !

L'extension .bat

Par convention, tous les fichiers batch sous DOS portent l'extension *.bat*. Un système Windows ne contient, par défaut, que très peu de fichiers batch. Seul le fameux *autoexec.bat* est parfois présent (mais, bien souvent, vide).

Par contre, de nombreux programmes utilisent encore, à l'heure actuelle, des fichiers batch pour effectuer l'une ou l'autre tâche. C'est le cas notamment de certains programmes destinés à tourner sur différentes plateformes (par exemple Windows, Mac OS X et Linux).

Pour voir si votre système contient déjà des fichiers portant l'extension *.bat*, vous pouvez vous rendre à la racine de votre disque dur et y entrer la commande suivante :

```
C:\> dir /s *.bat
```

La création d'un fichier batch

Pour créer un fichier batch, il suffit de créer un fichier texte, d'y placer des commandes DOS et de donner l'extension *.bat* à ce fichier. Notez qu'il n'est pas utile de donner le nom d'une commande existante à votre fichier batch : ce serait la commande qui serait généralement exécutée, à moins de spécifier le chemin d'accès complet à la commande.

Par exemple :

```
C:\> type cd.bat
echo "test"
C:\> cd temp
C:\temp> cd ..
C:\> .\cd temp
test
C:\>
```

Nous voyons ici que le répertoire C: contient un fichier malencontreusement nommé *cd.bat*. Lorsqu'on exécute la commande *cd temp* puis la commande *cd ..*, on se déplace bien vers le répertoire *temp* puis vers la racine du disque (la commande *cd* : sert à changer de répertoire). Par contre, en entrant *.\cd temp*, c'est le fichier batch *cd.bat* qui est exécuté : le texte « test » s'affiche à l'écran (alors qu'on pensait se rendre le répertoire *temp*).

Cet exemple peut vous sembler confus mais le comportement du DOS est pourtant tout à fait logique : pour éviter d'éventuels risques de confusion, le plus simple est de ne pas donner le nom de commande DOS à vos fichiers batch.

Notez qu'il n'est pas nécessaire de recourir à un éditeur de texte pour créer un fichier batch : on peut se contenter de demander à ce que l'entrée standard (le clavier) soit redirigée dans un fichier. Pour ce faire, il faut utiliser la commande suivante :

```
C:\essais> copy CON exemple.bat
```

Le terme CON est utilisé pour « console ». On entre alors son texte en utilisant la touche <ENTRÉE> pour passer à la ligne après chaque ligne de com-



mande. On termine le fichier en entrant un code de fin de fichier à l'aide, soit du raccourci <CTRL>+<Z>, soit de la touche de fonction <F6>. Les caractères « ^Z » apparaissent alors à l'écran et il suffit d'appuyer sur la touche entrée pour créer le fichier batch.

Par exemple :

```
C:\essais> copy CON exemple.bat <entrée>
REM programme d'exemple <entrée>
echo "essai" <entrée>
<ctrl>+<z> <entrée>
```

Le message « Un fichier copié » indique que l'opération s'est déroulée avec succès et un fichier nommé exemple.bat se trouve à présent sur le disque. Il contient le texte suivant :

```
C:\essais> type exemple.bat
REM programme d'exemple echo "essai"
```

Cette façon de faire n'est toutefois pratique que pour de petits textes, d'autant qu'elle ne permet pas la modification d'un fichier existant.

Les caractères spéciaux

Nous avons déjà vu qu'il était possible d'obtenir le caractère « \ » (nommé *backslash*) en utilisant une combinaison de touches quelque peu particulière. Il est possible, en fait, d'obtenir n'importe quel caractère à l'aide d'une telle combinaison.

Lorsqu'on travaille sous DOS, il peut arriver que le clavier ne soit pas correctement configuré ou, tout simplement, qu'un caractère dont on a besoin n'y apparaisse pas. C'est très courant lorsqu'on lance un DOS de secours car de nombreuses disquettes de démarrage utilisent un clavier ne correspondant pas au clavier du système. Vous pourriez ainsi, par exemple, obtenir un clavier QWERTY alors que le vôtre est AZERTY, ou encore un clavier AZERTY de type suisse alors que le vôtre est un clavier AZERTY de type français !

Voici une liste de quelques caractères dont vous pourriez avoir besoin :

- ▶ @ : pour demander à ce qu'une commande d'un fichier batch n'apparaisse pas à l'écran avant d'être exécutée ;
- ▶ \ : pour indiquer un répertoire ;
- ▶ ~ : pour indiquer un fichier dont le nom long a été tronqué au format 8.3 ;
- ▶ * : pour indiquer un caractère générique ;
- ▶ | : pour enchaîner deux commandes DOS ;
- ▶ > : pour rediriger la sortie d'une commande ;
- ▶ < : pour rediriger l'entrée d'une commande.

Ces caractères appartiennent tous au code ASCII, qui contient 128 caractères numérotés (et non 256 comme on peut souvent le lire), par convention, de 0 à 127.

On peut obtenir ces caractères en maintenant la touche <ALT> enfoncée pendant qu'on entre le numéro ASCII du caractère désiré, exprimé en notation décimale, sur le pavé numérique situé à la droite du clavier. On relâche ensuite la touche <ALT> et le caractère apparaît alors à l'écran. Par exemple, pour obtenir le caractère «|», dont le code ASCII est 124, on appuie sur la touche <ALT> puis sur <1>, <2> et <4> avant de relâcher la touche <ALT>.

L'explication peut sembler longue, mais la manoeuvre est fort simple dès qu'on l'a effectuée quelques fois.

Voici les codes ASCII correspondant aux caractères dont vous pourriez avoir besoin :

Caractère ASCII	code	nom (anglais)
!	33	<i>exclamation mark</i>
#	35	<i>number sign, square, hash</i>
\$	36	<i>dollar sign</i>
%	37	<i>percent sign</i>
&	38	<i>ampersand</i>
*	42	<i>asterisk, star</i>
/	47	<i>slash</i>



<	60	<i>lesser than sign</i>
>	62	<i>greater than sign</i>
?	63	<i>question mark</i>
@	64	<i>at sign</i>
\	92	<i>backslash</i>
^	94	<i>circumflex accent, caret</i>
_	95	<i>underscore</i>
	124	<i>pipe</i>
~	126	<i>tilde</i>

Le langage de programmation des fichiers batch

La création d'un fichier batch peut s'apparenter à la programmation. Les fichiers batch n'offrent pas les mêmes possibilités qu'un langage de programmation de haut niveau (tel Java, C++, C#, etc.) ni celles des langages de script (tel Perl, Python, etc.), mais ils peuvent tout de même se révéler très utiles pour automatiser des tâches répétitives.

La première bonne habitude à prendre en programmation, quel que soit le langage utilisé, consiste à placer des commentaires dans le code pour expliquer ce qu'il fait.

Pour cela, il faut utiliser la commande DOS nommée *rem*. Cette commande ne fait rien du tout, si ce n'est de dire au programme que la ligne qui suit la commande ne doit pas être interprétée.

Le système d'exploitation ignore complètement les lignes commençant par la commande *rem* tandis que vous, ou qui que soit d'autre, pourrez vous y retrouver, grâce à ces commentaires.

Le bon usage consiste à mettre d'office au moins une ligne *rem* au début du fichier batch, expliquant à quoi sert le fichier. Ensuite, si le fichier contient de nombreuses commandes, il est intéressant d'ajouter quelques commentaires aux endroits clés de votre fichier batch.

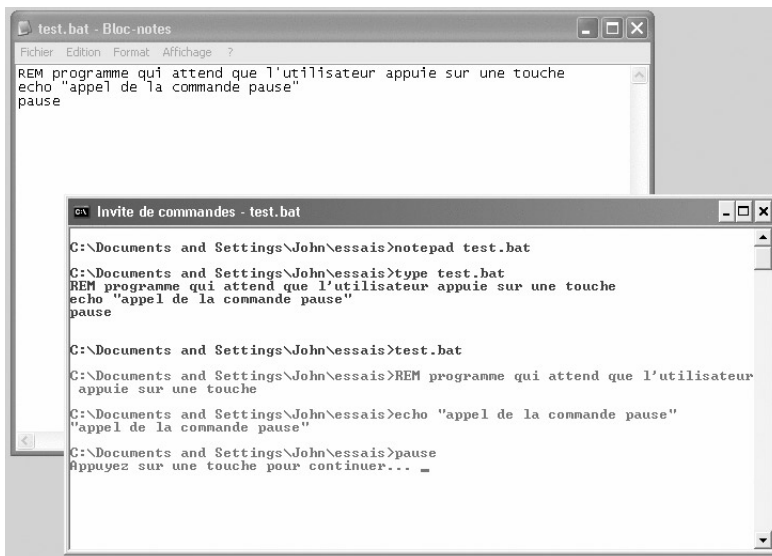
Un premier exemple

La façon la plus simple pour comprendre le fonctionnement des fichiers batch consiste à en créer quelques-uns et à les exécuter, pour voir ce qui se produit. Commençons par un premier exemple, qui contient les trois lignes suivantes :

```
REM programme qui attend que l'utilisateur appuie sur une touche
echo " appel de la commande pause "
pause
```

Créez un fichier texte (par exemple à l'aide de Notepad ou de la commande Edit) contenant ces trois lignes, sauvez-le sous le nom *test.bat* (par exemple dans votre répertoire personnel), assurez-vous qu'il contient bien ces trois lignes, puis essayez de l'exécuter depuis l'invite de commandes.

Sur la capture d'écran ci-dessous, vous pouvez constater qu'on appelle tout d'abord la commande Notepad. On s'assure ensuite que le fichier *test.bat* contient bien les trois lignes de commandes désirées (en tapant *more test.bat*), puis on exécute ce fichier.



L'édition et l'exécution d'un premier fichier batch.



Ne prenez pas les lignes de sortie de la commande *test.bat* pour des commandes entrées par l'utilisateur : dans cet exemple, tout ce qui suit l'appel au fichier *test.bat* a été communiqué au DOS automatiquement.

En effet, le DOS affiche par défaut toutes les informations contenues dans les fichiers batch. Pour désactiver cette fonctionnalité, on place souvent la commande suivante au début des fichiers batch :

```
@echo off
```

Vous pouvez à présent modifier ce premier exemple, en y ajoutant ce nouvel ordre au début. Le programme devient :

```
@echo off
```

```
REM programme qui attend que l'utilisateur appuie sur une touche
```

```
echo " appel de la commande pause "
```

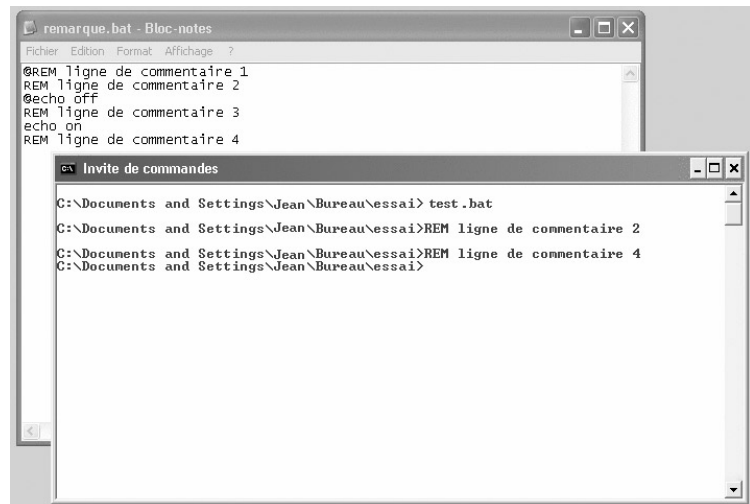
```
pause
```

Vous pouvez à nouveau exécuter ce programme :

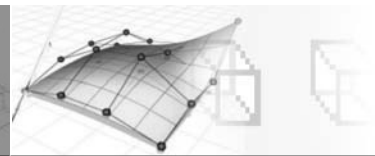
```
C:\Documents and Settings\Jean> test.bat
```

```
" appel de la commande pause "
```

```
Appuyez sur une touche pour continuer...
```



L'exécution d'un fichier batch contenant des commentaires.



A présent les deux seules informations apparaissant à l'écran sont les sorties standards des commandes *echo* et *pause*.

Les lignes comportant des remarques ou toute autre commande sont donc affichées à l'écran si la commande *echo* est activée, c'est-à-dire si elle est à la valeur « on » (c'est sa valeur par défaut). Une ligne est invisible, lors de l'exécution du fichier batch, si la commande *echo* est désactivée (par la commande *echo off*) ou si la ligne est précédée d'un arobase (le caractère « @ »).

La capture d'écran à la page précédente illustre le fonctionnement du caractère « @ » et des commandes *echo on* et *echo off* dans les fichiers batch. L'exemple contient quatre lignes de commentaires précédées par la commande *rem* :

- ❶ la première ligne ne s'affiche pas car elle est expressément précédée du caractère « @ » ;
- ❷ la deuxième ligne s'affiche car la commande *echo* est, par défaut, activée ;
- ❸ la troisième ligne de commentaire ne s'affiche pas car on a expressément désactivé la commande *echo* ;
- ❹ la quatrième et dernière ligne de commentaire s'affiche car on a expressément activé à nouveau la commande *echo*.

Enfin, vous pouvez constater que la commande *echo on* n'est pas affichée à l'écran lorsqu'on exécute le script alors qu'elle n'est pas précédée du caractère « @ ». C'est normal car au moment où le DOS exécute cette commande, la fonction *echo* est toujours désactivée.

Il faut savoir que, même lorsque la sortie est désactivée, la commande *echo* peut toujours être utilisée pour afficher des messages à l'utilisateur.

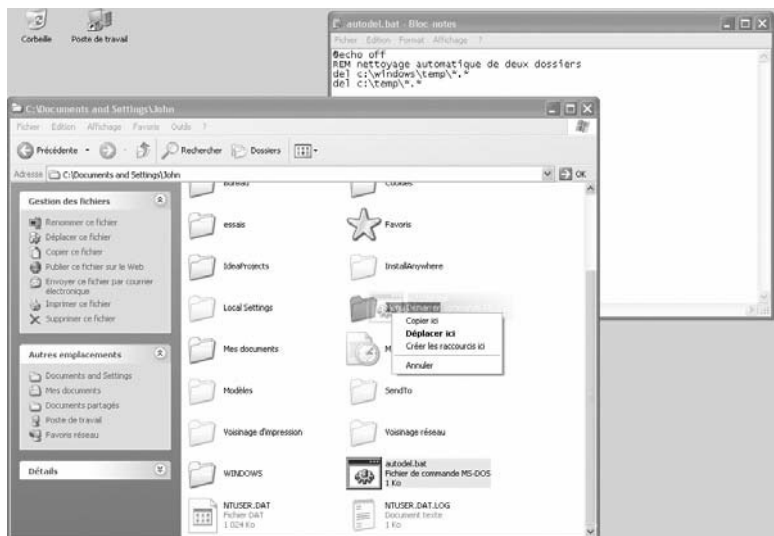
Simplement, la commande *echo off* empêche le DOS d'afficher le contenu du fichier batch à l'écran.



- Plutôt que de faire précéder toutes les commandes d'un fichier batch du signe « @ », on préfère généralement placer la commande *echo off* à la première ligne du fichier. Cependant, puisqu'à ce moment la commande *echo* n'est pas encore activée, on empêche le DOS d'afficher « *echo off* » à chaque exécution du script en faisant précéder cette commande du caractère « @ ».
- Vous pouvez également entrer la commande *echo off* à l'invite de commande du DOS : vous constaterez que l'invite disparaît de l'écran. Pour la récupérer, vous devez ensuite entrer la commande *echo on*.

L'automatisation du nettoyage du disque

Il existe différentes façons d'éliminer les fichiers temporaires présents sur le disque dur ainsi que différentes manières d'automatiser cette tâche (aussi bien sous DOS que sous Windows).



La création, dans le menu *Démarrer*, d'un raccourci vers un fichier batch.

Il est, par exemple, possible de créer un fichier batch nommé *autodel.bat* qui nettoie, à chaque démarrage, les répertoires *C:\Temp* et *C:\Windows\Temp*:

```
@echo off
REM nettoyage automatique de deux dossiers
del c:\windows\temp\*.*
del c:\temp\*.*
```

Ensuite, chaque fois que vous appellerez ce fichier, le contenu des deux dossiers *temp* et *windows\temp* sera automatiquement effacé.

Si vous désirez que cette commande soit exécutée à chaque démarrage de Windows, vous pouvez, par exemple, créer un raccourci vers cette commande dans le menu *Démarrer*.

Sur la capture d'écran à la page précédente, l'utilisateur Jean glisse, à l'aide du bouton droit de la souris, le fichier *autodel.bat* dans le menu Démarrer, en demandant de créer un raccourci. Ce faisant, le fichier batch *autodel.bat* sera exécuté chaque fois que l'utilisateur ouvrira Windows.

Soyez extrêmement prudent lorsque vous travaillez avec la commande *del* ainsi que toute autre commande DOS potentiellement dangereuse : une fausse manœuvre et vous pouvez perdre tous vos fichiers ! Pour éviter ce genre de désagrément, nous ne pouvons que vous conseiller d'effectuer régulièrement des copies de sécurité.

L'utilisation de pauses

De nombreux fichiers batch peuvent être améliorés en plaçant la commande *pause* à des emplacements adéquats. En effet, certaines opérations sont si dangereuses ou si définitives qu'il vaut mieux donner à l'utilisateur la faculté de les interrompre avant terme.

D'autres, comme l'affichage d'un message, bénéficient parfois d'un délai dans l'exécution donnant à l'utilisateur le temps de le lire. Enfin, il est par-



fois utile d'interrompre momentanément un fichier batch le temps que l'utilisateur effectue une manipulation (par exemple insérer une clé USB dans l'ordinateur). La commande DOS *pause* permet d'insérer des temps de pause dans les fichiers batch.

La commande *pause* prend, éventuellement, un argument : le message à afficher avant de bloquer l'exécution du fichier batch.

L'utilisation de la commande *pause* peut aussi donner à l'utilisateur la possibilité d'interrompre l'exécution, en donnant un avertissement. Par exemple :

```
C:\essais> superdel
```

Le fichier batch *superdel* vient d'archiver tous vos fichiers utilisateurs dans un fichier nommé *C:\backup\jean.zip*

Tous vos fichiers utilisateurs (excepté le fichier d'archive) vont à présent être effacés, sauf si vous interrompez ce fichier batch à l'aide de la combinaison <ctrl>+<c>

Appuyez sur une touche pour continuer...

Avant d'effectuer une opération délicate, le fichier batch prévient l'utilisateur qu'il est encore temps de changer d'avis. La commande *pause* vous donne ensuite tout le temps nécessaire pour choisir entre interrompre l'exécution à l'aide du raccourci <CTRL>+<C> ou appuyer sur n'importe quelle touche pour continuer l'exécution du programme.

Nous avons déjà vu qu'il était toujours possible d'interrompre un fichier batch à l'aide de <CTRL>+<C>. Toutefois, sans *pause*, vous interrompez un fichier batch n'importe quand. Dans le cas présent, si vous réalisez que, tout compte fait, vous ne désirez pas effacer tous vos fichiers mais que le script ne dispose pas de *pause*, vous pourriez, par exemple, interrompre l'exécution de la commande après que la moitié des fichiers ait déjà été effacés !

Enfin, sachez que pendant que vous réaliserez vos propres fichiers batch, la commande *pause* représente un excellent moyen de « déboguer » vos programmes. Dès qu'un programme devient quelque peu complexe, on y intro-

duit souvent quelques erreurs de distraction. L'ajout, momentané, de commandes *pause* permet de voir pas à pas ce qui se passe.

Comme la plupart des commandes pouvant apparaître dans les fichiers batch, la commande *pause* peut également être utilisée à partir de la ligne de commande, mais l'intérêt de la manœuvre est loin d'être évident.

La combinaison de pause et de echo

Lorsqu'on désire utiliser la commande *pause* pour afficher un message à l'écran, il faut que la commande *echo* soit activée (*echo on*) ; dans ce cas, en plus du message désiré, on voit apparaître la commande *pause* à l'écran, tout comme l'invite de commandes !

Cela n'est pas esthétique du tout et cela rend la lecture des informations plus difficile.

Pour remédier à ce problème, il faut combiner les commandes *pause* et *echo* de façon créative : il faut non seulement désactiver l'affichage des commandes en utilisant *echo off* mais également afficher un message personnalisé, puis modifier la sortie de la commande *pause*, en la dirigeant vers la sortie NUL (c'est-à-dire une sortie inexistante).

Par exemple :

```
@echo off
```

```
echo Placez le support USB dans le PC puis appuyez sur une touche svp.
```

```
pause > nul
```

```
...
```

L'avantage de cette méthode est d'obtenir un message sur mesure sans avoir besoin de réactiver *echo*, généralement désactivé au début du fichier batch.



Sur la capture d'écran ci-dessous, un programme nommé *backusb.bat* est appelé une première fois et affiche beaucoup trop d'informations à l'écran : pas moins de huit lignes de sorties sont affichées pour simplement demander à l'utilisateur d'effectuer une manipulation, puis d'appuyer sur une touche. Le début de ce programme est ensuite modifié et on peut constater que lorsqu'on l'appelle à nouveau, seule une ligne, très lisible, apparaît à l'écran.

```
backusb.bat - Bloc-notes
Fichier Edition Format Affichage ?
@echo off
REM programme qui effectue une backup sur un support USB
REM en demandant tout d'abord de placer ce support dans l'ordinateur
echo "Placez le support USB dans l'ordinateur et appuyez sur une touche"
pause > nul

C:\Documents and Settings\Jean\essais>backusb.bat
C:\Documents and Settings\Jean\essais>REM programme qui effectue une backup sur
un support USB
C:\Documents and Settings\Jean\essais>REM en demandant tout d'abord de placer ce
support dans l'ordinateur
C:\Documents and Settings\Jean\essais>pause "Placez le support USB dans l'ordina
teur et appuyez sur une touche"
Appuyez sur une touche pour continuer...
C:\Documents and Settings\Jean\essais>
C:\Documents and Settings\Jean\essais>
C:\Documents and Settings\Jean\essais>backusb.bat
"Placez le support USB dans l'ordinateur et appuyez sur une touche"
C:\Documents and Settings\Jean\essais>
```

La combinaison des commandes *pause* et *echo*.

Les paramètres

La plupart des fichiers batch s'exécutent au seul énoncé de leur nom, mais d'autres exigent l'entrée de paramètres. Il existe également certains fichiers batch pouvant travailler, soit avec, soit sans argument.

On appelle paramètre tout mot entré sur la ligne de commande. Le DOS affecte la valeur %0 au premier mot rencontré c'est-à-dire au nom du batch, puis la valeur %1 au deuxième mot c'est-à-dire au premier paramètre, etc. jusqu'à un maximum de neuf paramètres. Ces paramètres permettent de

moduler l'exécution du fichier batch en fonction de données choisies par l'utilisateur. On constate que le DOS précède le numéro des paramètres du signe « % ».

Supposons qu'un fichier batch nommé *garde.bat* serve à copier des fichiers dans un répertoire de façon à en conserver une copie. En utilisant la ligne de commande *copy %1 %2*, on s'assure que le premier argument indique quels fichiers copier tandis que le second argument indique où les copier. Pour copier tous les fichiers portant l'extension *.nfo* depuis un répertoire vers un lecteur D:, on utilise alors la commande *garde *.nfo d:*.

Prenons un exemple en appelant ce fichier *garde.bat* de la façon suivante :

```
garde *.nfo d: /V
```

- ▶ le paramètre %0 reçoit la valeur « garde » ;
- ▶ le paramètre %1 reçoit la valeur « *.nfo » ;
- ▶ le paramètre %2 reçoit la valeur « /V ».

Le fichier *garde.bat* ressemble à ceci :

```
@echo off
REM Programme utilisant deux variables
echo Le nom de ce programme est : " %0 "
copy %1 %2
```

Si on spécifie plus de paramètres que n'en gère le programme appelé, le DOS laisse tout simplement tomber les paramètres non utilisés !

On peut modifier le fichier *garde.bat* de façon à ce qu'il utilise 9 arguments :

```
@echo off
REM exemple de programme utilisant des variables
echo Le nom de ce programme est : " %0 "
copy %1 %2 %3 %4 %5 %6 %7 %8 %9
```

Seuls sont nécessaires les arguments indispensables au fonctionnement du fichier batch. Les autres, superflus, sont utilisés s'ils sont entrés sur la ligne de commande mais n'affectent en rien l'exécution s'ils sont omis.



Notez que l'exemple donné ici n'a d'autre intérêt que didactique : au lieu d'utiliser cette nouvelle commande *double.bat*, autant utiliser la commande *copy* prévue à cet effet !

Les commandes propres aux fichiers batch

Il existe quelques commandes qui n'ont de sens qu'utilisées dans des fichiers batch. Voici une liste des principales commandes destinées à contrôler les fichiers batch :

- ▶ *call* : appelle, depuis le fichier batch, un deuxième fichier batch ;
- ▶ *echo* : désactive l'affichage des commandes (et permet, éventuellement, d'afficher un message personnalisé) ;
- ▶ *for* : boucle qui applique successivement la même suite de commandes à plusieurs fichiers ;
- ▶ *goto* : effectue un branchement à un autre endroit du fichier batch ;
- ▶ *if* : permet l'exécution conditionnelle de certaines commandes ;
- ▶ *pause* : stoppe momentanément l'exécution du fichier batch ;
- ▶ *rem* : insère des commentaires expliquant le rôle du fichier batch ;
- ▶ *shift* : permet de décaler les paramètres.

Ces commandes sont reprises plus en détails dans le chapitre 7 – Les principales commandes – (voir page 151).

Les labels

Les labels s'utilisent conjointement à la commande *goto*. Un label, parfois appelé « étiquette », marque l'endroit où l'exécution reprendra à la suite d'une instruction *goto*. Un label se définit en plaçant le caractère « : » en début de ligne : toute ligne d'un fichier batch commençant ainsi est considérée par le DOS comme un label.

Voici un exemple de fichier batch :

```
@echo off  
:beginning
```

```
goto next
echo "Le fichier batch ne passe jamais par ici"
:next
echo "Le fichier batch passe par ici sans jamais s'arrêter"
goto beginning
```

Ce fichier comporte les deux labels *:beginning* et *:next*, situés aux deuxième et cinquième lignes de cet exemple. Le DOS n'exécutera jamais la quatrième ligne car à la troisième ligne le programme passe au label *:next*. Il affiche ensuite le message présent à la sixième ligne, avant de retourner à la deuxième comme le lui demande la dernière ligne de ce programme. Ce fichier batch, sans fin, n'a aucun intérêt et l'utilisateur doit recourir à la combinaison de touches <CTRL>+<C> pour l'interrompre.

Pour éviter de mauvaises surprises lors de l'exécution de vos fichiers batch, nous vous conseillons :

- ▶ de ne pas utiliser de caractères spéciaux ni de caractères accentués dans les noms de label ;
- ▶ de ne pas utiliser de caractère d'espacement entre le caractère « : » et le nom du label.

Un script pour analyser la connexion

Maintenant que nous avons les principes de base des fichiers batch, nous pouvons créer un premier script utile : nous allons lancer successivement les commandes *ipconfig* et *ping* pour nous assurer que l'ordinateur ait bien une adresse IP et que Internet soit accessible.

Pour ce faire, entrez le texte suivant dans un fichier que vous nommerez « dosping.bat » :

```
@echo off
REM fichier batch DOS qui appelle ipconfig et effectue un
REM ping puis attend que l'utilisateur appuie sur une touche
title Appuyez sur une touche pour quitter ce programme
ipconfig
```



```
ping -n 1 yahoo.com
pause > NUL
```

Vous pouvez ensuite créer un raccourci vers cette commande sur votre bureau (on peut voir ce raccourci en haut à gauche de la capture d'écran) afin de pouvoir lancer ce fichier batch depuis le bureau.

Analysons à présent ce fichier ligne par ligne :

- ❶ *@echo off* : afin d'éviter que le DOS n'affiche tout le programme ;
- ❷ *REM* : une ligne de commentaires expliquant ce que réalise le fichier batch ;
- ❸ *idem* ;
- ❹ *title* : pour modifier le titre de la fenêtre MS-DOS ;
- ❺ *ipconfig* : pour afficher l'adresse IP de la machine et de la passerelle ;

```
dosping.bat - Bloc-notes
Fichier  Edition  Format  Affichage  ?
@echo off
REM fichier batch DOS qui appelle ipconfig et effectue un
REM ping puis attend que l'utilisateur appuie sur une touche
title Appuyez sur une touche pour quitter ce programme
ipconfig
ping -n 1 yahoo.com
pause > NUL
```

```
Appuyez sur une touche pour quitter ce programme
Configuration IP de Windows

Carte Ethernet Connexion au réseau local:
    Suffixe DNS propre à la connexion :
    Adresse IP. . . . . : 192.168.0.175
    Masque de sous-réseau . . . . . : 255.255.255.0
    Passerelle par défaut . . . . . : 192.168.0.1

Envoi d'une requête 'ping' sur yahoo.com [66.94.234.13] avec 32 octets de données :
Réponse de 66.94.234.13 : octets=32 temps=160 ms TTL=51

Statistiques Ping pour 66.94.234.13:
    Paquets : envoyés = 1, reçus = 1, perdus = 0 <perte 0%>,
    Durée approximative des boucles en millisecondes :
        Minimum = 160ms, Maximum = 160ms, Moyenne = 160ms
```

L'utilitaire *dosping* permet de visualiser rapidement l'état de la connexion.

- ⑥ *ping -n 1* : pour envoyer un paquet PING vers le serveur yahoo.com ;
- ⑦ *pause* : pour éviter que la fenêtre ne se ferme de suite.

Par souci d'esthétisme et, surtout, pour illustrer différentes commandes, nous avons changé le titre de la fenêtre MS-DOS. Au lieu d'afficher « Raccourci vers dosping.bat », elle présente un message informant l'utilisateur qu'il peut fermer la fenêtre en appuyant sur une touche. Le message de la commande *pause* n'est alors plus nécessaire et on peut supprimer sa sortie (> NUL).

La commande *pause* est nécessaire pour donner le temps à l'utilisateur de lire les messages de sorties des commandes *ipconfig* et *ping*. En effet, lorsqu'un fichier batch n'est pas lancé depuis une fenêtre DOS, il en crée lui-même une mais celle-ci se ferme aussitôt l'exécution terminée !

Il s'agit d'une commande fort pratique si vous avez de temps à autre des petits problèmes de connexion.

Par exemple, si vous tentez de vous connecter à un de vos sites favoris mais que le navigateur semble rester bloqué, vous avez envie de savoir d'où vient le problème. Plutôt que de lancer un nouveau navigateur (voir note) et d'essayer d'accéder à un autre site (qui pourrait, lui aussi, être inaccessible), vous pouvez vous rendre sur le Bureau (touche <WINDOWS>+<D>, pour « Windows desktop ») et cliquer sur le raccourci dosping. L'opération nécessite un raccourci clavier et deux clics de souris : c'est extrêmement rapide !

Après deux ou trois utilisations, vous déchiffrez la sortie des commandes *ipconfig* et *ping* en une fraction de seconde. Généralement, si le site *yahoo.com* (mais vous pouvez en utiliser un autre) répond, c'est que la connexion est établie.

Internet Explorer lance autant de navigateurs qu'il y a de sites ouverts. Par contre, d'autres navigateurs tel l'excellent (et gratuit) navigateur Firefox, se contente d'ouvrir non pas un nouveau navigateur mais simplement un nouvel onglet. Cela présente principalement l'avantage de consommer beaucoup moins de mémoire.



Notez qu'il est possible d'apporter de nombreuses améliorations à ce script. Vous pourriez, par exemple, d'abord effectuer un ping vers une adresse IP, puis un second ping vers *yahoo.com*, ce qui permettrait d'isoler un éventuel problème de DNS. Vous pourriez encore utiliser diverses commandes DOS pour modifier les messages de sorties afin de les rendre plus lisible, etc.

Un script pour effectuer une sauvegarde

Si vous avez besoin d'échanger régulièrement des données entre deux endroits, par exemple entre votre domicile et votre lieu de travail, vous pouvez créer un script qui effectue une sauvegarde de vos répertoires importants sur une clé USB.

Vous pourriez également utiliser un tel script pour, tout simplement, effectuer une copie de sauvegarde. Ainsi, en cas de gros problème, tel le crash d'un disque dur ou le vol du PC, il vous resterait malgré tout une copie de vos données importantes.

Les clés USB représentent à présent un excellent moyen de sauvegarde : on trouve des clés dont la taille va de 256 à 512 Mo, soit presque autant qu'un CD-ROM. De plus, ce type de périphérique de stockage ne dispose d'aucune pièce mécanique (contrairement à un lecteur de CD-Rom ou de DVD) et est donc extrêmement rapide.

Vous pourriez, dès lors, créer un script très simple ressemblant à ceci :

```
@echo off
REM fichier batch qui effectue une copie de tous les fichiers
REM importants du dossier C:\Documents and Settings\Jean\importants
REM sur le support USB de 256 Mo qui a la lettre I: et ce seulement
REM si le support USB est bien accessible.

if exist I:\jeton.txt xcopy I:\importants\*. * C:\Documents and
Settings\Jean\importants /e /c /i /h /r /y /d /f

if exist I:\jeton.txt xcopy C:\Documents and Settings\Jean\importants\*. *
I:\importants /e /c /i /h /r /y /d /f
```

Dans cet exemple, le périphérique USB reçoit toujours la lettre I: et c'est donc sur ce lecteur que nous allons chercher et que nous copions les fichiers.

L'utilisation de la condition *if exist* permet de s'assurer de la présence d'un fichier nommé *jeton.txt* sur le support USB avant d'effectuer la copie. En procédant de la sorte, on évite de se tromper de clé USB.

Pour que ce genre de script fonctionne pour vous, vous devez bien évidemment modifier les noms de lecteurs et répertoires ainsi que placer un « jeton » sur votre support USB.

Une solution fort simple consiste à utiliser, comme dans cet exemple, un jeton nommé *jeton.txt*. Peu importe ce qu'il contient. Prenons le cas d'un petit texte expliquant sa présence sur le support USB :

```
C:\> more i:\jeton.txt
```

Ce fichier indique que ce support USB est bien celui sur lequel les copies de sauvegarde doivent avoir lieu. Notez que la sauvegarde est faite dans le dossier « importants ».

En connaissant bien les différentes commandes et les possibilités de programmation du DOS, il est possible de réaliser des scripts bien plus complexes et, surtout, bien mieux adaptés à vos besoins.

Il existe d'autres moyens pour partager des données qui doivent être accessibles sur plusieurs ordinateurs. Si les différents ordinateurs sont reliés à Internet, il est parfois plus facile de centraliser toutes les informations sur un serveur plutôt que sur un support physique telle la clé USB de notre exemple.