L'interpréteur de commande CMD
(Windows CLI)

Table des matières :

1 L'INTERPRETEUR DE COMMANDES CMD 2

2 VERSION DE L'INTERPRETEUR, ET INTERPRETEUR 2

3 AIDE SUR LES COMMANDES 2

4 DEFINITIONS 2

4.1 FICHIER, NOM ET EXTENSION 2
4.2 LES REPERTOIRES (OU DOSSIERS) 3
4.3 LES UNITES LOGIQUES 3
4.4 L'ARBORESCENCE 3
4.5 LE CHEMIN D'ACCES A UN FICHIER (LE PATH) 4
4.6 LE REPERTOIRE COURANT DE L'INVITE 4
4.7 LE CHEMIN ABSOLU D'UN FICHIER 4
4.8 LE CHEMIN RELATIF D'UN FICHIER 4
4.9 LES COMMANDES 4
4.10 L'INVITE DE COMMANDE (OU LE PROMPT) 4

5 LISTE ET ROLE DES COMMANDES INTERNES (MSDOS 6.22) 5

6 LISTE ET ROLE DES COMMANDES EXTERNES (MSDOS 6.22) 6

7 COMMANDES POUR LE SYSTEME DE FICHIER 8

7.1 CHANGEMENT DE REPERTOIRE 8
7.2 VISUALISATION DU CONTENU D'UN REPERTOIRE AVEC DIR 8
7.3 LES OPTIONS DE LA COMMANDE DIR 8
7.4 LE CHANGEMENT D'UNITE (LECTEUR) 9
7.5 CLASSEMENT ET GESTION DES REPERTOIRES 9
7.6 GESTION DES FICHIERS : CREER, COPIER, EFFACER, DEPLACER, RENOMMER 9

8 LES PROGRAMMES 12

8.1 LANCER UN PROGRAMME 12
8.2 VARIABLE PATH ET LA RECHERCHE DE PROGRAMME 12

9 QUELQUES VARIABLES DE L'ENVIRONNEMENT 12

9.1 LA COMMANDE SET 12
9.2 REPERTOIRE TEMPORAIRE 12
9.3 L'INVITE DE L'INTERPRETEUR OU PROMPT 13
9.4 LISTAGE 13

10 LE FICHIER BATCH 14

10.1 CREATION 14
10.2 ECHO, ET @ 14
10.3 REM, REMARQUE 14
10.4 BREAK 14
10.5 LES ARGUMENTS (PARAMETRES) 15
10.6 LES VARIABLES 15
10.7 LABEL / BRANCHEMENT 15
10.8 TEST 15
10.9 BOUCLE 16
10.10 CALL 17
10.11 PAUSE 17
10.12 UN EXEMPLE UTILE 17

11 DESIGNATION DES PERIPHERIQUES, ET SPECIFICITE 18

11.1 RE-DIRECTION ENTREE/SORTIE 18

11.2 LE TUBE (OU FILTRE) 18
11.3 FICHIERS MATERIELS, NOM DE PERIPHERIQUE : CON, NUL, PRN, LPTN, AUX, COMN 18

L'ensemble de la documentation suivante, bien que relative au système d'exploitation DOS Microsoft 6.22, peut s'appliquer avec la plupart des versions de CMD. Toutefois, certaines instructions seront absentes dans ces versions : soit parce que la version est trop ancienne, soit parce que ces instructions ont été remplacées. Je vous conseille d'entrer la commande « help » pour avoir la liste des commandes (travail à réaliser).

1 L'interpréteur de commandes CMD

CMD est l'interpréteur de commande de Windows . En anglais, CMD est le « Windows Command Line Interpreter » (CLI). Avec lui, on travaille en « mode console » dans un « terminal » virtuel. CMD est l'héritier du système d'exploitation DOS "*Disk Operating System*". Le fichier exécutable s'appelle **CMD.EXE** .
La suite de ce document a pour but de vous expliquer le "langage" de cet interpréteur, qui vous permettra de gérer vos fichiers, et d'exécuter vos programmes.

2 Version de l'interpréteur, et interpréteur

Il peut être intéressant de connaître la version de l'interpréteur, pour savoir quelles sont les commandes internes et externes disponibles. Il faut savoir que beaucoup de commandes externes ont évoluées, disparues, apparues, voire mêmes changées de nom au cours des versions. Avec la commande interne "**VER**", vous affichez le nom et le numéro de version de votre interpréteur (travail à réaliser).

3 Aide sur les commandes

La plupart des commandes affichent des informations sur leur utilisation en utilisant le paramètre "/?".
Il y a aussi la commande "HELP " qui vous donnera les informations relatives à l'ensemble des commandes et programmes de l'interpréteur.
Pour découvrir votre arborescence, parfois, la commande **tree** est disponible.

4 Définitions

4.1 Fichier, nom et extension

Un fichier est un regroupement de données désigné par un nom suivit d'une extension. La taille du fichier est variable et ces données peuvent représenter du texte, des images, du code exécutable, un mélange des trois, voire tout autre chose qui appartient à un autre programme. Pour savoir de quel type est le fichier, son nom de huit lettres est suivit d'une extension de trois caractères. Voici quelques extensions bien connues, et réservées pour des fichiers définis :

ASM, BAS, C, COB, FOR, PAS	Source d'un programme respectivement en Assembleur, Basic, C, Cobol, Fortran et Pascal.
BAK	Sauvegarde d'un fichier source avant sa dernière édition.
TXT	Texte ASCII DOS. Lisible avec la commande "type" du DOS.
DOC	Document lisible par la plupart des traitements de texte récent.

DAT	"DATA" ; regroupement de donnée relatif à un programme.
DLL	Bibliothèques de codes logiciels.
BAT	"BATCH" ; liste de séquences exécutables par l'interpréteur DOS.
SYS, DVD	Fichier de configuration des périphériques.
EXE et COM	Programme exécutable par l'ordinateur utilisant le DOS par lui-même.

Le nom d'un fichier sous DOS est de onze caractères, car le nom et l'extension sont séparés par un point. Exemple : "IMAGE.BMP". On peut aussi trouver le terme "8+3" pour désigner ce format.

4.2 Les répertoires (ou dossiers)

Les répertoires sont des regroupements de fichiers et de sous-répertoires. Cela permet à l'utilisateur de classer ses fichiers comme il le ferait avec des feuilles dans un classeur. Ainsi, nous pouvons mettre tous les fichiers relatifs à un sujet dans un même dossier (répertoire). Comme pour les fichiers, vous avez huit caractères pour le nom et trois pour l'extension, bien que cette dernière ne soit pas utilisée dans la plupart des cas. Exemple : le répertoire "DOS" ne porte pas d'extension.

4.3 Les unités logiques

Avant d'accéder à un fichier, vous devez choisir un disque : "C:" pour le disque dur. Il peut y avoir d'avantage de noms de disque sur votre ordinateur. Certains d'entre eux sont à lecture seule, comme le CD-ROM, ou une disquette protégée en écriture.

4.4 L'arborescence

L'arborescence, c'est l'ensemble des répertoires d'un même disque. Nous appelons "racine", la base d'un disque (c'est-à-dire, un niveau où nous n'appartenons à aucun répertoire). Dans cette racine, nous pouvons trouver des fichiers et des répertoires. Ces répertoires peuvent eux aussi regrouper des fichiers et encore des répertoires, et ainsi de suite. L'arborescence est donc la schématisation (par niveau) des liens de ces répertoires entre eux. Exemple : si je crée dans la racine du "C:" des répertoires "1", "2", "3" et dans les deux premiers respectivement "11", "12", "13" et "21", "22" ; puis "111" et "112" dans le "11" du "1". J'obtiens l'arborescence suivante :

```
C:\---+-1---+-11-+-111
      !   !   \-112
      !   +-12
      !   \-13
      !-2---+-21
      !   \-22
      \-3
```

Le "\" désignant la "racine" du disque. Pour visualiser l'arborescence d'un disque ou d'un répertoire, CMD propose la commande externe "TREE".

4.5 Le chemin d'accès à un fichier (le path)

Pour accéder à un fichier sur un disque, il ne suffit pas de connaître juste son nom, il faut aussi (dans la plupart de cas) connaître sa localisation dans l'arborescence (le chemin). C'est la problématique du nommage des éléments à retrouver.
Par exemple : la chaîne "C:\1\12\FICH.DAT" désigne le fichier "FICH.DAT" dans le répertoire "12" du répertoire "1" du disque "C:"
Rq : il s'agit du chemin absolu puisqu'il part de la racine de l'unité (voir plus loin)

4.6 Le répertoire courant de l'invite

(Working Directory)
L'invite nous informe de l'unité et son répertoire dans laquelle nous allons travailler. C'est ce que nous appellerons l'unité active, avec son **répertoire courant** (répertoire de travail).
Pour l'interpréteur de commande, chaque unité ou lecteur logique a son répertoire courant. De ce fait, en désignant seulement une unité logique, nous accédons au répertoire courant de cette unité. Nous ne serons donc pas obligé de désigner le chemin d'accès si le répertoire courant de l'unité est déjà actif. Exemple : "DIR C:" liste le contenu du répertoire courant dans le lecteur "C". En résumé, chaque unité logique a un répertoire actif.

4.7 Le chemin absolu d'un fichier

Exemple : Le répertoire racine du disque C se nomme **C:**
Exemple : La racine du disque courant se nomme \
Exemple : Le **fichier** demo.txt se nomme C:\windows\test\demo.txt
Exemple : Le **répertoire** test qui se trouve dans le répertoire windows du disque C se nomme C:\windows\test
Conclusion : Un chemin absolu commence par **\...** (racine de l'unité)

4.8 Le chemin relatif d'un fichier

Le chemin relatif d'un fichier tient compte du répertoire courant (Working Directory) du processus en cours (le terminal).
Exemple : **..\test\demo.txt**

4.9 Les commandes

Nous appellerons **commande interne**, l'ensemble des "mots" reconnus par l'interpréteur. Ils constituent le langage. Il est facile d'imaginer qu'un fichier ou qu'un répertoire puisse avoir le nom d'une commande interne. De ce fait, les commandes les plus indispensables pourront être écrites de deux façons différentes : en entier, ou en abrégé (ce dernier étant le plus courant, bien que le premier soit le plus portable dans l'ensemble des systèmes).
Une **commande externe**, est en réalité un programme fournit avec l'interpréteur. Il permet une évolution, voire une meilleure souplesse. Par exemple, la commande "*subst*" qui permet d'affecter un nom d'unité logique (lettre de lecteur) à un répertoire d'un disque, est en fait un programme exécutable.

4.10 L'invite de commande (ou le prompt)

Une fois l'interpréteur exécuté, il vous affiche "un invite" (prompt) représentant (dans la plupart des cas) le lecteur et le répertoire courant dans l'arborescence (unité et répertoire actuelle).
Exemple : soit le prompt suivant "C:\>" - Il signifie que vous êtes sur le disque dur "C:" à la racine (répertoire racine "\"). Cet invite change lorsque vous entrez ou sortez d'un répertoire et

aussi lorsque vous changez de disque. Vous pouvez modifier les caractéristiques de l'invite, en utilisant la commande interne "PROMPT".

Prompt PG
Ou
SET PROMPT = PG

5 Liste et rôle des commandes internes (MSDOS 6.22)

Commande	rôle
BREAK	Active ou désactive le contrôle étendu CTRL+C.
CALL	Appelle un fichier .BAT secondaire.
CD / CHDIR	Affiche ou modifie le nom du répertoire en cours.
CHCP	Affiche le numéro du jeu de caractères (page de codes) en cours.
CLS	Efface l'écran.
COPY	Copie ou concatène un ou plusieurs fichiers.
CTTY	Change le périphérique de commande du système.
DATE	Affiche/modifie la date du système.
DEL / ERASE	Efface le (ou les) fichier(s) spécifié(s) du disque.
DIR	Affiche les informations sur les fichiers et répertoires.
ECHO	Active et désactive l'affichage du texte des programmes de commandes.
EXIT	Quitte l'interpréteur de commandes et revient à son programme de lancement.
FOR	Applique une commande sur une liste d'objets.
GOTO	Poursuit l'exécution d'un programme de commandes au label spécifié.
IF	Exécute un traitement conditionnel dans des programmes de commande.
LH / LOADHIGH	Charge un programme en zone de mémoire supérieure.
MD / MKDIR	Crée un sous-répertoire.
PATH	Liste des répertoires de recherche des fichiers exécutables.
PAUSE	Suspend un programme de commandes jusqu'à l'appui d'une touche.
PROMPT	Modifie l'invite de l'interpréteur.
RD / RMDIR	Supprime un répertoire.
REM	Permet d'inclure commentaires dans un fichier de commande.
REN / RENAME	Renomme un ou plusieurs fichiers.
SET	Affiche, définit ou supprime des variables d'environnement.
SHIFT	Décalage des paramètres d'un fichier de commandes.

TIME	Affiche l'heure système et permet de la modifier.
TYPE	Affiche le contenu d'un fichier texte.
VER	Affiche le numéro de la version du DOS utilisée.
VERIFY	Affiche, active et désactive la vérification de bonne écriture sur disque.
VOL	Affiche le label d'un ou de plusieurs volumes.

6 Liste et rôle des commandes externes (MSDOS 6.22)

Commande	**rôle**
APPEND	Spécifie la localisation de fichier de donnés.
~~ATTRIB~~	~~Affiche ou modifie les attributs de fichier.~~
CHKDSK	Remplacé par SCANDISK.
CHOICE	Attend que l'utilisateur fasse son choix dans un menu.
COMMAND	C'est l'interpréteur lui-même.
COMP	Remplacer par FC.
DEBUG	Programme de mise au point des fichiers exécutables.
DELTREE	Destruction d'un répertoire avec l'ensemble de son contenue.
DISKCOMP	Compare le contenu de deux disquettes.
DISKCOPY	Copie le contenu de la disquette sur une disquette, formatée ou non.
~~EDIT~~	~~Éditeur de fichier texte ASCII.~~
FC	Compare deux fichiers et affiche les différences
FDISK	Programme de configuration des disques durs : à utiliser avec précaution.
FIND	Recherche une chaîne de texte dans un ou plusieurs fichiers.
HELP	Affiche l'aide
FORMAT	Formate le disque du lecteur spécifié.
KEYB	Définit le clavier pour une langue donnée.
LABEL	Créer/modifier ou supprimer un nom de volume (disque ou disquette).
MODE	Configure les périphériques du système.
MORE	Affiche un écran de données à la fois.
MOVE	Transfère de fichier ou renomme un répertoire.
NLSFUNC	Charge en mémoire les informations spécifiques à un pays.
PRINT	Imprime un fichier texte pendant que l'ordinateur reste utilisable.
SCANDISK	Crée et affiche un relevé d'état du disque et corrige les erreurs décelées.
SHARE	Installe le partage et le verrouillage de fichiers.

SORT	Filtre de tri de données par ordre alphanumérique.
SUBST	Affecte une lettre de lecteur au chemin d'accès spécifié.
SYS	Transfère les fichiers système sur un disque.
TREE	Affiche l'arborescence pour le répertoire ou le disque spécifié.
XCOPY	Copie des fichiers et des répertoires ainsi que leurs sous-répertoires.

7 Commandes pour le Système de Fichier

7.1 Changement de répertoire

Pour évoluer dans l'arborescence de vos disques, vous devez utiliser la commande interne "CHDIR "(change directory) ou "CD" en abrégé. Cela vous permet de changer de répertoire de travail (courant). Pour entrer dans un répertoire, il vous suffit de faire suivre la commande "CD" du nom du répertoire : "CD DOS" permet d'entrer dans le répertoire "DOS" si ce dernier existe. Votre "invite" (prompt) s'ajustera immédiatement. Pour sortir de ce répertoire, vous devez utiliser la command "CD **..**". Les deux points successifs désignent le répertoire parent. En fait, c'est comme si l'on entrait dans ce répertoire parent. Pour sortir directement de tous les répertoires dans lesquels on se trouve (pour aller à la racine), on saisira la commande suivante : "CD \".

```
CD  nom  pour entrer
CD  ..   pour sortir
CD  \    pour allez à la racine.
```

7.2 Visualisation du contenu d'un répertoire avec DIR

Pour visualiser le contenu d'un répertoire (la racine est le premier répertoire), utilisez la commande interne "DIR". Elle vous affiche les noms des fichiers et des répertoires contenus. De plus, chaque fichier est suivit de sa taille en octet, puis de sa date de modification. Cette information est très utile. La commande "DIR" informe aussi sur la place disponible restant sur le disque. Il est possible de visualiser le contenu d'un autre répertoire, en indiquant sa localisation : "DIR C:\DOS" permet de visualiser le contenu du répertoire "DOS" même si l'on se trouve dans un autre répertoire.

```
Le volume dans le lecteur C est GOWAP_C
Le numéro de série du volume est CAFE-CAFE
Répertoire de C:\DOS

.            <REP>       24/02/97  19:39
..           <REP>       24/02/97  19:39
ANSI     SYS         9 079  31/05/94   6:22
MEM      EXE        32 838  31/05/94   6:22
KEYB     COM        15 851  31/05/94   6:22
KEYBOARD SYS        34 599  31/05/94   6:22
 ... ...
HIMEM    SYS        29 216  31/05/94   6:22
COUNTRY  SYS        26 937  31/05/94   6:22
CHKDSK   EXE        12 456  31/05/94   6:22
       69 fichier(s)         4 952 650 octets
        2 répertoire(s)     949 233 408 octets libres
```

Le "<REP>" désigne un répertoire. "**.**" et "**..**" sont donc bien des répertoires.

7.3 Les options de la commande DIR

La commande interne "DIR" possède beaucoup de paramètres. Les options les plus utilisées sont les suivantes : "/W", "/P", "/S". Elles permettent respectivement d'afficher la liste des fichiers et des répertoires sur plusieurs colonnes, par écran, en affichant le contenu des sous-répertoires.
Pour plus de détail, tapez "DIR /?".

7.4 Le changement d'unité (lecteur)

Il est souvent plus facile de travailler sur un lecteur (une unité logique) lorsqu'il est actif (le lecteur courant). Cela permet de simplifier (minimiser) la frappe. Pour changer d'unité logique, il suffit simplement de la désigner : "**D:**" permet d'activer le lecteur "D" comme unité courante. Dès lors, nous pouvons visualiser son contenu sans pour cela rappeler l'unité : "**DIR**" au lieu de "**DIR D:**".

7.5 Classement et gestion des répertoires

Un répertoire, c'est pour moi la possibilité de ranger dans un même emplacement, des fichiers de même nature, ou couvrant le même sujet. Ainsi, nous pouvons envisager de créer un répertoire dans lequel nous placerons l'ensemble de nos fichiers images. Pour créer ce répertoire, nous devons utiliser la commande interne "MKDIR" (Make Directory), ou "MD" en abrégé, suivit d'un nom de répertoire. "MD C:\IMAGES" crée le répertoire "images" dans la racine de l'unité "C" et "MD IMAGES" crée ce répertoire dans le répertoire courant. Nous pouvons maintenant nous déplacer dans ce répertoire.
Pour qu'un disque soit aisément consultable, il faut rapidement prendre l'habitude de le classer par catégorie. Ainsi, je vous propose de créer un répertoire "JEUX" dans lequel vous y placerez l'ensemble de vos jeux ; de la même manière, vous créerez un répertoire "Document" (Ce que fait *Windows* avec le répertoire "*Mes Documents*"). Pour chaque jeu, je vous conseille de créer un répertoire dans le dossier "jeux", avec un nom significatif ; à défaut d'un thème, utilisez le nom du jeu. Ainsi structuré, il sera rapide de supprimer un jeu, en détruisant simplement son répertoire, cela évitant tout danger si le nom du répertoire est suffisamment suggestif.
Pour détruire un répertoire avec la commande "RMDIR" (Remove Directory), "RD" en abrégé, vous devez préalablement effacer son contenu (les fichiers et les sous-répertoires). Vous pouvez aussi utiliser la commande externe "DELTREE", qui effacera en même temps l'ensemble des éléments contenus dans le répertoire.

7.6 Gestion des fichiers : créer, copier, effacer, déplacer, renommer

7.6.1 Créer

Gérer des fichiers, c'est pouvoir les créer, les déplacer, les copier, et même les effacer. Toutes ces tâches peuvent être (dans une certaine restriction) effectuées avec l'interpréteur CMD. Nous pouvons créer des fichiers avec le programme EDIT, aisément assimilable à une commande externe, car il est quasiment indispensable. On peut aussi créer rapidement un fichier texte ASCII avec les filtres de CMD, mais cela est plus complexe à comprendre et à expliquer. Un exemple tout de même : essayez "COPY CON**:** T.TXT". Tapez tout ce qui vous passe par la tête, et terminez en validant un CTRL-Z. Vous avez créé un fichier nommé "T.TXT" contenant ce que vous venez de taper. En fait, le clavier est assimilable à un fichier nommé "CON**:**". Pour résumer, nous avons copié un fichier clavier en un fichier "T.TXT".

7.6.2 Copier

La commande interne "COPY" permet de copier des fichiers. Pour cela, vous devez désigner un fichier et une destination. Cette commande peut être dangereuse si elle est mal utilisée. En effet, vous pouvez envisager d'écraser un autre fichier portant le même nom, ou même un autre nom si vous avez désigné une destination avec un nom de fichier. Pour exemple, copions le fichier "CC.BAT" dans le répertoire "DOS". Cela nous donne :

"COPY C:\CC.BAT C:\DOS"

Le premier paramètre de notre commande "COPY" désigne le fichier à copier. Le second désigne le répertoire de destination. Si ce répertoire ne devait pas exister, le fichier sera copié en un nouveau fichier nommé "DOS". Les fichiers "AUTOEXEC.BAT" et "DOS" seront les mêmes au nom de fichier près. Pour une information plus précise sur la commande "COPY", je vous propose de saisir la ligne suivante : "HELP COPY" ou "COPY /?"
Voici néanmoins la syntaxe de la commande "COPY"

```
COPY [/A¦/B] source [/A¦/B] [+ source [/A¦/B] [+ ...]][destination [/A¦/B]] [/V]
```

Source et destination peuvent être un nom d'unité, un répertoire, un nom (ou masque) de fichier ou une combinaison de ces éléments.
Les commutateur /A et /B désigne l'état des fichiers à copier. Il s'applique à la spécification du fichier les précédant et restent actifs jusqu'à ce qu'un autre commutateur soit rencontré. Dans le cas où le commutateur est déclarer en premier, il agit simplement sur les fichiers le succédant.
/A : fichier texte : la copie du fichier s'arrête dès qu'un caractère CTRL-Z (27, 1Ah) est trouvé. Ce caractère n'est pas copié. Pour le fichier destinataire, un CTRL-Z est ajouté à la fin du fichier.
/B : fichier binaire : le fichier est copie entièrement.
/V : demande une vérification de la copie.

7.6.3 Effacer

Pour effacer un fichier, nous utiliserons la commande "ERASE", ou "DEL" abrégé de '*delete*', suivit du nom de fichier.

7.6.4 Déplacer

Pour déplacer un fichier, nous pouvons utiliser la commande interne "COPY" suivit de la commande interne "DEL". Mais il sera plus simple d'utiliser la commande externe "MOVE". Cette commande prend le même style de paramètres que la commande "COPY".

7.6.5 Renommer

Il peut aussi être intéressant de renommer un fichier. Nous avons à cet effet, la commande interne "RENAME", "REN" en abrégé. Pour renommer un répertoire, nous utiliserons la commande externe "MOVE". Dans d'autres interpréteurs, il est souvent possible d'utiliser aussi la commande "REN", ce qui semble plus intelligent.

7.6.6 Masque, caractère générique, joker (wildcard)

Lorsque l'on désire faire référence à plusieurs fichiers à la fois, il est possible d'utiliser les caractères génériques (joker) point d'interrogation et étoile : "?" et "*". Un point d'interrogation permet de remplacer tous les caractères à l'endroit où il est placé, alors que l'étoile remplace tous les caractères suivant sa position. Comprenons par l'exemple. Si je désire lister l'ensemble des fichiers portant un nom commençant par la lettre "C", je vais saisir la ligne suivante : "DIR C*.*". En fait, l'interpréteur va remplacer les étoiles par autant de points d'interrogation qu'il peut mettre pour compléter un nom de fichier. C'est-à-dire, que "C*.*" sera transformé en "C???????.???". Comme les points d'interrogation sont des caractères génériques, ils peuvent être remplacés par n'importe quel autre caractère. Ainsi, la commande "DIR" affichera tous les fichiers ayant un "C" en première position. Par exemple :

"CHKDSK.EXE, CHOICE.COM, COUNTRY.SYS". Autre exemple : "DIR MO?E.C*", affichera "MODE.COM, MORE.COM".

8 Les programmes

8.1 Lancer un programme

L'interpréteur ne permet pas seulement de gérer les fichiers, il permet aussi d'exécuter des programmes. Ces derniers sont des fichiers dont l'extension (et l'ordre de priorité) est "COM", "EXE" ou "BAT". Il est intéressant de remarquer que les commandes externes sont en fait des programmes, car se sont des fichiers portant l'une de ces extensions. Nous pouvons donc exécuter un jeu, ou un utilitaire, en tapant simplement le nom du fichier.
Nous omettrons volontiers l'extension de ce programme, à une exception près : si nous avons par exemple des programmes "GO.COM" et "GO.EXE" dans le même répertoire, l'interpréteur exécutera le "GO.COM" si nous omettons l'extension. Ceci provenant de l'ordre de priorité. Pour exécuter le "GO.EXE", il faudra l'écrire de toutes lettres.
L'interpréteur peut interpréter des fichiers de commandes dont l'extension est "BAT". Ces derniers sont en fait un regroupement de commandes dans un même fichier.

8.2 Variable PATH et la recherche de programme

L'interpréteur peut exécuter des commandes externes qui sont en réalité des programmes. Ces derniers ne sont pas forcément dans le répertoire courant. En réfléchissant, il n'y a aucun intérêt d'obliger l'utilisateur de stocker ces programmes dans un répertoire unique. De ce fait, l'interpréteur propose une variable dans laquelle l'utilisateur y placera les chemins des recherches dans l'ordre de lecture séparés par un point virgule. Cette variable porte le nom "PATH".
"PATH" est aussi une commande interne. En faite, cette commande affecte la variable "PATH".
Exemples :

```
PATH  %PATH%;C:\BATCH;C:\DOS
        ou
SET  PATH =%PATH%;C:\BATCH;C:\DOS
```

9 Quelques variables de l'environnement

Avec la commande "SET", nous pouvons redéfinir les variables utilisées par l'interpréteur CMD.

9.1 La commande set

Cette commande permet de définir et d'initialiser des variables globales. Personnellement, je l'utilise pour définir les variables suivantes :

```
SET | MORE
SET TEMP = C:\TEMP
SET TMP = C:\TEMP
SET PROMPT = $P$G
SET PATH =%PATH%;C:\BATCH;C:\DOS
```

9.2 Répertoire temporaire

Si vous créez un répertoire "TEMP" sur votre disque, vous pouvez le déclarer comme étant le répertoire de travail temporaire du DOS pour l'ensemble des programmes utilisant des fichiers de données temporaires, en affectant la variable "TEMP" du chemin vers ce répertoire. Dans les anciens DOS, elle s'appelait "TMP". Pour nous assurer que l'ensemble des programmes fonctionnent correctement, nous déclarerons les deux variables.

```
SET TEMP = c:\temp
SET TMP = c:\temp
```

9.3 L'invite de l'interpréteur ou prompt

L'invite fait référence à la variable "PROMPT". Il est donc possible de l'initialiser dans le fichier de configuration. En utilisant des majuscules dans sa définition, l'invite est portable dans d'autres interpréteurs.

```
SET  PROMPT = $P$G
```

9.4 Listage

La commande "DIR" permet de liste le contenue d'un répertoire. Nous pouvons dès la version 6 définir les paramètres de cette commande à défaut de spécification. Nous utiliserons la variable "DIRCMD" :

```
set dircmd = /P/W
```

10 Le fichier BATCH

Un fichier BATCH, ou **fichier de commande**, ou script, est un regroupement de commandes. Dans un fichier BATCH, nous pouvons utiliser des commandes qui seraient inutiles en saisie directe. C'est le cas des commandes conditionnelles, d'affichage, d'attente, de branchement et aussi de boucle. Un fichier BATCH (*.BAT) peut être exécuté comme une commande externe, avec des paramètres, si vous en avez l'utilité. **Dans un explorateur de fichiers, si on clique sur un fichier batch, alors le process a comme répertoire courant le répertoire ou se trouve le fichier BAT**.
Les quelques paragraphe qui suivent, listerons les commandes internes relatif au programme BATCH : "Echo", "For", "Goto", "If", "Pause", "Rem", "Shift", "Call", "Break" ...

10.1 Création

Pour créer un fichier BATCH, je vous conseille d'utiliser le programme notepad++. Exemple commenté d'un fichier de commande :

```
@Echo off
Cls
Echo Coucou, c'est moi.
Pause
Ver
```

La première ligne est pour moi indispensable. Elle indique à l'interpréteur de ne pas afficher sur l'écran les commandes qui vont être exécutées. Cela permet une transparence du traitement. Vous pouvez l'enlever ou bien la mettre en remarque (ce qui revient au même) en insérant "REM" devant "@Echo off" pour rétablir l'affichage des commande exécuter. Je pense que vous devriez essayer pour comprendre le phénomène. La deuxième ligne efface l'écran ; rien d'extraordinaire. Ensuite, nous affichons un message à l'écran avec la commande "ECHO" puis nous attendons l'appui d'une touche. Et pour terminer, nous affichons la version. Moralité, ce programme (fichier de commande) n'a aucune utilité autre que d'être un exemple.

10.2 Echo, et @

La commande "Echo" permet d'affiche à l'écran une ligne de texte. Je dis "une ligne", car une fois le texte affiché, il y a automatiquement un retour à la ligne : c'est-à-dire, que deux commande "echo" à la suite, afficherons deux lignes l'une en dessous de l'autre. La commande "echo" permet aussi d'informer interpréteur de commande, que nous ne désirons pas qu'il affiche les lignes du programme BATCH. La commande "@" permet d'informer l'interpréteur que la ligne ne doit pas être affiché à l'écran, même si la commande "echo" est sur "on".
Syntaxe : "Echo off" pour ne pas affiché les lignes du fichier de commande. La commande "echo" est sur "on" par défaut.

10.3 REM, Remarque

Dans un fichier de commande, vous pouvez insérer des commentaires, grâce à la commande interne "REM" ou « **;** » .

10.4 Break

La commande interne "Break" autorise (ou non) l'utilisateur d'interrompre un fichier de commande avec les touches d'interruption "CTRL-C" ou "CTRL-BREAK".
Syntaxe : "Break ON/OFF"

10.5 Les arguments (paramètres)

Un fichier BATCH peut avoir plein de paramètres, à concurrence de 127 caractères sur la ligne de saisie. Dans le fichier, nous ne pouvons accéder qu'à dix paramètres à la fois avec les variables argument %0, %1, %2, %3, ... , %9. La variable %0 représente à l'origine, le nom du fichier BATCH, ou plus précisément la saisie au prompt du nom avec le chemin tapé. Pour atteindre les autres paramètres, nous devrons utiliser la commande interne "SHIFT", qui permet de cycler les paramètres. Ainsi, le contenu de la variable %1 sera positionné dans la variable %0, ..., le contenu de %9 dans %8, et ainsi, le paramètre suivant qui n'était pas accessible sera placé dans %9.

10.6 Les variables

Avec la commande interne "SET", vous pouvez définir et affecter des variables textes. Ces variables sont conservées tant qu'elles ne sont pas effacées, même après la fin d'un fichier BATCH. Ce ne sont pas des variables locales au fichier de commande, mais bien des variables globales. Exemple : "SET V=M??E.*" définit et affecte la variable texte "V" de la valeur "M??E.*". Pour changer de valeur, il suffit d'affecter de nouveau la variable : "SET V=LO*.*". Puis, pour effacer la variable, il suffira d'affecter une valeur vide : "SET V=".

Une fois cette variable définie, nous pouvons accéder à son contenu en utilisant son nom entre deux caractères pour-cent : "Echo %V%" affiche le contenu de la variable "V".
Je vous conseille d'effacer vos variables dès que vous n'en avez plus besoin.
Attention. Il existe d'autres variables qui sont utilisées en locale ; elles sont du format "%i" ou "%%i", cela dépend de interpréteur, et de sa version. On les trouve dans les boucles "FOR" par exemple (défini plus loin). J'ai pris "i" comme exemple ; vous pouvez utilisez un autre nom.

10.7 Label / branchement

Un label est un marqueur désignant une ligne dans un fichier de commande. Un label est utilisé de paire avec la commande de branchement "GOTO". Un label commence par un caractère deux points, suivit d'un nom. Exemple :

```
@Echo Off
... ...
Goto L1
... ...
:L1
... ...
```

Lorsque l'interpréteur rencontre la commande "GOTO L1", il poursuit sont interprétation après la ligne désignée par le label (il saute au label).

10.8 Test

Un test, c'est la possibilité d'exécuter un traitement si le résultat comparant deux éléments est vérifié. Il est engendré par la commande "IF" dont voici ces paramètres et sa structure :

```
IF [NOT] ERRORLEVEL nombre commande
IF [NOT] chaîne1==chaîne2 commande
IF [NOT] EXIST fichier commande
```

NOT	Spécifie que interpréteur exécute la commande seulement si la condition est fausse
ERRORLEVEL nombre	Condition vraie si le dernier programme exécuté a renvoyé un

	code de sortie supérieur ou égal au nombre spécifié.
Commande	Commande à exécuter si la condition est satisfaite.
Chaîne1==chaîne2	Condition vraie si les chaînes spécifiées concordent.
EXIST fichier	Condition vraie si le fichier spécifié existe.

Exemple de code :

```
@echo off
IF "%1"=="" GOTO Err
Echo Le paramètre est : "%1"
Goto Fin
:Err
Echo Entrez un paramètre.
:Fin
```

10.9 Boucle

Une boucle, c'est la possibilité de répéter plusieurs fois une opération sur un élément d'un ensemble définit. Souvent, l'ensemble est défini avec les noms de fichier, mais il peut aussi être du texte, voir des paramètres. Cette boucle est engendrée par la commande "FOR" dont voici la syntaxe :

FOR %variable IN (ensemble) DO commande [paramètre]

%variable	Paramètre remplaçable.
(ensemble)	Ensemble de fichiers séparés par des espaces. Caractères génériques permis.
Commande	Commande à exécuter pour chaque fichier.
paramètre	Paramètres ou commutateurs pour la commande spécifiée.

Pour utiliser FOR dans les fichiers de commandes, spécifiez %%variable au lieu de %variable.

Exemple compliqué : recherche la localisation de lui-même (le fichier BATCH) sur les disques du système. Nous admettrons que l'utilisateur saisisse au prompt seulement le nom du fichier de commandes.

```
Set DD_OLI=
For %%i in (C D E F G) do If "%DD_OLI%"=="" If exist %%i:\BATCH\%0.BAT SET DD_OLI=%%i
```

J'initialise à vide la variable DD_OLI, au cas où elle existerait déjà. Puis, je recherche successivement dans les disques durs la présence du fichier de commande (moi-même, vu par le fichier). Dès que je me trouve, j'affecte la variable DD_OLI du nom de l'unité dans lequel je me suis trouvé. — *Il est possible que je trouve un fichier portant le même nom que moi, sans pour cela que se soit moi ! Certes, mais sur mon propre ordinateur, j'évite autant que je peux, d'avoir des fichiers de commande portant le même nom. Pourquoi rechercher la localisation du fichier de commande ? parce que ce disque est amovible, et qu'il peut porter n'importe quelle lettre de lecteur, lorsqu'il est inséré dans un autre ordinateur.* - Une fois le fichier de commande trouvé, la variable DD_OLI est affecté. Dès lors, le test d'existence ne sera plus effectué car DD_OLI n'est plus vide. Comme il n'est pas possible de quitter la boucle, j'utilise un test sur le contenu de la variable. S'il était possible de quitter la boucle, j'aurai plutôt écris un truc du genre :

```
For %%i in (C D E F G) do If exist %%i:\BATCH\%0.BAT GOTO Trouvé
```

10.10Call

Dans un fichier BATCH, on peut désirer faire traiter une opération par un autre fichier de commande déjà existant. Pour cela, nous utiliserons la commande interne "CALL" suivie du nom du fichier de commande. Si nous omettons cette commande, le fichier BATCH ainsi appelé interrompt automatiquement l'exécution de l'appelant. C'est-à-dire, que la commande "CALL" informe l'interpréteur d'exécuter un BATCH comme étant un sous-programme. Sinon, l'appelle sera interprété comme étant un branchement conditionnel ou un saut (goto) dans un nouveau programme.

10.11Pause

La commande "PAUSE" affiche un message à l'écran, jusqu'à ce que l'utilisateur appui sur une touche. Il est possible de ne pas affiché le message, en le détournant avec un filtre, vers une sortie nulle : "PAUSE >NUL", attends l'appui d'une touche, sans afficher le message. L'information sur ce sujet sera donnée dans le paragraphe des filtres.

10.12Un exemple utile

Ajouter un chemin à la variable "PATH", pour la recherche de programme exécutable.

```
@echo off
If "%1"="" GOTO Err
Set PATH=%1;%PATH%
Goto Fin
:Err
Echo %0 [nouveau chemin de recherche]
:Fin
```

11 Désignation des périphériques, et spécificité

11.1 Re-Direction entrée/sortie

En standard, CMD considère que les entrées proviennent du clavier et que les sorties sont dirigées vers l'écran. Le clavier est donc l'entrée standard et l'écran la sortie standard. Vous pouvez modifier les entrées et sorties standard en utilisant respectivement les symboles "<" et ">" sur la ligne de commande.
Pour obtenir l'entrée standard à partir d'un fichier : "... < nom_fichier"
Pour rediriger la sortie standard dans un fichier : "... > nom_fichier"
De plus, il y a le symbole "|" (touche AltGr+6) qui permet de rediriger la sortir standard d'un programme "prog1" vers l'entrée un programme "prog2 ": "prog1 | prog2".

11.2 Le tube (ou filtre)

En anglais « pipe », symbole : **|**
Grâce aux redirections, nous pouvons faire appelle aux programmes "MORE", "SORT" et « FIND ». "MORE" permet d'afficher des données en marquant une pause après chaque écran, et "SORT" de trier les données par ordre alphabétique croissant ou décroissant. Ainsi, nous pouvons filtrer une commande.
Exemple : "DIR | SORT /r" affiche le contenu du répertoire dans l'ordre alphabétique décroissant.
Exemple : set | more
Exemple : set | find “_HOME”
Exemple : netstat -an|find "TCP"

11.3 Fichiers matériels, nom de périphérique : CON, NUL, PRN, LPTn, AUX, COMn

Avec la redirection, nous pouvons admettre que le clavier et l'écran sont respectivement des fichiers de lecture et d'écriture. De ce fait, le CMD les nomme "console" ou terminal.
La commande "COPY CON**:** T**.**TXT" permettra de créer le fichier "T**.**TXT" avec des données saisies au clavier, terminées par un "Ctrl-Z" ou "F6". Puis "COPY T**.**TXT CON**:**" enverra le contenu du fichier vers l'écran. Le caractère deux points est facultatif, mais il permet de faire remarquer que nous désignons un fichier matériel, et non un fichier physique.

CMD nous offre une redirection nommé "NUL", en écriture seule. On l'utilisera pour rediriger et ignorer les sorties standard.
La commande "PAUSE > Nul" permet de faire une pause sans afficher le message. *Cette démarche est utile dans un fichier de commande, dans le cas ou le système d'exploitation ne serait pas dans la même langue*.
Nous pouvons designer l'imprimante (ou tous autre sortir sur port parallèle) avec le nom matériel "PRN". Mais "PRN" représente l'imprimante par défaut. Il est possible que vous utilisiez plusieurs imprimantes. De ce fait, vous pouvez la désigner avec le nom "LPT" suivit du numéro du port parallèle affecté : "LPT1**:**" pour l'imprimante sur le port parallèle numéro un. "LPT2**:**" et "LPT3**:**" respectivement pour les ports deux et trois.
Nous pouvons aussi recevoir ou émettre sur les ports de communications séries avec le nom matériel "AUX" et "COMn" où "n" représente le numéro du port (de 1 à 4).

RESTE

En fait, le DOS n'est pas accessible directement à l'utilisateur. C'est un ensemble de routines matérielles et logicielles gérant les différents éléments de l'ordinateur. Ces routines sont regroupées en catégories nommées interruptions. Il n'est pas nécessaire pour un utilisateur d'en savoir davantage. Ce dernier ne pouvant accéder de lui-même à ces routines (fort heureusement pour lui, car cela serait fastidieux).
Un programme "*Interpréteur de commande* " **CMD** .EXE (COMMAND.COM sur les anciennes versions) a été conçu pour palier à ce problème.