Windows Command Line Processor (WCL)

WCL

WCL is a command line processor and interpreter, batch processor, file manager, applications manager, scheduler, scripting tool, and suite of utilities, for Microsoft Windows 3.1x, and IBM Win-OS/2. This program can either supplement, or completely replace the combination of the File Manager, Program Manager, and the MS-DOS prompt.

Internal commands and features available in WCL are described under the different headings listed below. A proper introduction to WCL is contained within the "INTRODUCTION" section below. If you are new to WCL please read that section first.

Many thanks to those who have registered WCL. If you have not registered your copy, please consider doing so.

REGISTRATION FEE:

Standard Registration = £18.00 (U.K.) or $26 (U.S.), or equivalent; please see the "REGISTRATION" section below for the fee if paid in other currencies, and for details of the extra charges for other types of registration).
Please see the "REGISTRATION SITES" section for your LOCAL REGISTRATION SITE.

Alphabetical List of Commands

Below is an alphabetical list of internal commands and features supported by WCL. They can all be found grouped under the various services listed in the index screen.

Introduction

WCL contains too many features and commands to be fully summarised here. Please read this help file carefully.

Windows Command Line (WCL) is a command line interface program for Windows 3.1x and Win-OS/2. The program simulates the infamous C:\> prompt of the DOS command line, but from within Windows, or while running as a "seamless" Windows application on the OS/2 Workplace Shell desktop. This is useful for those DOS/Unix hackers who find themselves having to use Windows for certain applications, or for people who want a very quick and easy way to multi-task Windows programs, either within Windows itself, or from the OS/2 desktop, or for those who like to have a command line window available at all times.

From WCL, you can run all Windows, DOS and OS/2 (under OS/2 2.1x) programs just by typing the program name, and pressing <ENTER> as you do would at the DOS prompt for DOS programs. When you run a program through WCL, the program's window becomes the Active Window. You can go back to WCL by clicking on any part of the WCL window that is visible to you, and then run other programs from there.

Those who are already familiar with command line shells (e.g. the Dos or OS/2 command shells) can start using WCL straight away, as they will find that the commands that they are used to (e.g. DIR, COPY, DEL, REN, MD, CD, TYPE, etc) will also work as expected under WCL. By design, WCL internal commands are similar to those of the DOS COMMAND.COM, and OS/2's CMD.EXE. You can later start to learn the WCL extensions and internal command set at your leisure.

With WCL, you can transmit multiple commands on one line, by separating each command with a semi-colon. You can also execute commands which are internal to your own DOS command interpreter (the "COMSPEC" variable) by using the RUNDOS or SPAWN commands (eg for the 4DOS "select copy" command, you can type "RUNDOS SELECT COPY [*.EXE] A:" - if 4DOS.COM is your COMSPEC, then this will load 4DOS.COM in a separate session, and pass to it your SELECT COPY command). Note that when using the RUNDOS command, you have to supply the full pathnames of the files that you want to operate on - otherwise, use the SPAWN command instead.

WCL consists of 2 main executables (1) WCL.EXE - the main executable (2) BIGWCL.EXE - the "big" version of WCL.EXE, in the sense that it's main window is big, and all the output is directed to that window.

WCL.EXE and BIGWCL.EXE are alternative forms of the main WCL program. There is nothing stopping you from using both of them, but the idea is that while some people will prefer WCL.EXE's small and unobtrusive main window, others may not like the fact that it uses popup windows for some of its output. The ONLY significant differences between WCL.EXE and BIGWCL.EXE lies in the size of their main windows, and the fact that all the output from BIGWCL.EXE is in the same main window. Note that WCL.EXE **needs** BIGWCL.EXE for much of its functionality.

PATH:

If the application you wish to run is not in a directory which is in DOS Path, you will have to supply the full path name (e.g. "C:\WPWIN\WPWIN", to run WordPerfect for Windows, if C:\WPWIN is not in the DOS Path). If the application is situated in a directory that is in the DOS Path, all you need do is type its name, and press <ENTER> (e.g. "WRITE" <ENTER>, to run Windows Write).

All Windows programs can be run from within WCL. This includes DOS programs for which a Windows .PIF file exists. Most DOS programs can also be run directly from WCL without creating a PIF file for them. In this case, they will run in full screen mode.

Note that most internal DOS commands (i.e, those that are resident in COMMAND.COM) can NOT be run directly from WCL (except by using the RUNDOS or SPAWN commands). However, a number of DOS-like commands are supported through built-in technology. Below is a list of some of them;

[1] CD or CHDIR   (change directory)

[2] MD or MKDIR   (create a new directory)

[3] RD or RMDIR   (remove/delete a directory)

[4] DEL or ERASE   (delete files. Wild cards are accepted)

[5] REN or RENAME (rename a file or a directory)

[6] COPY (copy files. Wild cards are accepted).

[7] TIME (show current system time)

[8] DATE (show current system date)

[9] SET (show SOME enviroment variables)

[10] PROMPT (Change the WCL prompt)

[11] TYPE or MORE   (display the contents of an ASCII file)

[12] PRINT (print a file)

[13] DIR (list the files in the directory)

Apart from changing drives (e.g. "A:" to change to drive A or "D:" to change to drive D, etc.,) INTERNAL DOS commands different from those listed above cannot be directly run from WCL. Attempting to run them will either produce an error message from Windows, or lead to the DOS prompt being invoked through a DOS Shell.

External DOS commands (i.e, those which have their own .EXE, .COM, or .BAT files, e.g "FORMAT", "GWBASIC", "XCOPY", etc.) can normally be run directly from WCL. However, I would not attempt to run programs such as "CHKDSK" or programs which access the hardware directly (such as disk compressors) when in Windows. A lot of grief can result from this. Basically, any DOS program which can be used safely under Windows can be used safely in WCL since everything that WCL does is done through Windows API calls (i.e. Windows itself does all the actual processing. WCL only acts as a command line interface between you and Windows).

NOTE: the command "DO"

With regard to the way WCL runs external programs, some users have complained about the fact that WCL sometimes changes to the directory in which the executable for the program was found, before running it. This feature exists for many reasons (partly to do with Windows itself), but it can be circumvented. I have decided to keep that feature as the default, but to permit users to circumvent it. This is implemented when the user types "DO" before the name of the program to be executed.

If this is typed before the name of an external program then WCL will remain in the current directory while running that program, and will not change to the program's directory as it sometimes does. The only use of this command is as a prefix to whatever you would normally have typed. Thus if you would normally type;

```
PKUNZIP WINCMD*.ZIP
```

you would now type;

```
DO PKUNZIP WINCMD*.ZIP
```

Most people will never need to use this "DO" command, but it is there anyway.

You can dispense with having to type "DO" all the time by setting the "DO=" line in WCL.INI to "1", or "ON" (i.e., "DO=1"). With this setting, WCL will behave as if you have typed "DO" before the name of *every* program that you want to execute. This setting is NEW (with version 7.0), and is disabled by default.

If you still have problems with a DOS program and "DO" does not fix the problem, then try the "RUNDOS" command.

If there is any program which you should not run under Windows, then please do NOT attempt to run it via WCL.

Note that you can use the UNIX names of some of these commands, although they do not operate like the UNIX commands.

e.g.

CP for COPY

MV for RENAME

RM for DELETE

CWD for CHANGE DIRECTORY

LS for DIRectory listing

These commands operate more or less like their DOS equivalents. For file copying, wildcards are accepted for SOURCE file specifications only. You cannot use wildcards in TARGET file specifications.

e.g

```
COPY *.DOC A:\MSDOS   - This is valid
COPY *.DOC A:\MSDOS\*.BAK   - This is invalid.
```

Both the COPY and DIR commands produce their own Windows on the screen. However, if you are running BIGWCL.EXE, all the output is in the main window. BIGWCL.EXE is almost an exact reproduction of the MSDOS prompt, within Windows/Win-OS/2.

See also;
Alphabetical_List
Directory_Services
File_Services
System_Services
Miscellaneous_Services
Command_Line_History
Batch_Files
DOS_Programs

Command_Line_History

WCL supports a limited form of command line history by keeping a record of the LAST 30 commands typed at the prompt.   There are a number of commands for accessing the history function. They are enumerated below;

1.   !! - (two exclamation marks) - this will execute the most recent command.

2.   ! -   (one exclamation mark) - If this is typed by itself, WCL will list the last 20 commands (each of them with a number) in a message box. When the message box is closed you are prompted for the number of the line that you want to execute. The command is then executed.

   If you do not want to execute any of the listed commands, type 0 (zero) or just press <ENTER>   If the single exclamation mark is followed by a space and then a number (e.g., ! 10), WCL fetches the command with that number (if any exists). Thus for example, "! 6" means fetch the sixth to the last command. You can use a hyphen instead of a space (e.g., "!-6")

3.   LIST - Show a numbered listing of the last 20 commands typed at the WCL prompt (alternative command is HISTORY).

4.   CLEAR - Clear the command line history list. This gets rid of all the entries present on the list of the last 20 commands. The list will then start to build from the scratch. It is a great way to stop prying eyes (e.g., the boss) from seeing what commands you have been typing all day. This command can take an optional parameter - the number of the item to clear (e.g. "CLEAR 5"   - to delete item 5 from the list). If no parameter is supplied, then ALL the items in the list are cleared.

DOSKEY EMULATION

WCL features support for a LIMITED emulation of the DOSKEY function of scrolling through a list of past commands (by using the arrow keys - LEFT or UP arrow keys, or CTRL-Z for ("up") and DOWN or RIGHT arrow keys, or CTRL-X (for "down")).

This feature is enabled by a setting "EMULATE-DOSKEY" in WCL.INI. A setting of "1" or "ON" enables this feature, and any other setting disables it (eg "EMULATE-DOSKEY=ON"). It can also be enabled or disabled temporarily within WCL by typing "DOSKEY ON" or "DOSKEY OFF".

One limitation is that you can only edit the commands by using the BACKSPACE key, unless the command-line editing function is turned on (see below).

FULL COMMAND-LINE EDITING

WCL now supports full command-line editing. This feature is optional, but it is turned on by default in WCL.INI (with the setting "LINE-EDITOR="). You can turn it off permanently by putting a "0" or "OFF" on that line. You can also turn command-line editing off and on temporarily by using the new command "LINE-EDITOR ON" (to turn it on) or "LINE-EDITOR OFF" (to turn it off).

If DOSKEY emulation is not enabled, command-line editing will not work. When command-line editing is enabled,   you can use the left and right arrow keys, Backspace, Delete, Ins, Home, and End, to move about and edit the commands being entered, and also to edit the commands that have been brought up through DOSKEY scrolling. line editing function.

When command-line editing is enabled (the default setting in WCL.INI), the text being typed at the command line will scroll horizontally on the same line, and you can move left or right with the cursor keys. If the end of the prompt is less than 20 characters from the end of the WCL window, the cursor will move to the next line to allow more room for the commands to be typed.

When command-line editing is turned off, then the text being typed will wrap onto the next line. In this case, you cannot go back to the previous line.

Windows Batch Files and Scripts

WCL supports sequential processing of commands and the running of scripts by allowing you to put commands into a BATCH or SCRIPT file. Batch or script files must have **.CBF** as their extension. "CBF" stands for "Command Batch File".

e.g **COPYBAK.CBF**

WCL batch files and WCL scripts are basically the same thing - except that I use the term "script" to refer to the more complex batch files which use a lot of the batch and script commands. In this documentation, I will use the term "batch file" to include WCL scripts.

Batch files can contain any command that WCL supports - ie. internal WCL commands, DOS .BAT, .EXE and .COM programs, Windows programs and Windows PIF files for DOS programs, and OS/2 programs.

Batch files must be in ASCII format, and each command must be on a separate line. Each batch file can be up to **180 lines** - in this respect, **REM**, **$CONST**, and **$DEFINE** lines are **not** counted as part of the 180 lines). However, one .CBF file **cannot** contain a reference to another WCL .CBF file (except by using the **$INCLUDE** command).

Once set up, all you need to do is to type the name of the batch file at the WCL prompt. You do not need to type it's extension. With the example above, you only need to type "COPYBAK". WCL will then try to execute the commands in the file on a line-by-line basis. This may result in some interesting screen maneouvres as each program is given the input focus by Windows, and tries to display its messages and main window.

Remember that Windows programs do not have the whole PC to themselves, unlike DOS programs, so each Windows program will allow another to be immediately loaded after it, as soon as it sets up its main window. If the batch file contains a mixture of DOS and Windows programs, the screen maneuovres are yet more interesting. The import of this is that the processing of batch commands in Windows will not always be as you expect, if looked at from a DOS batch file point of view. This is due to the nature of Windows itself, and there isn't much that I can do about it. In the event, if you want to be sure that each command will be completed before the next one is executed, you should use the **EXECWAIT** command.

See also;
BATCH-FILE.MAX-LINES

Script and Batch File Commands

$CONST
CHGSTR
$DEFINE
DELAY
ECHO
$ELSE
EXECWAIT
GOTO
IF
$INCLUDE
PAUSE
REM

RETURN
WCL AUTOEXEC Batch File
Batch File Parameters
Sample Batch File

NOTES:

1. Many of the script and batch file commands in WCL are almost equivalent to constructs in a programming language. This means that writing a script or a batch file in WCL is often identical to writing a program. Therefore, please ensure that you DEBUG your scripts and batch files thoroughly, before running them. Untold damage can be caused by running a buggy script or batch file.

2. Because of the multi-tasking functions in Windows, the NORMAL use of batch commands in situations where things depend on commands being executed in a certain order is impossible. This is because there is no way of telling the order in which the commands in your WCL batch file will be executed, since they will NORMALLY be executed in rapid succession. - i.e., unless you use the EXECWAIT, PAUSE, or DELAY commands.

3. If you MUST run the commands in the WCL batch file in a certain order, then you MUST use the command "EXECWAIT" - for EXTERNAL programs and commands, and either "PAUSE" or "DELAY" for WCL INTERNAL commands. problem.

4. NOTE also that WCL treats .CBF batch files as INTERNAL commands. This means that .CBF files will be executed in preference to any program of the same name, even if that program is in the current directory and the .CBF file is not. Thus for example, MYPROG.CBF will be executed in preference to MYPROG.EXE or MYPROG.COM. If you want the EXE or COM file to be executed instead, you need to type its EXTENSION as well.

5. Please ensure that batch files do NOT have the same names as any DOS or Windows program file that you will call from the batch files. For example, if you will call KERMIT.EXE from you batch file, make sure that the batch file is not called KERMIT.CBF - if you do not heed this advice, you are SURE to get a SYSTEM CRASH when you try to run the batch file. Please note this warning.

$CONST or #CONST

This is part of **the scripting features of WCL.** This command is used to define some **GLOBAL constants** in a WCL batch file. The defined constants then apply throughout the batch file. Wherever WCL encounters the defined constant in the batch file, it is replaced by the value which you assigned to it. Such constants should be defined **at the beginning of the batch files.**

This command is similar to the $DEFINE command, but is different in a very important respect - with $DEFINE, parts of any word that matches will be replaced - however, with $CONST, **only whole words will be replaced.**

The changes are done in memory - so the physical contents of the batch file are unaltered.

Restrictions and features;

1. $CONST can only be used **ONCE** for any particular constant in a batch file.

2. Each $CONST entry must be on a line by itself.

3. You cannot use one $CONST constant in the definition of another one - but you can use a value defined in a $DEFINE constant in defining a $CONST value.

4. A maximum of 30 $CONST lines (**inclusive of any $DEFINE lines**) is allowed.

5. If typed from the command line it will produce an error message.

6. If you use the constant's name (or any part of its name) as part of its value, it will be taken as a literal value.

7. A value assigned to a constant with $DEFINE can be used in defining the value of a $CONST constant. **This is one important advantage over using $DEFINE.**

The syntax is:

$CONST <constant>   <=value>

Examples;

$CONST OLDDIR=C:\WINDOWS\WCL\V760

#CONST COMMAND=C:\4DOS\4DOS.EXE

Note that the "=" sign **MUST** be present, otherwise the define will be ignored.

See also;
$DEFINE
$INCLUDE

CHGSTR

This command is for use with the "IF" conditions, in WCL batch files. If the "IF" condition is satisfied, CHGSTR replaces ALL the occurences in the batch file being executed, of the string pointed to by the "$Variable" with the string returned by the "IF" condition.

So, for example, if CHGSTR is used with an "IF" condition involving a variable called "$DIR", and the condition returns "C:\TEST\BIN", all occurences of "$DIR" in the batch file are replaced with "C:\TEST\BIN", and the current copy of the batch file will be executed accordingly.

The changes are done in memory - so the physical contents of the batch file are unaltered.

Restrictions;

CHGSTR can only be used ONCE for any particular "$variable" in a batch file.

Each $variable used in this way MUST be entirely unique. This is because partial matches will be changed as well. Thus, for example, you cannot have one variable called "$DIR" and another one called "$DIREC", if you are going to use CHGSTR with "$DIR". This is because when occurences of "$DIR" are being changed, the places where those letters occur in "$DIREC" (i.e., the first 4 letters of "$DIREC") will be changed as well - and this is probably not what you want. Please note this point.

If typed from the command line it does nothing.

The syntax is exact, and must be followed correctly.

The syntax is:

CHGSTR @variable=$variable

i.e., you create a pointer to the $variable with "@". The "@" sign must replace "$" in the 1st occurence of the $variable, followed by "=" sign, followed by the full $variable itself

Example;

IF GETSTRING "Copy Files to: $dest" CHGSTR @dest=$dest

This allows the user to enter a string for where to copy files to. The string is kept in the variable called "$dest". CHGSTR then replaces ALL occurences of "$dest" in the batch file with the string entered by the user.

See also;
IF

$DEFINE or #DEFINE

This is part of **the scripting features of WCL.** This command is used to define some **GLOBAL constants** in a WCL batch file. The defined constants then apply throughout the batch file. Wherever WCL encounters the defined constant in the batch file, it is replaced by the value which you assigned to it. Such constants should be defined **at the beginning of the batch files.**

The changes are done in memory - so the physical contents of the batch file are unaltered.

Restrictions;

1. $DEFINE can only be used **ONCE** for any particular constant in a batch file.

2. Each define must be on a line by itself.

3. **You cannot use one defined constant in the definition of another one.**

A maximum of 30 $DEFINE lines (**inclusive of any $CONST lines**) is allowed.

5. If typed from the command line it will produce an error message.

6. **You cannot use the constant's name (or any part of its name) as part of its value - you will get a BIG crash. However, you can use the constant's name in a constant define with the $CONST command**

7. Each constant used in this way **MUST be entirely unique**. This is because partial matches will be changed as well. Thus, for example, you cannot have one constant called **DIR** and another one called **DIREC**, if you are going to use $DEFINE with **DIR**. This is because when occurences of **DIR** are being changed, the places where those letters occur in **DIREC** (i.e., the first 3 letters of **DIREC**) will be changed as well - and this is probably not what you want. Also, defining a constant as **COMMAND** and another one as **COMMAND2** or **MYCOMMAND** is not advisable, because they have **COMMAND** common to them all. **Please note this point**.

TIP: If you must use similar names, the best thing to do is to interpose another character after the FIRST character of the common parts. For example, you could have a constant called **COMMAND**, and others called **C1OMMAND, C2OMMAND, C3OMMAND, etc.**

FURTHER TIP: Most of these restrictions do not exist when you use the **$CONST** command. That command is extremely identical to this one, except that it only changes **whole words**.

The syntax is:

$DEFINE <constant> <=value>

Examples;

$DEFINE OLDDIR=C:\WINDOWS\WCL\V760

#DEFINE COMMAND=C:\4DOS\4DOS.EXE

The following examples are NOT allowed;

#DEFINE COPY=COPY *.* A:\

#DEFINE COMMAND=COMMAND.COM

#DEFINE COMM=C:\DOS\COMMAND.COM

This is because they represent an attempt to use the constants which are being defined (or part of the constants' names) in the values assigned to the constants. **This will either lead to an almighty CRASH, or to unpredictable and random results.**

Note that the "=" sign **MUST** be present, otherwise the define will be ignored.

See also;
$CONST
$INCLUDE

DELAY

This command is for use in running either EXTERNAL or INTERNAL commands through WCL batch files.   Put DELAY on the line following a command, and WCL will stop for a bit, and then continue with the batch execution. "DELAY" can take one parameter - the number of SECONDS that WCL should wait for.   If no parameter i supplied, WCL assumes a DELAY value of 1 second.

NOTE: there is an UPPER LIMIT of 300 (5 minutes) for this parameter. This number is arbitrary, and could be higher. However, I have decided to place it at this limit, as a safety margin. If enough people feel that I should remove this upper limit, then I will do so.

Examples;

DELAY	will wait for 1 second
DELAY 5	will wait for 5 seconds
DELAY 180	will wait for 180 seconds - 3 minutes
DELAY 1800	will wait for 300 seconds - 5 minutes

See also;
PAUSE
Batch_Files

ECHO

Commands in WCL batch files will be echoed on the screen before they are executed. If you want this behaviour to change, you need to put the line "ECHO OFF" at the beginning of each WCL .CBF batch file. You can turn the ECHO on again in the batch file by putting "ECHO ON" before the next command that you want to be echoed to the screen.

$ELSE

This is for use in WCL batch files, with the "IF" conditions, to specify an alternative course of action, if the "IF" condition is not true.

Example

IF ISDIR "C:\BIN $i" GOTO CONTINUE $ELSE GOTO QUIT

See also;
GOTO
IF

EXECWAIT

This command is for use in running EXTERNAL commands. What this command does is to cause WCL to WAIT until the program being run has terminated, before returning you to the command prompt, or, if running a WCL batch file, before running the next command in the batch file. This command can be abbreviated to "EW".

To use EXECWAIT, all you need to do is to put it before the command that you would normally type. It can however be set permanently in the "small" version of WCL by the ALWAYS-EXECWAIT.SMALL setting in WCL.INI, and for the "big" version, by the ALWAYS-EXECWAIT.BIG setting. These settings are turned OFF by default.

eg

```
EXECWAIT NOTEPAD TEST.TXT

EW WRITE MYDOC.WRI

EW PKUNZIP -V MYZIP.ZIP
```

This function does not work for running DOS sessions or OS/2 sessions under OS/2 - and I suppose it will not work under Windows NT either.

See also;
ALWAYS-EXECWAIT.BIG
ALWAYS-EXECWAIT.SMALL
Batch_Files

GOTO

This is used to jump to a LABEL in a WCL batch file. Processing of the batch file then continues from that LABEL onwards. The command takes only one parameter - and that is the name of the LABEL to jump to.

LABELs must be identified by their names, but preceded by a COLON. For example, if you have the statement "GOTO END" in a WCL batch file, there must be a line which reads ":END". When this line is found, WCL jumps there, and continues processing the batch file from there. If such a LABEL does not exist, then WCL just continues processing the batch file as normal (from the point where the GOTO was encountered). Such an event can often lead to unpredictable results - so when you use GOTO in a batch file, please ensure that you properly DEBUG your batch file.

$INCLUDE or #INCLUDE

This is part of **the scripting features of WCL.** This command is used to **read the contents of another WCL batch file** into a WCL batch file. The contents of the INCLUDEd batch file thereafter become part of the contents of the current batch file, from the line in which it was included (but any $DEFINE or $CONST constants defined in the included file apply **globally** in the current batch file).

Note that this is the only way in which one WCL batch file can refer to another WCL batch file, and that **this command MUST be used with great care. Unpredicable results will ensue from improper use of this command. You have been warned.**

The changes are done in memory - so the physical contents of the batch files are unaltered.

Restrictions and features;

1. An INCLUDEd file CANNOT itself use $INCLUDE to include another file - i.e., **nested INCLUDES are not allowed**. Failure to heed this warning WILL lead to unpredictable results.

2. Each $INCLUDE entry must be on a line by itself.

3. You can INCLUDE any number of other batch files into a particular batch file, **subject to point 4 below.**

4. The maximum of 30 $DEFINE and $CONST lines, and 180 lines in a batch file still apply - i.e., **the current batch file and ALL the files included into it must together not exceed these limits.**

5. If typed from the command line it will produce an error message.

6. The **FULL path name** of any INCLUDEd file must be supplied - otherwise WCL will only look for the file in the current directory.

7. INCLUDEd files can have any name and any extension - as long as they are ASCII files, and contain valid commands.

8. If the INCLUDEd file does not exist, it is simply ignored.

The syntax is:

$INCLUDE <filename>

Examples;

```
$INCLUDE BACKUP2.CBF

#INCLUDE CLEANUP.INC

$INCLUDE C:\WINDOWS\WCL\BACKUP\CLEANUP.INC
```

See also;
$CONST
$DEFINE

PAUSE

This command is for use in running either EXTERNAL or INTERNAL commands through WCL batch files. Put PAUSE on the line following a command, and then WCL will pause for a key press before executing the next line in the batch file.

See also;
DELAY
Batch_Files

REM

You can use "REM" in a WCL batch file. Any line that starts with a REM will not be executed. Note that there MUST be a space between the REM and any other text on the line.

e.g.,

```
REM This is a comment line!
```

RETURN

For use in WCL batch files. Anytime this command is encountered, processing of the batch file STOPS, and the user is returned to the WCL command prompt - so, this is a way to terminate a batch file very quickly. It is good for use with LABELS and GOTOs.

Example

IF ISDIR "C:\BIN $i" GOTO CONTINUE $ELSE RETURN

See also;
$ELSE
GOTO
IF

WCL AUTOEXEC Batch File

AUTOEXEC.CBF

WCL supports an "AUTOEXEC" batch file for Windows/Win-OS/2. You can put any commands that you wish to be executed every time WCL is started in a WCL batch file called AUTOEXEC.CBF. WCL will then run the commands in this file whenever WCL is started.

Batch File Parameters

You can pass parameters to your WCL batch files, just as in DOS. To do this, use "%1" as you would in DOS. This feature is still quite new, but it has worked well so far in my tests. You will have to experiment to see what works in this regard. Please use this only for the purpose of passing parameters to external programs. Do not use it for WCL's internal commands. Note: WCL can only process a maximum of 7 parameters in this way (i.e., %1, %2, and %3, etc., up to %7).

Sample Batch File

Example of a .CBF file's contents;

```
ECHO OFF
SAY     This is a test     .CBF file!
CD C:\WINDOWS
COPY *.INI A:\
DELAY 3
CD C:\DOCS\LETTERS
COPY *.LET A:\LETTERS
DELAY 2
SAY   I have finished the back ups
```

Comments;

This file starts by printing a message that it is a test .CBF file. It then backs up all the .INI files in the Windows directory, and all the .LET files in the C:\DOS\LETTERS directory. It finishes by telling you that its has completed the back ups. The last "SAY" message may have actually appeared BEFORE the operations which it claimed to have completed if there hadn't been a DELAY statement.

INI File Settings

There are various entries in the file WCL.INI which determine the way in which WCL works. Below is an explanation of the purpose of each of the settings, including the defaults;

Note that most of the settings described below can be changed and set in the WCL Configuration program WCLCFG.EXE. You may want to run that program instead of editing the WCL.INI file manually.

ALWAYS-EXECWAIT.BIG
ALWAYS-EXECWAIT.SMALL
BACK-UP-INI-FILES
BATCH-FILE.MAX-LINES
BIGWCL-DEFAULT-FONT
CONFIRM-OVERWRITES
DEFAULT-FONT
DESKTOP-TIMER-ALWAYS-VISIBLE
DESKTOP-TIMER.LEFT
DESKTOP-TIMER.TOP
DIR-WINDOW.LENGTH
DIRCMD
DO
EMULATE-DOSKEY
ENABLE-DOS-SERVER
EXPAND-PATHS
FILE-DESCRIPTIONS
INSERT-MODE
LINE-EDITOR
LOCATION-HORIZONTAL
LOCATION-HORIZONTAL.BIG
LOCATION-VERTICAL
LOCATION-VERTICAL.BIG
SAVE-DESKTOP
SAVE-WINDOW-PARMS
SEARCH-DRIVE
SHOW-RESOURCES
SHOW-TIMER
STARTUP1
TEXT-BACKGROUND
TEXT-COLOR
TIMER-ON-DESKTOP
TOPMOST-WINDOW
US-DATE-FORMAT
USER-MENUS.BIG
USER-MENUS.SMALL
WCL-PROMPT
WINDOW-BORDER
WINDOWHEIGHT
WINDOWHEIGHT.BIG
WINDOWLENGTH
WINDOWLENGTH.BIG
YIELD-TIMER-IN-OS2

IMPORTANT NOTES

[1] Most of these settings can be changed using WCL's configuration program WCLCFG.EXE. The best way to run this program is by typing "CFG" from the WCL prompt. This caused WCL to pass some parameters to the configuration program, and every time you save your changes, the configuration program sends an UPDATE command to WCL. This way, you can see most of the changes immediately.

[2] The responsiblity for suppying correct and sensible values for these window co-ordinates is TOTALLY YOURS. The default settings are quite adequate for most needs, and you can always re-locate and re-size the window using your mouse.

[3] If WCL cannot find the file WCL.INI at startup time, then the default values will apply - but WCL may not run correctly.

[4] The WCL.INI file is only read ONCE - when WCL is loaded. If you change any thing in the file, you will have to close and restart WCL for the changes to take effect. The only exception to this is when you make changes through the configuration program, by typing CFG at the WCL prompt.

[5] If you are experimenting with different settings for the WCL window, there is nothing to stop you from testing your settings on another copy of WCL. You can run another copy of WCL by typing "WCL" at the prompt. You will then see another copy loaded, and reflecting the window co-ordinates in the current version of WCL.INI. NOTE that if you have not changed the window coordinates, then the other copies of WCL will have their windows right on top of the current copies.

[6] If WCL is NOT your Windows shell, then the SAVE-DESKTOP setting in WCL.INI will be ignored.

ALWAYS-EXECWAIT.BIG

If this setting is turned ON, then the "big" version of WCL will always wait for any EXTERNAL program that is run from it to terminate, before returning to the prompt, or, if running in a WCL batch file, before running the next command in the batch file. This setting does NOT work for spawning OS/2 or DOS sessions when you are running under OS/2.

NOTE: commands scheduled with the TIMED-RUN command will not run while WCL is waiting for a program run with this setting enabled to finish.

See also;
EXECWAIT

ALWAYS-EXECWAIT.SMALL

If this setting is turned ON, then the "small" version of WCL will always wait for any EXTERNAL program that is run from it to terminate, before returning to the prompt, or, if running in a WCL batch file, before running the next command in the batch file. This setting does NOT work for spawning OS/2 or DOS sessions when you are running under OS/2.

NOTE: commands scheduled with the TIMED-RUN command will not run while WCL is waiting for a program run with this setting enabled to finish.

See also;
EXECWAIT

BACK-UP-INI-FILES

This setting is effective *only* if WCL is your Windows Shell. If set to 1, then WCL will back up your WIN.INI and SYSTEM.INI files everytime Windows is started. WCL tries to make TWO backups of each file, with extensions of ".WCL". This is useful for restoring your Windows setup in cases when some program has messed up your INI files. If set to 0 (zero) then the backups will not take place.

BATCH-FILE.MAX-LINES

This setting determines the maximum number of lines supported in a WCL batch/script file. This number represents the actual number of lines that WCL will allocate memory for throughout a WCL session. This setting has an internal default value of 180 (i.e., 180 lines in a batch file) and this will take effect if the setting in WCL.INI is empty. The value assigned here can be anything from 1 to a maximum of 600.

Although this setting can be as high as 600, users should be careful not to set the value higher than is necessary. Although it is called a "maximum number", any value put here actually becomes the default size of WCL batch files, and WCL will allocate sufficient memory to hold that number of lines every time a batch file is executed. A high value may therefore result in a performance loss, and if system memory is low, then there may be other problems. The default value of 180 lines is more than enough for most people, especially since empty lines and "REM" lines are not included in the line count (i.e., you can have as many blank and comment lines as you care for, in a WCL batch file).

See also;
Batch Files

BIGWCL-DEFAULT-FONT

This is the setting for the default font used in the BIGWCL window. There are several variations of the fonts, in ITALICS, and BOLD. These are all presented in a list when you run the WCL configuration program. The fonts should be set from the configuration program, by typing "CFG" from the WCL command prompt.

CONFIRM-OVERWRITES

This sets the behaviour of the File Copy routines. When set to 0 (zero; this is the default) existing files will be overwritten by the versions being copied, WITHOUT WARNING (this is like the DOS Copy command).

If set to 1 (one) then you will ALWAYS be prompted for confirmation before an existing file is overwritten.

NOTE: The WCL.INI file is polled for this setting only ONCE (when the program is loaded) so any change you make to this setting will take effect only after you run WCL again (or if you run another copy of it by typing "WCL" at the prompt, and Close the original copy)

DEFAULT-FONT

This is the setting for the default font used in the WCL window. There are several variations of the fonts, in ITALICS, and BOLD. These are all presented in a list when you run the WCL configuration program. The fonts should be set from the configuration program, by typing "CFG" from the WCL command prompt.

DESKTOP-TIMER-ALWAYS-VISIBLE

This setting determines whether the WCL ticking clock (if enabled, and if it is on the Windows desktop) should always be visible or not. If this setting is enabled, then the clock will always be on top of all other windows. This sometimes leaves "droppings" behind.

See also;
DESKTOP-TIMER.LEFT
DESKTOP-TIMER.TOP
TIMER
TIMER-ON-DESKTOP

DESKTOP-TIMER.LEFT

This setting determines horizontal location of the WCL ticking clock (if enabled, and if it is on the Windows desktop) - i.e., the left hand corner of the clock.

See also;
DESKTOP-TIMER-ALWAYS-VISIBLE
DESKTOP-TIMER.TOP
TIMER
TIMER-ON-DESKTOP

DESKTOP-TIMER.TOP

This setting determines vertical location of the WCL ticking clock (if enabled, and if it is on the Windows desktop) - i.e., the top of the clock display.

See also;
DESKTOP-TIMER-ALWAYS-VISIBLE
DESKTOP-TIMER.LEFT
TIMER
TIMER-ON-DESKTOP

DIR-WINDOW.LENGTH

This is useful in the "small" version of WCL only. If "FILE-DESCRIPTIONS" are enabled, this setting determines the width of the popup window that opens when you run the DIR command. The setting should be wide enough for you to be able to see your file descriptions.

See also;
DESCRIBE
FILE-DESCRIPTIONS

DIRCMD

This setting determines the default switch for the DIR command. Any valid switch or combination of switches can be put here (the default = /ON   (i.e., sort by name)).

Note that any entry on this line will ALWAYS be evaluated LAST. Thus it will ALWAYS override any conflicting switch that is passed from the WCL command line. For example, if you put "/Q" on this line (i.e., no pausing after each screen), and then pass "/P" at the command line (i.e., pause for a key press after each screen), the "/P" that was passed at the command line conflicts with the "/Q" in the entry, and will be ignored.

DO=

With regard to the way WCL runs external programs, some users have complained about the fact that WCL sometimes changes to the directory in which the executable for the program was found, before running it. You can circumvent this feature now. This is implemented when the user types "DO" before the name of the program to be executed. This is explained in more detail under the "DO" command.

You can now avoid having to type "DO" all the time by setting the "DO=" line in WCL.INI to "1", or "ON" (i.e., "DO=1"), or by clicking on the Checkbox titled "Don't Change Directory" in the WCL Configuration program WCLCFG.EXE. With this setting, WCL will behave as if you have typed "DO" before the name of *every* program that you want to execute. This setting is disabled by default.

Note that turning on this setting may cause some programs not to work properly by virtue of not being able to locate their own data files. Thus it may be better to leave this setting disabled, and to use the "DO" command on a case-by-case basis.

EMULATE-DOSKEY

This setting turns on support for a LIMITED emulation of the DOSKEY function of scrolling through a list of past commands (by using the arrow keys - LEFT or UP arrow keys, or CTRL-Z for ("up") and DOWN or RIGHT arrow keys, or CTRL-X (for "down")).

This setting also turns on support for assigning commands to FUNCTION KEYS (F1-F9, and F11-F12); and the CONTROL KEY + AN ALPHABET (CTRL-A to CTRL-Y, but MINUS CTRL-G, CTRL-H, CTRL-M, CTRL-T, and CTRL-X - these are reserved). F10 also is reserved for use by Windows. You can put assign any command to any of the free keys by just putting the command after the "=" sign. WCL will execute the command EXACTLY as it appears here.

These key assignments are carried out manually in KEYS.WCL, which is just a plain ASCII file that can be edited with the Notepad or any other text editor.

A setting of "1" or "ON" turns this setting ON. Any other setting turns it OFF You can also turn it on/off in WCL by typing "DOSKEY ON" or "DOSKEY OFF"

One limitation is that unless you have enabled the LINE-EDITING functions (see below) you can only edit the commands by using the BACKSPACE key.

PLEASE NOTE that turning DOSKEY EMULATION off is NOT recommended.

ENABLE-DOS-SERVER

This is used to turn on support for using WCL to run Windows programs from an "MS-DOS prompt". This feature is just a stop over until the release of Windows 95 which has this feature built-in.

See also;
The WCL DOS Server

EXPAND-PATHS

This setting determines whether the WCL RESERVED WORDS are expanded into their full meanings or not, whenever they are encountered at the command line, or in a WCL batch file. They serve as short-cuts to the full significance of the reserved words.

See also;
WCL_Reserved_Words

FILE-DESCRIPTIONS

This setting determines whether support for file descriptions is enabled or not. If enabled, then every time the DIR command is run, each file name is checked to see if it has a description, and if so, the description is added to its name in the DIR display (on screen only). This may result in slower scrolling of the DIR file display, especially if FILES.WCL becomes big.

The descriptions are truncated in memory, so that the whole display can fit into the WCL window. Thus, how much will be visible depends entirely on the width of the WCL window.

See also;
DIR-WINDOW.LENGTH
DESCRIBE
WINDOWLENGTH.BIG

INSERT-MODE

This setting determines the default mode of the doskey emulation. If it is turned on, then the default state of the insert mode in the doskey emulation will be ON (instead of the normal default to over- write mode). This setting has no effect if doskey emulation is not turned on.

LINE-EDITOR

This setting turns the option command-line editing ON or OFF. A setting of "1" or "ON" enables the function, and any other setting disables it. You can also turn command-line editing off and on temporarily by using the new command "LINE-EDITOR ON" (to turn it on) or "LINE-EDITOR OFF" (to turn it off).

If DOSKEY emulation is not enabled, command-line editing will not work. When command-line editing is enabled, you can use the left and right arrow keys, Backspace, Delete, Home, and End, to move about and edit the commands being entered, and also to edit the commands that have been brought up through DOSKEY scrolling. Thanx to Stephen Ryan for the line editing function.

When command-line editing is enabled (the default setting in WCL.INI), the text being typed at the command line will scroll horizontally on the same line, and you can move left or right with the cursor keys. If the prompt is less than 20 characters from the end of the WCL window, the cursor will move to the next line to allow more room for the commands to be typed.

When command-line editing is turned off, then the text being typed will wrap onto the next line. In this case, you cannot go back to the previous line.

PLEASE NOTE that turning the LINE-EDITOR off is NOT recommended.

LOCATION-HORIZONTAL

This is the location of the LEFT HAND side of the WCL window. By default, this is the left edge of the screen. You can increase this if you want the window to be moved to the centre, or the right side of the screen for example.

NOTE: assuming that the screen width is 80 characters, for a Standard VGA screen, multiply each character by 8. So, for the left side of the window to be moved to the CENTRE of the screen for example, you can put LOCATION-HORIZONTAL=320).

The EFFECT of this setting depends entirely on the RESOLUTION of your screen. So for SuperVga modes (e.g.800x600; 1024x768) you will need to increase the multiplication ratio.

The easiest way of setting this is to move the WCL window to the desired location, leave it there, and then type "SAVE" at the WCL command prompt. This will cause the location of the WCL window to be saved in WCL.INI.

Note that when you change this setting, you have to allow for the length of the WCL window as set in WINDOWLENGTH.

LOCATION-HORIZONTAL.BIG

This is the location of the TOP LEFT corner of the BIGWCL window.

LOCATION-VERTICAL

This is the location of the TOP of the WCL window. By default, this is set to the top of the screen. You can increase it if you wish to move the window DOWN, perhaps to the bottom of the screen.

NOTE: assuming that the screen length is 25 lines, for a standard VGA screen, multiply each line by 19. So, to move the window to the bottom of the screen, you can put LOCATION-VERTICAL=475).

This setting determines the TOP of the WCL window. The window itself occupies about 6 lines. So, you effectively have only 19 lines to play with. In the example given above, 475 is the 25th line of the screen. If you use that setting, you WILL NOT SEE any part of the WCL window (it the rest of it will be below the bottom of the screen). The safe setting for the bottom of the screen on Vga mode (640x480) is LOCATION-VERTICAL=361

For SuperVga modes, you again have to increase the multiplication ratio.)

LOCATION-VERTICAL.BIG

This is the location of the TOP of the BIGWCL window.

SAVE-DESKTOP

(this setting *ONLY* has effect when WCL.EXE is your Windows Shell. If set to 0, then the desktop is NOT saved when you exit from WCL. If set to 1 (one) then WCL will save the current state of the Windows Desktop (i.e., all active programs) when you exit. The next time you run Windows, WCL will automatically restore the Windows Desktop to the position it was when you last quit from WCL (i.e., all those active programs will be run automatically). The desktop is saved in a file called "WCL.DSK" in the Windows Directory. This file is in a binary format, so please NEVER try to EDIT it with a text editor. You can of course delete it any time you want.

NOTE: The Desktop will only be saved when you exit from WCL by using one of the WCL exit commands (i.e., "QUIT", "EXIT", "HALT", or pressing the ESCape key). If you exit by pressing Alt F4 or by selecing "Close" from the system menu, then the Desktop will NOT be saved. If you are fond of exiting Windows programs in these ways, then you can save the Desktop manually by typing "SAVE" at the WCL prompt, immediately before quitting.

SAVE-WINDOW-PARMS

This setting decides whether the coordinates of the WCL window are saved each time you EXIT from WCL.

SEARCH-DRIVE

This setting determines whether the "implied CD" feature will search the whole drive for any directory that is supplied at the WCL prompt. This feature is turned ON by default - meaning that anything in that setting other than a "0" or "OFF" will be interpreted as "ON". If the setting is turned OFF, then only the "path" will be searched for a matching directory.

This setting was introduced because searching through the whole drive all the time may be time-consuming on large or networked drives.

SHOW-RESOURCES

This is only valid for the "small" version of WCL. If turned OFF, then the normal display of the system resources and free memory will be turned off.

SHOW-TIMER

This setting dictates whether the on-screen clock is displayed or not.

See also;
DESKTOP-TIMER-ALWAYS-VISIBLE
DESKTOP-TIMER.LEFT
DESKTOP-TIMER.TOP
TIMER
TIMER-ON-DESKTOP

STARTUP1 to STARTUP4

These are for indicating the programs to be loaded by WCL when you start a WCL session. WCL does not load the programs in your Windows start up group file, and thiis the way of compensating.

So if for example, you want CONTROL.EXE to be loaded every time you start WCL, you can put "STARTUP1=CONTROL.EXE". If the programs are not in the DOS path, then you have to type in the FULL PATH of the program (e.g. "STARTUP2=D:\MYDIR\MYPROG\MYPROG.EXE"). NOTE that the program name/path must not exceed 78 characters, or it will be truncated.

Only 4 start up programs are supported here. If you must have more than 4, then put one on the line that reads "RUN=" and one on the line that reads "LOAD=" in your WIN.INI file. Alternatively you can create a WCL batch file (e.g., "STARTUP.CBF") and include its name in one of the startup lines. With such a batch file, you can load as many programs as you wish.

TEXT-BACKGROUND

This setting determines the color of the background of the WCL windows. Although you can theoretically use ANY color here, practically, dark colors are better, and some colors may cause unsightly flashes.

See also;
TEXT-COLOR

TEXT-COLOR

This setting determines the color of the text in the WCL windows. You can use any color here.

See also;
TEXT-BACKGROUND

These settings should contain whole numbers (i.e., without decimals, or commas) that represent the colors that you want for the text and the window background respectively. Since users may be using any number of display cards and any number of screen drivers at any time, the color codes will vary significantly between systems.

If you ever want to return to the default BLACK text on WHITE background, leave this setting empty, and WCL will use the defaults. Also, because of difficulties that users may face with the use of numbers, I have decided to support a number of non-numerical color codes which are constant on systems with 16 or more colors. The colors that you may specify by NAME for the text color and the text background are;

cyan

white

black

red

green

blue

yellow

magenta

gray

lightgray

darkgray

darkyellow

These can be entered in uppercase or lowercase letters - it does not matter.

If you want to use any color other than the above, then you have to use a numeric value that represents its color. Sorry, I can't help you further here. But if it helps, you can use hexadecimal values (ones that begin with $00, and then are followed by SIX values). C or Pascal programmers will be familiar with these. The six values that follow the $00 are RGB values, but used backwards (i.e., the first two for BLUE, the next two for GREEN, and and the last two for RED). "FF" turns the value to full intensity, and "00" turns the color off. Any number between those two will vary the intensity.

e.g.,

$00000000 = black

$00FFFFFF = white

$00FF0000 = blue

$00808080 = gray

$006F9FFF = lightgray

$000000FF = red

$0000FF00 = green

TIMER-ON-DESKTOP

This setting dictates whether the on-screen clock (if enabled) is displayed on the Windows Desktop or not. If this is not enabled, then the clock is displayed inside the WCL window.

See also;
DESKTOP-TIMER-ALWAYS-VISIBLE
DESKTOP-TIMER.LEFT
DESKTOP-TIMER.TOP
TIMER

TOPMOST-WINDOW

This setting determines whether the WCL window ("small" version only) is always on top.
This is turned OFF by default.

US-DATE-FORMAT

This setting determines whether the date and time stamps in the filenames are listed according to the US date format (mm/dd/yy) or the (default) UK format (dd/mm/yy). Set it to 1 or "ON" for US date format.

USER-MENUS.BIG

This determines whether support for user defined menus is enabled in the "big" version of WCL. See the USERMENU command.

See also;
USERMENU

USER-MENUS.SMALL

This determines whether support for user defined menus is enabled in the "small" version of WCL. See the USERMENU command.

See also;
USERMENU

WCL-PROMPT=pg

This is the default mode of the WCL command line prompt. It displays the current Drive and Directory (like DOS). If this line is empty, then this is still the default prompt. If you wish to customise the WCL environment, you can change this setting. Anything after the "=" sign is taken LITERALLY and will appear EXACTLY as written. The only exception is "PG" which simulates the ubiquitous DOS prompt, and the other "$" values (eg $V, $L, $D, $T, $Q).

So you can simulate the famous DBase "dot prompt" by putting on this line, "WCL-PROMPT=."

You can also simulate the UNIX % prompt by "WCL-PROMPT=%" Alternatively, use your own name, "WCL-PROMPT=JOE BLOGGS>" (Note: The longer the prompt, the LESS space you have at the command line for typing commands)

If you want a space to appear after your prompt, add a hash ("#") to the end of the prompt. e.g. "PROMPT FRED>#" -- this will change the prompt to "FRED> ") To return the prompt to one that shows the current directory, type "PROMPT PG". "$T" returns the current time, "$D" returns the current date, and so on. Note that you should NOT use the hash for "PG".

You can change the prompt at the command line at any time by using the "PROMPT" command.

e.g. "PROMPT %"   or "PROMPT .#"

This change will be saved into WCL.INI if you quit WCL through one of its own exit commands (e.g., "EXIT", "HALT", "QUIT")

WINDOW-BORDER

This setting determines whether the WCL window has a border and title bar or not. When the WINDOW-BORDER is set 1 or ON, the WCL window will have a border and a title bar. Otherwise the window will have no title bar, no minimize button, and no system menu. The window will be resizeable, but will not be moveable. This setting is ON by default. Turning it OFF will not work if DOSKEY EMULATION is not enabled.

WINDOWHEIGHT

This setting determines the height of the "small" WCL window (i.e., the default height of the window, when the program is started - it can always be resized afterward). WCL has some internal defaults depending on driver resolution.

If the setting here is left empty (default), or is less than 40, or contains non-numeric values, it is ignored in favour of the internal defaults. The value you should put here depends on your video mode. However, the best way to handle this setting is by resizing the window manually (with the mouse) and then typing "SAVE", or just exiting WCL. Each time you exit WCL, the current window coordinates are saved, and are used when the program is next executed.

WINDOWHEIGHT.BIG

This setting determines the height of the BIGWCL window (i.e., the default height of the window, when the program is started - it can always be resized afterward). BIGWCL has some internal defaults depending on driver resolution, which should be sufficient for most needs.

If the setting here is left empty (default), or is less than 300, or contains non-numeric values, it is ignored in favour of the internal defaults. The value you should put here depends on your video mode. However, the best way to handle this setting is by resizing the window manually (with the mouse) and then typing "SAVE", or just exiting BIGWCL. Each time you exit BIGWCL, the current window coordinates are saved, and are used when the program is next executed. When resizing the window, make sure that it is high enough for the scrolling in the "DIR" command.

WINDOWLENGTH

This is the length of the WCL window. You can reduce of increase the number from 50. Note that to have enough space for typing commands, 42 is the suggested minimum).

WINDOWLENGTH.BIG

This is the setting for the length (or width) of the BIGWCL.EXE main window. In order for the DIR/W command to work properly, this setting should be at least 75.

See also;
FILE-DESCRIPTIONS

YIELD-TIMER-IN-OS2

This is used to control the WCL timer count under OS/2. Turn this setting OFF to stop yielding the timer under OS/2. Unless the setting is turned OFF manually, WCL will always default to yielding under OS/2. Turning it off is necessary for scheduling commands with the TIMED-RUN command, but is detrimental to multi-tasking under OS/2.

WCL Reserved Words

WCL features a number of **RESERVED WORDS**. These are words which have internal significance, and are automatically expanded into their full meanings when they are encountered at the WCL command prompt, or in a WCL batch file. This automatic expansion takes place only if the **EXPAND-PATHS** setting in WCL.INI is enabled.

The reserved words are;

WINDIR

SYSDIR

WCLDIR

THISEXE

There is also a **RESERVED CHARACTER** (**the semi-colon**).

;

WINDIR : expands into the Windows directory.

SYSDIR : expands into the Windows SYSTEM directory.

WCLDIR : expands into the WCL directory

THISEXE : expands into the full path name of the executable file for this running copy of WCL.

; - this **RESERVED CHARACTER** should only appear on the command line or in WCL batch file **when it is intended to serve as a separator for multiple commands**. Anytime the semi-colon is encountered, WCL **will** take it as a separator for multiple commands. If you do not want it to serve this purpose, then please do not use it at all in a WCL command. The only times when the semi-colon can be used as a literal character are with the **NEWCOMMAND** and **USERMENU** commands.

Note that the RESERVED CHARACTER is effective, regardless of the EXPAND-PATHS setting.

EXAMPLES:

```
IF EXIST "sysdir\bwcc.dll $i" REM $ELSE copy bwcc.dll sysdir
CD wcldir
COPY thisexe a:\
```

Example of the use of the semi-colon for command separation

```
CD C:\BAK; COPY *.DOC A:\; CD C:; DIR *.DOC > LOG.TXT; COPY *.TXT A:\; WCLDIR
```

See also;
EXPAND-PATHS
IF CONDITIONS

DOS PROGRAMS

DOS programs can be run from WCL as easily as Windows programs. You can either create a PIF file for the DOS program, with the PIF Editor, or run the program's binaries directly. NOTE that when running a DOS program for which there is no specific PIF file, Windows will use the settings in a file called _DEFAULT.PIF (note the underscore) which exists in the Windows directory. You can either change the settings in this PIF file to achieve a standardised setting for your DOS programs, or you can delete the file. If you choose the latter option, then Windows will use some other defaults for running your DOS programs.

If you ever have any problem running some DOS programs under WCL, check first for the settings in the _DEFAULT.PIF file, because this is usually where the problem lies. See also the SPAWN, DO, RUNDOS, and TIMER commands (below).

NOTE: there are problems if you try to run FULL SCREEN DOS programs when the timer feature is ON - often you just get dumped unceremoniously to the DOS command prompt. Thus, if you are going to run FULL SCREEN DOS programs from WCL, then you should NOT enable the timer feature, or you should type "TIMER OFF" before running the program.

See also;
DO
EXITRUN
RUNDOS
SPAWN
TIMER

Directory Services

WCL supports a number of internal commands for directory services. The supported internal commands operate similarly to the DOS equivalents. Sometimes however, there may be minor variations. The directory commands are outlined below

ALIAS

ALIAS - VIEW the current list of Directory Aliases in WCL.INI (the first 20), or CREATE a new Directory Alias, or CHANGE an existing one. If the command is used without any parameter, a list of current Aliases is presented.

To CREATE a new alias, or CHANGE an existing one, use ALIAS <ALIAS-NAME> <DIRECTORY-PATH>

e.g. ALIAS   BACKUP   C:\DOCUMENTS\SECRET\BACKUP

If you want to create an ALIAS for the directory in which you are, you can use a dot (".") for the directory path, e.g.,

ALIAS THIS-DIR .

AUTOMATIC DIRECTORY CHANGING

WCL features automatic directory changing (also known as "implied CD"). If you type the name of a directory at the WCL command line, and then add a back-slash or a front-slash to it, the program will attempt to change to that directory, without your having to type "CD" first. In this respect, WCL first searches the "path", and then the whole disk for a matching directory.

e.g.   WINDOWS\

This will take you to the "WINDOWS" directory.

SYSTEM\

If you are in the \WINDOWS directory, this will take you into the \WINDOWS\SYSTEM directory. If you are elsewhere, this will take you to the first occurence of SYSTEM that is found (e.g. \CPBACKUP\SYSTEM).

CD

CD - Change to a directory (alternative commands are CHDIR and CWD). Using this command you can also change to an ALIASed directory.

See also;
AUTOMATIC_DIRECTORY_CHANGING

DESCRIBE

WCL features support for attaching long descriptions to filenames (this cannot be used for directory names). The descriptions are kept in FILES.WCL, a text file which can either be edited manually with a text editor, or in which entries can be made with this command. The descriptions are displayed along with the corresponding filenames when the DIR command is executed (if the FILE-DESCRIPTIONS setting is enabled in WCL.INI.

You should enter only the names of the files (and not their full path names). Each desscription can be up to 55 characters in length. The only limit to the NUMBER of file descriptions is that FILES.WCL must not be larger than 16384 bytes (16kb). If it is larger, then only the first 16k will be read. This limitation is for performance purposes - the file FILES.WCL is read in one go, each time the user runs "DIR".

The Syntax is;
DESCRIBE <filename> <description>

EXAMPLES:

```
DESCRIBE   WCL.EXE A Great Command Line Shell

DESCRIBE   MEMO1.DOC Memo to the MD about promotions
```

If you want to delete an entry, supply "NIL" as the description (e.g., DESCRIBE MEMO1.DOC NIL).

See also;
DIR-WINDOW.LENGTH
FILE-DESCRIPTIONS

DIR

List the files in the directory. This can take parameters. e.g "DIR A:\*.EXE", "DIR C:\MSDOS", etc. If no parameter is supplied, then there will be a listing of the CURRENT directory. The list displays about 20 lines and then pauses for a key press - much like "DIR/P" in DOS.) By default, the directory listing is sorted, according to the names of the files, with sub-directories appearing first before files. You can afterward scroll up and down the file list window with the mouse, or by using the PgUp, PgDn, and arrow keys.

The DIR command takes other parameters to change the order of sorting, or the format of the file listing. To see the options available here, type "DIR /?" (note that there must be a space between the "DIR" and the "/?" parameter).

The Directory listing is sorted, with sub-directories appearing first before files. You can afterward scroll up and down the file list window with the mouse.

The default sorting is by NAME, but you can change the sort order by the /O<D,E,S> switch.

/OD = sort by date

/OE = sort by extension

/OS = sort by size

Note that the directories are ALWAYS sorted by name. These switches only apply to normal files.

Other switches are;

/W = use wide list format

/A = align the file names like under DOS (i.e., no dots).

/Q = ("quick" display) - do NOT pause after each screen.

/S = list matching files in all subdirectories

The "/S" switch cannot be used in addition to any other switch (except "/Q"). The files are NOT sorted at all, and the output cannot be redirected to a file.

NOTE: The DIR command shows in its first column the attributes of each file in the directory, enclosed within "<>"

H	stands for Hidden
S	stands for System File
R	stands for Read Only
A	stands for Archive (i.e, normal file)
DIR	stands for Directory (i.e, this is a sub-directory)
N/A	stands for "No Attribute"

If a file has more than one attribute, they are all listed e.g. <HRSA> for the DOS system files.

Like with the DOS equivalent, the output of the 'DIR' command can be redirected to a FILE or to the printer (LPT1) with ">". You can also re-direct the output of the "DIR" command to the clipboard, by supplying "CLIP" or "CLIPBOARD", after the redirection character ">".

e.g. DIR *.EXE > EXEDIR.TXT (output to a FILE)

```
DIR C:\WINDOWS > LPT1     (output to the PRINTER)
DIR C:\WINDOWS > CLIP   (output to the Windows CLIPBOARD)
```

With the "small" version of WCL, the popup window that displays the directory listing can be RESIZED and MOVED, and there is nothing to stop you from having many DIR windows open. It's quite a straight- forward matter to ensure that only one copy of DIR is running, but I am convinced that there are good reasons for allowing multiple copies. You can compare the contents of two directories by having DIR windows of both of them on screen, for example.

See also;
CLIP
PRINTSCR

DU

DU - This command tells you how much disk space is occupied by the files in a directory (and all its subdirectories), without going through a directory listing. The command can take one parameter - the name of the directory whose disk usage you want. If "DU" is typed without any parameter, it will process the current directory, and all its subdirectories.

Note that if you type just "DU" in the root directory of a large hard disk, you may have a long wait while all the directories in the drive are being processed.

e.g.

DU

means - show the disk usage of ALL files in the current directory tree.

DU *.EXE

means - show the disk usage of all .EXE files in the current directory tree.

DU D:\OS2\*.DLL

means - show the disk usage of all .DLL files in the directory tree of D:\OS2.

HOME

HOME - Change to the WCL directory (alternative commands are GOHOME and HOMEDIR).

MD

MD - Create a Directory (alternative command is MKDIR).

e.g.

```
MD C:\UTILS
```

means - create a new subdirectory called "C:\UTILS". If the directory already exists, you will get a beep.

PWD

PWD	- Show the current directory.

RD

RD - Remove a directory. This will only work if the directory is empty of files (alternative command is RMDIR)

e.g.

```
RD C:\UTILS
```

means - remove the subdirectory called "C:\UTILS". If the directory is not empty, you will get a beep.

File Services

WCL supports a number of internal commands for file services. The supported internal commands operate similarly to the DOS equivalents. Sometimes however, there may be minor variations. The file commands are outlined below

ADD

ADD - Add the contents of one file to another. The file that you want to ADD TO is to be specified LAST, and the the file that you want to add to it is to be specified FIRST (alternative command is CONCAT).

The syntax is thus;

"ADD <File to Add>   <File Added To>"

e.g.   "ADD SECOND.TXT   FIRST.TXT"

This will append or add the contents of SECOND.TXT to FIRST.TXT. This means that after the operation, the file FIRST.TXT will now contain both the original contents of FIRST.TXT, with the contents of SECOND.TXT.

If for Example, FIRST.TXT originally contained "ABC" and SECOND.TXT originally contained "DEF", after the ADD operation, FIRST.TXT will now contain "ABCDEF". The contents of SECOND.TXT   remain unchanged.

Always remember that the SECOND file to be specified is the file that will be ADDED to and that what will be added to it are the contents of the FIRST file to be specified.

Note: You CANNOT use wildcards in this command.

ATTRIB

ATTRIB - VIEW and/or CHANGE the attributes of a file or a group of files. To VIEW the attributes of a file, use ATTRIB <FILENAME>.   You cannot use wildcards if the ATTRIB command is used in this way.

To CHANGE the attributes of a file or files,   use ATTRIB <ATTRIBUTES> <FILESPECS> You can use wildcards when the ATTRIB command is used in this way.

The ATTRIBUTES are represented by;

R for READ ONLY;

S for SYSTEM FILE;

H for HIDDEN;

A for ARCHIVE.

You turn them ON or OFF by supplying a plus (+) or minus (-) BEFORE the attributes.

e.g   ATTRIB   +RH   HIDDEN.DOC   (set HIDDEN.DOC to Read Only and Hidden)

e.g. ATTRIB   -RS   +HA   SYSTEM.DOC   (set   SYSTEM.DOC to Hidden and Archive, and remove the Read Only and System settings)

COPY

COPY - Copy a file or a number of files. This again is similar to the DOS Copy command and wildcards are allowed. You can copy to another drive/directory, etc., or to the printer "LPT1" (alternative command is CP).

NOTE: You can create an ASCII file with "COPY CON <FILENAME>", just as under DOS (e.g., "COPY CON LOADWCL.BAT"). This is useful for quick creation of TEXT FILES from within Windows without loading the NOTEPAD, or any other Text Editor. When the "COPY CON command is invoked, an Edit Window is opened for the text to be typed in. Each line is NUMBERED by WCL, so you can know how many lines you have left (a MAXIMUM of 100 lines of text is permitted, and each line cannot be more than 128 characters in length).

NOTE that the editor is a LINE EDITOR, just as in the DOS command line. Each line must be terminated by a Carriage Return and you CANNOT go back to edit previous lines.

When the editing is complete, type a period or full stop (".") on a line by itself, or type "end" on a line by itself to finish. It is at this point that the file is written to Disk.

The Lines Numbers supplied at the edit screen by WCL will NOT be written into the file, neither will the "end" or the period "." which inform WCL that you have finished editing.

NOTE: That you can also copy a file to the printer. "LPT1" and "PRN" are the only printer ports supported.

e.g. "COPY COMMANDS.SUM LPT1"

This will cause the file COMMANDS.SUM to be printed.

The COPY function tries to ensure that there is enough space on the destination drive for the files to be copied, on a file-by-file basis. If there is insufficient space for a file, there will be an error message to that effect, but the function will then proceed to try and copy any other file listed for copying. This is better than DOS in that DOS terminates the COPY function when there is insufficient space for ANY file, even if there are smaller files that will fit into the the target drive. WCL will copy these smaller ones.

The COPY function also tries to verify that the actual number of bytes copied are equal to the size of each Source file. If there is any discrepancy in the sizes of the copied file and it's copy, there is an error message informing you of this, and the copy is deleted.

NOTE: the files to be copied can be delimited by the "/DATE=" switch. If this switch is used, it *MUST* be the LAST parameter that is passed to the COPY command.

The syntax is;   /DATE=[+][-]<date[today]>

The "=" sign is compulsory. However, the "+" or "-" signs are optional, and can be used to specify files AFTER or BEFORE the specified date. If neither is used, then only files created ON the specified date will be copied. For files created on the CURRENT day you can use "TODAY" instead of a numeric date.

NOTE: if you have turned on the US date format setting, then the dates must be expressed in mm-dd-yyyy format, else, it should be in dd-mm-yyyy format.

EXAMPLES;

```
   COPY *.DOC C:\DOCS /DATE=TODAY   (today's files)
```

```
COPY *.DOC C:\DOCS /DATE=+28-04-1992 (files AFTER 28 April '92)
COPY *.DOC C:\DOCS /DATE=-22-10-1994 (files BEFORE 22 Oct. '94)
COPY *.DOC C:\DOCS /DATE=-TODAY (files BEFORE today)
```

See also;
COPYTREE
MOVE

COPYTREE

COPYTREE   - This command attempts to copy a file specification in a given diretory tree. This includes all the files matching the required specifications in that directory, and in its subdirectories.

The command attempts to re-create the directory tree structure of the SOURCE directory on the TARGET drive/directory. If a particular sub-directory in the tree cannot be created for some reason, the files that belong there will be copied into the root target directory.

If there is any problem with copying any file, the process will abort.

This command has certain restrictions. If it is used on the ROOT directory of any drive, it will NOT re-create the directory tree structure on the target drive/directory. It will just copy all the matching files into the target directory - period.

The syntax for the command is

```
COPYTREE <FILESPEC> <TARGET DIRECTORY>
```

If no directory path is supplied for the "filespec" then it will assume that the directory tree to be copied is the current directory.

Examples;

1. COPYTREE   C:\WP\*.DOC   E:\MYDOCS

Means copy all the .DOC files in C:\WP and ALL its sub-directories, to E:\MYDOCS, re-creating the directory structure of C:\WP under the directory E:\MYDOCS.

2. COPYTREE   *.*   G:\BACKUP

Means copy all the files in the current directory and ALL its sub-directories, to G:\BACKUP, re-creating the directory structure of the current directory under the directory G:\BACKUP.

See also;
COPY
DELTREE

DECODE

DECODE - This command is a Windows implementation of the UU-DECODE function. Use it to UUdecode a file. The command takes only one parameter - the name of the file to UUdecode. The name for the decoded file is retrieved from the UUencoded file. If the filename supplied as a parameter is supplied without any extension, the extension .UUE is assumed. Note that only one copy of WCL can run this command at any particular time - because of shared memory in the DLLs.

eg

```
DECODE THISPROG        - looks for THISPROG.UUE to decode
```

See also;

ENCODE
UNIXENCODE

DEL

DEL   - DELete a file or a number of files. This again is similar to the DOS equivalent. You can use wildcards here (e.g. "DEL *.BAK" - to delete all files with the .BAK extension). (alternative commands are   DELETE, RM, ERASE).

e.g.

DEL FRED.TXT

means - delete the file FRED.TXT

DEL C:\UTILS

means - delete all the files in the subdirectory called "C:\UTILS". It basically means "DEL C:\UTILS\*.*"; in this type of case, you will be asked to confirm that you do want to delete all the files in that directory.

You can delete files by reference to their dates, with the "/DATE=" parameter. This parameter must be the LAST one supplied to the DEL command, and is similar to that of the COPY command (see the COPY command for full explanation of the date parameters).

See also;
COPY
DELTREE

DELTREE

DELTREE   - This command attempts to delete ALL the files in a given diretory tree. This includes all the files in that directory, and in its subdirectories. It then attempts to erase all the subdirectories in that directory tree. An alternative command is NUKE.

NOTE:

I have implemented this command against my better judgment, because users demanded it. I do not think that deleting files should be made easy, since recovering them again may be impossible. In my opinion, DELTREE is a command that is best left well alone. If anybody proceeds to use it, I am cannot accept any responsibility for any loss of data that may ensue.

Because of the drastic nature of what DELTREE does, I have imposed some limitations;

1. You will be asked to confirm TWICE that you wish to proceed.

2. You must supply the name of a valid directory to the command. Just typing "DELTREE" will be rejected. You need to type something like "DELTREE D:\JUNKMAIL"

3. The command will reject any attempt to apply it to the ROOT directory of any drive.

e.g., "DELTREE \"

or   "DELTREE C:\"

these will NOT be accepted.

4. If there is any problem at all with deleting any file, then the process will abort.

See also;
COPYTREE

ENCODE

ENCODE - This command is a Windows implementation of the UU-ENCODE function. Use it to UUencode a file. The command can take 2 parameters - the name of the file to UUencode, and the name of the target UUencoded file. If no name is supplied for the target UUencoded file, the target file is the name of the file being encoded, with a .UUE extension. Note that only one copy of WCL can run this command at any particular time - because of shared memory in the DLLs.

By default, the UUencoded file is in the MS-DOS format. If you want to produce a Unix format UUencoded file, use the UNIXENCODE command (see below). It is exactly like this command, except that it produces a Unix format file.

eg

```
ENCODE THISPROG.ZIP        - UUencodes THISPROG.ZIP to THISPROG.UUE.
ENCODE THIS.ZIP THAT.UUE      - UUencodes THIS.ZIP to THAT.UUE.
```

See also;

DECODE
UNIXENCODE

FILE EXTENSIONS

WCL offers support has been introduced for associating file extensions with an application. If you type the name of a file at the WCL prompt, and the file's extension has an application associated with it in the "Extensions" section of WIN.INI, WCL will execute the relevant application, with the file loaded.

The file extensions which you can associate with WCL (for the use of the File Manager) are .ZIP (PKZIPped files) and .CBF (WCL command batch files). These extensions are already associated with WCL internally - so that if you type for example "FREDDY.ZIP" at the WCL prompt, WCL will unzip the file, if it exists, and if it is a ZIP archive.

FILE NAME COMPLETION

WCL offers limited support for filename completion. If the user types part of a file/directory name and then presses TAB, WCL will attempt to complete the file/directory name from the list of files and directories in the CURRENT directory. If you want to abandon this function, press ESC to clear the command line.

If there is no matching file/directory, you will get a beep. If there is only one match, then the name will be expanded. If the match is a directory, '\' will be appended to its name - unless if the user had typed something else before the file completion was activated. If the '\' character is added to a directory name, pressing ENTER will take you into that directory.

If there is more than one possible matching file/directory, you will also get a beep. Then the names of all possible matches (up to 25) will be put in a list which you can scroll through with the UP and DOWN ARROW KEYS. This feature will not be available if DOSKEY EMULATION is not enabled. This feature is still a bit buggy, but I am working on it.

FOREACH

This implements a limited subset of the Unix FOREACH command, (for operations involving multiple files) without resorting to creating batch files, or to transimtting commands one by one. This is especially useful in cases where commands cannot take wildcards. The FOREACH command transmits the filenames which match the supplied file specifications one-by-one.

The syntax is;

FOREACH variable (filespecs)

After this, there is a prompt "?" which allows you to enter each command of what to do with "variable" and "filespecs" on a line by itself, and then type "end" on a line by itself to transmit the commands (or press ENTER on an empty line to abort). On each line with the "?" prompt, precede any occurence of "variable" with the dollar sign "$" - eg if you call your variable "COUNT" then each time you refer to it in response to a "?" prompt, you will type "$COUNT".

NOTE: If you just type "FOREACH" at the BIGWCL prompt without suppyling "variable" and "filespecs" then you MUST transmit them in response to the FIRST "?" prompt.

e.g., - the input to a FOREACH command can be like this;

```
C:\>      FOREACH i   (*.txt)

?         EXECWAIT notepad $i

?         copy $i a:

?         copy $i e:

?         del $i

?         end
```

What this does is to;

[a] load all .TXT files into NOTEPAD, waiting until you close each one before the next one is loaded

[b] make 2 backups of the amended .TXT files,

[c] and then to delete them.

The method described above is the interactive way of using the FOREACH command. You can run it in another way - by typing everything at once, with each statement terminated by a semi-colon (i.e., ";"), and then typing "END" after the last semi-colon, to transmit the command.

e.g.,

FOREACH i (*.txt); EW notepad $i; EW copy $i a:; del $i; end

If you are not familiar with the Unix FOREACH command, you should not use this command, and IF you use this command, please use it with greeat care, since once you transmit the "end" line, it is very difficult to stop the command. Try not to use with destructive commands like "DEL".

GREP

This implements a limited subset of the GREP utility, which is used for searching for strings in text files. This implementation is limited in that only a few of the switches available in other implementations are supported here.

Syntax = GREP [-n -c -v]   <SearchString>   <Filespec(s)>

-n	= Precede each matching line with its line number
-v	= Print all lines EXCEPT those that match
-c	= Print only the number of matching lines

NOTE: Multiple switches must be separated by spaces

Filespecs can include wildcards.

e.g.

```
GREP -n lpstr *.C
```

means - show all the occurences of the word "lpstr" in all the files with a .C extension - and print the line number of each occurence.

See also;
REPLACE

HIDE

HIDE - Hide a file or a number of files by setting their attributes to "hidden" (alternative command is CONCEAL).

e.g.

```
HIDE *.DOC
```

means - hide all the files with the .DOC extension. Note that they will be listed if you use the WCL "DIR" command.

LESS

This command is a more "Windows" version of TYPE or MORE. It displays the contents of an ascii file in a Windows dialog. From there you can scroll up and down, and copy text to the clipboard. Another feature of this command is that if the file specified does not exist in the current directory, then it will be sought for in all the directories in the "PATH".

There is a limitation in that it can only display files of up to 50kb. If you use it on a bigger file, only the first 50kb of it will be displayed.

The Syntax is;
LESS <filename>

EXAMPLE:

```
LESS WIN.INI
```

See also;
TYPE

MOVE

MOVE - MOVE a file or a number of files from one place to another. This involves two operations; first, copying the file(s), and then, if the copy was successful, deleting the original file(s). This command takes exactly the same parameters as the COPY command.

e.g.

```
MOVE *.DOC A:\
```

means - MOVE all the files with the .DOC extension to the root directory of drive A:

See also;
COPY
COPYTREE
DELTREE

PRINT

PRINT - Print a file, i.e., send it to the printer "LPT1" (alternative command are LPR and LPT). You can print both ASCII and BINARY files in this way. This is basically equal to "COPY /B <FILENAME> PRN" command in DOS.

e.g.

```
PRINT ASCII.TXT
```

means - print the file called ASCII.TXT.

REN

REN - This command is used REName a file, or a directory.

The syntax is REN <OldName> <NewName>

It can be used to rename files and directories whose names are regarded as illegal by DOS (eg because they have spaces in their names). In such cases, enclose the old name of the file in quotes - eg REN "FRED DY" FREDDY

It can also take wildcards - but you MUST use the wildcards in both the "OldName" and "NewName". For example;

```
REN    *.DOC    *.TXT
```

```
REN    WCL*.TXT    *.WCL
```

- the "NewName" in such cases MUST be a wildcard file specification that begins with "*".

eg REN WCL*.TXT FRED*.TXT - this is NOT valid.

REPLACE

This command is for searching for a string in an ASCII file, and replacing all occurences of that string with another one. Before processing the specified file, a backup of it is created, with the extension ".BKK". Please do NOT attempt to use this command on a binary file.

Syntax = REPLACE <OldString>   <NewString>   <Filename>

This means replace all occurences of "OldString" with "NewString" in the file(s) "Filename". "Filename" can include wildcards. The search is not case sensitive, and the "NewString" will be entered just as it is specified at the command line (i.e., the case is preserved).

See also;
GREP

REVEAL

REVEAL - Restore a file or a number of files from "hidden" to normal (alternative command is UNHIDE). Actually, all that this command does is to set the file's attribute to ARCHIVE.

e.g.

```
REVEAL *.DOC
```

means - set the file attributes of all .DOC files to ARCHIVE.

SEEK

SEEK   - Try to locate a file or a group of files which match the specified filespec. The whole drive is searched for the files, and any matches found are listed. This command accepts wildcard characters. An alternative command is LOCATE.

e.g. "LOCATE WINWORD.EXE" - will look for all occurences of WINWORD.EXE.

"SEEK WP*.*" - will look for files matching this specification.

This command can now take an extra parameter ("/DELETE") after the file specification. This is useful for getting rid of files of a particular specification. The deletion takes place in the current directory tree only. If you want to use it to delete a file spefication through out the whole drive, you have to run this from the ROOT directory.

Use this parameter with care! You are only given ONE warning.

Example;

SEEK *.BAK /DELETE

This will seek for all files with a *.BAK extension, in the CURRENT directory tree, and delete any matching ones.

If the file specification is "*.*", the /DELETE parameter will NOT be accepted.

TEXT2COM

This command converts a text (ascii) file into a self displaying executable file. You can give the converted file an extension of .EXE or .COM (although it is actually a .COM file). Because it is a .COM file, the ascii file you are trying to convert cannot be larger than 64000 bytes.

When the executable file is run, it will ask whether you want output to the (S)creen or to the (P)rinter. If you output to the screen,, there will be a pause for a key press after each screen is displayed.   Note that you should only use this command to process ascii files. Trying to use it on binary files will lead to unpredictable results.

The Syntax is;

```
TEXT2COM <ascii file> <file[.exe][.com]>
```

EXAMPLES:

```
TEXT2COM FRED.TXT FRED.COM
TEXT2COM HELP.ASC HELP.EXE
```

TYPE

This command displays the contents of an ASCII file, or a group of files, 20 lines at a time. Note that you should not put the "<" character before the file name (unlike DOS). This command can take wildcards.

Examples;

```
TYPE MYFILE.TXT
MORE HELLO.DOC (not "MORE<HELLO.DOC")
TYPE *.TXT
```

See also;
LESS

UNIXENCODE

UNIXENCODE - this command UUencodes a file in the Unix format.

If you want to produce an MS-DOS Unix format UUENCODEd file, use the ENCODE command (see below).

eg

UNIXENCODE THISPROG.ZIP - UUENCODEs THISPROG.ZIP to THISPROG.UUE, in the Unix format.

UNIXENCODE THIS.ZIP THAT.UUE - UUENCODEs THIS.ZIP to THAT.UUE, in the Unix format.

See also;

ENCODE
DECODE

UNZIP

This command is designed to extract files from ZIP archives which have been produced by PKZIP(tm) or other ZIP utilities. The command can handle most forms of compression used by ZIP archivers, and can reproduce any directory structure found in the ZIP file, although this feature is turned OFF by default (you have to use the -D parameter to turn it on).

This command requires only one parameter - the name of the ZIP file to be processed. If no extension is supplied, a .ZIP extension is assumed. You can use wildcard characters here.

You can optionally supply a TARGET directory for the files to be UNZIPped into. If no target directory is supplied, the files are extracter into the CURRENT directory.

The UNZIP command can take further optional parameters, and these parameters can be in ANY order;

-V = VIEW the contents of the ZIP archive.

-D = Restore the DIRECTORY structure found in the ZIP archive

-F=<filespecs> = process only the specified filespecs. Note that there should be NO space between the "=" sign and the file specifications.

NOTE that the -D and -V parameters cannot be used together, and that only one copy of the program can be unzipping files at any particular time - because of shared memory in the DLLs.

EXAMPLES:

UNZIP -V WINCMD73

- view the contents of wincmd73.zip

UNZIP -D WINCMD73

- extract the files in the ZIP file wincmd73.zip into the current directory, and restore any DIRECTORY structure found in that ZIP file.

UNZIP -D -F=*.EXE WINCMD73

- extract all files with the .EXE extension from the ZIP file wincmd73.zip into the current directory, and restore any DIRECTORY structure found in that ZIP file.

UNZIP WINCMD73 C:\WCL

- extract the files in wincmd73.zip to the directory C:\WCL - do not restore any directory structure in the ZIP file.

UNZIP -D WINCMD73.ZIP C:\WCL

- extract the files in wincmd73.zip to the directory C:\WCL - and restore any directory structure in the ZIP file.

Note: you can unzip a ZIP archive from WCL by just typing the name of the ZIP archive from the WCL prompt (in this case, you have to type the ZIP extension as well). Also, you can associate the ZIP extension with WCL.EXE or BIGWCL.EXE in the "extensions" section of WIN.INI (for use by the Windows File Manager and other Windows equivalents).

Miscellaneous Services

WCL offers a number of services which do not readily fall under any of the previous headings. Thus they are all grouped under the miscellaneous services heading. The miscellaneous commands are outlined below

ABOUT
BEEP
DOS
FREE
HELP
PLAY
SAY
SUM
TYPEWRITE

ABOUT

ABOUT   - Show information about WCL (alternative commands are VER and ID).

BEEP

BEEP - Make the annoying beep sound.

DOS

DOS   - Open a DOS Shell. You return to Windows by typing "exit" (alternative command is SHELL).

FREE

FREE - Show the amount of free space on the drive which you specify after this command (e.g. FREE C: or FREE A:).

HELP

HELP- Load this help screen (alternative commands are H and ?).

PLAY

PLAY - Play back any .WAV sound file. For example, to play the sound file CHIMES.WAV in the windows directory, type "PLAY CHIMES.WAV". Note that Windows requires a sound card to be installed before sound files can be played. If no sound card is installed, this command will produce no result. Alternative command is SOUND.

SAY

SAY	- Show a Dialog Box displaying whatever is typed after this command.

SUM

SUM - Load the Windows Notepad program with WCL.SUM. NOTE that WCL.SUM is a plain ASCII text file. (alternative commands are H2, HELP2, and ??).

TYPEWRITE

TYPEWRITE - This takes WCL into "Type Writer Mode". You are presented with an Edit Window wherein you can type text. When you press <ENTER> the LINE of text is sent to the printer ("PRN"). This command thus turns your Windows and Printer into a pretend Electric Typewriter. You can type as many lines of text as you wish, but bear these in mind;

[a] Each line must terminate with a carriage return

[b] Each line must not be more than 78 characters long

[c] You can have empty lines, just by pressing <ENTER>

[d] This command will NOT work properly with Page Printers (i.e., Laser printers). This is because lasers print one page at a time, and not line by line like dot matrix, inkjet, and daisy wheel printers.

To EXIT from typewriter mode, just type "END" on a line by itself, or a full stop "." ("period" in American) on a line by itself.

System Services

WCL offers a number of system services. Some of the commands are named like some internal DOS commands, but there are several which are unique to WCL. The system commands are outlined below.

In addition to the system services listed below, WCL allows you to use an "AUTOEXEC" batch file. This file is read everytime Windows is loaded, but only if WCL is your Windows Shell. This file should be called AUTOEXEC.CBF and should reside in your WCL directory, or in any directory which is in the DOS path. This file is treated as any normal WCL batch (.CBF) file, and should contain only commands that you wish to run EVERYTIME Windows is loaded. Note that batch commands may behave strangely because of the re-entrant nature of Windows and Win-OS2.

Please NEVER use WCL batch files in situations where things depend on commands being executed in a certain order. There is no way of telling the order in which the commands in your WCL batch file will be executed by Windows. Please note this warning.

BACKUPTHEINIS
CFG
CHANGE
CHG2
CHILDWINS
CLIP
CLOSE-ALL-AND-EXITRUN
CLS
COMMAND_ALIASES
DATE
DO
DOSKEY
DRIVEINFO
EXIT
EXITRUN
GETCOLOR
HALT
IF
KEY_ASSIGNMENTS
KILLPROG
LINE-EDITOR
LISTCOMMANDS
LISTWINS
MEM
NEWCOMMAND
NEWLOGO
NETWORKS
NOTOPMOST
OS2-YIELD-TIMER
PASTE
PATH
PROMPT
RENAMEWIN
RESTART
RUNDOS
RUNHIDDEN

BACKUPTHEINIS

BACKUPTHEINIS - Make backup copies of the WIN.INI and SYSTEM.INI files. WIN.INI is backed up as WIN.WCL and WIN2.WCL, and SYSTEM.INI is backed up as SYSTEM.WCL and SYSTEM2.WCL. Alternative command is BACKINI.

CFG

When this command is run, it loads the WCL configuration program, with access to the correct user INI file. If you just run WCLCFG.EXE from Program Manager, it may be reading/writing from/to the wrong INI file (if there are multipile users and multiple INI files on a network). Thus it is better to run WCLCFG.EXE by running the CFG command from the WCL prompt.

Each time you save the configuration from the WCL configuration program (if that program was started with this command), WCLCFG sends an "UPDATE" message to the WCL window. This will cause the new settings to take effect immediately (the WCL window is redrawn, with a flash).

See also;
CHG2
NETWORKS
UPDATE

CHANGE

CHANGE - Load the WIN.INI, SYSTEM.INI, and WCL.INI files into the Windows Notepad program for editing (alternative commands are CHG and CONFIGURE).

CHG2

CHG2 - Load WCL.INI into the Windows Notepad program for editing (alternative command is WCLINI.).

See also;
CFG
NETWORKS

CHILDWINS

This command does a listing of the child windows of any specified window. You can specify a window by its window title (as it appears on the title bar, in quotation marks) or by its numeric window ID (which you can obtain by the **LISTWINS** command). If any child windows are found, their titles, and window handles (numeric IDs) are listed. Windows which have no titles are listed as "zombie (phantom)".

You can then send messages to such windows with the SENDMESSAGE command. Please be careful what messages to send to such windows. Many windows listed as "zombie (phantom)" are hidden windows of various running applications, which perform very critical tasks. Closing them or sending them some inappropriate messages could lead to unpredictable results. You have been warned!

The Syntax is;

```
CHILDWINS <window ID>
```

EXAMPLES:

```
CHILDWINS "Program Manager"

CHILDWINS 5265
```

See also;
LISTWINS
SENDKEYS
SENDMESSAGE

CLIP

CLIP - this command copies the contents of the BIGWCL window buffer into the Windows Clipboard. Note that you cannot mark text for copying to the clipboard - all the contents of the window are copied. In the "small" WCL version, the equivalent is implemented by clicking on the "Print Window" menu option in the popup windows. You then get an option to print the window buffer to a disk file, or to the clipboard.

In cases when the contents of the WCL windows are copied to the clipboard, it may appear that the text in the clipboard is not as neatly formatted as it appears in the WCL window. This is because the text in the WCL window is formatted with a mono-spaced font, and the text in the clipboard is usually displayed in a proportional font.

In both the big and small version of WCL, if the CLIP command is given a filename as an argument, the contents of that file are copied to the clipboard, in text format only. Note that you should only use this with ASCII files.

You can also re-direct the output of the "DIR" command to the clipboard, by supplying "CLIP" or "CLIPBOARD", after the redirection character ">". (e.g., "DIR *.EXE /W > CLIP").

See also;
PASTE
PRINTSCR

CLOSE-ALL-AND-EXITRUN

This command closes all active Windows sessions, **WITHOUT WARNING**, then closes Windows, runs a specified DOS program, and then restarts Windows again. It has the same syntax and limitations as the **EXITRUN** command.

Note: **you should close all DOS sessions manually**, before running this command. The command will actually attempt to close DOS sessions as well as Windows sessions, but trying to close DOS sessions in this way can lead to unpredictable results.

Note also that **this command by-passes the normal exit routines of all the sessions being closed.** This means that whatever activities are normally carried out by the programs when you close them (e.g., prompting to save any unsaved work, writing to configuration files, or saving the session's settings, will **NOT** be carried out. If these things do not matter much to you, then you can freely use this command. It offers a quick-and-dirty way to exit Windows and run a DOS command, and should be used only for that purpose. If you want your active sessions to behave as they normally do when they are closed, then you should use the **EXITRUN** command instead.

See also;
EXITRUN

CLS

CLS - Clear the screen. In WCL, this does nothing. In BIGWCL, it clears the screen, INCLUDING the contents of any scroll-back buffer.

COLOR

COLOR - Change the color settings for text-color and text-background in WCL.INI. The command takes 2 parameters which determine each of the color settings. The settings that you specify are written to WCL.INI and will take effect immediately. The available colors are discussed in relation to the INI file settings. Ideally, the colors should be set from the configuration program, by typing "CFG" at the WCL prompt. Alternative command is SETCOLOR.

e.g.,   COLOR black white

COMMAND ALIASES

COMMAND ALIASES - Users can create an alias for any command, up to a MAXIMUM of 30 aliases. This has required a new "[commands]" section to be added to WCL.INI, just before the "[directories]" section. Unlike the directory aliases which require WCL.INI to be read each time, command aliases are loaded into memory only ONCE - when WCL is loaded. Thereafter, they are processed from memory, until you exit the current WCL session. This makes the feature as fast as internal WCL commands. But it also means that any new command alias that is created in the current WCL session will not be evaluated until the next time you run WCL (unless in cases where the new command alias was created by using the "NEWCOMMAND" command).

Restrictions are that the whole line on which the alias exists cannot be more than 79 characters in length, and that you have to create the command aliases by manually adding each new alias to the "[commands]" section in WCL.INI, or by using the new command "NEWCOMMAND" (see below).

Examples of manual entries in WCL.INI;

```
CWCL=CD C:\WCL
CT=COPYTREE
```

Note that the command aliases are evaluated *before* internal and external commands. Thus, if you create an alias with the same name as a WCL internal command, it is that alias, instead of the WCL internal command that will be executed.

Note also that WCL interprets the aliased commands LITERALLY. Thus, it is the user's responsibility to ensure that the commands for which aliases are being created are correct and non-destructive.

Finally, do NOT create a command alias which involves the execution of a WCL batch file. Please note this point, as I cannot guarantee how the whole thing will be evaluated, if it involves the execution of a .CBF batch file.

DATE

DATE - Display the current time and date.

DO

DO - with regard to the way WCL runs external programs, some users have complained about the fact that WCL sometimes changes to the directory in which the executable for the program was found, before running it. This feature exists for many reasons (partly to do with Windows itself), but it can be circumvented. I have decided to keep that feature as the default, but to permit users to circumvent it. This is implemented when the user types "DO" before the name of the program to be executed.

If this is typed before the name of an external program then WCL will remain in the current directory while running that program, and will not change to the program's directory as it sometimes does. The only use of this command is as a prefix to whatever you would normally have typed. Thus if you would normally type;

```
PKUNZIP WINCMD*.ZIP
```

you would now type;

```
DO PKUNZIP WINCMD*.ZIP
```

Most people will never need to use this "DO" command, but it is there anyway. See also the RUNDOS, and SPAWN commands.

See also;
RUNDOS
SPAWN

DOSKEY

DOSKEY - Use this command to enable or disable DOSKEY emulation during the current WCL session. Nothing is written to WCL.INI. This command takes one parameter (either "ON" or "OFF"). Note that using this command also affects support for assigning commands to the funtion keys. If the command is typed with no parameter, then you are just shown the current status of the DOSKEY emulation.

DRIVEINFO

This command gives some information about your hard disk drives. Some types of drive (e.g. Stacked drives and CD-ROM drives) may be identified wrongly.

EXIT

EXIT - Quit from WCL. If WCL is your Windows Shell, then this will quit from Windows. You will be invited to confirm that you DO want to quit (alternative commands are QUIT or pressing   ESC)

See also;
EXITRUN
HALT
WMCLOSE

EXITRUN

EXITRUN - this command CLOSES DOWN WINDOWS, and then runs and DOS program which you supply as an argument. After the DOS program has terminated, it loads Windows again. Note that the program supplied MUST be a DOS program. If you use this command to run a Windows program, all you will succeed in doing is to close down Windows and restart it again.

The syntax is   EXITRUN <dosprogram> [parameters]

This command is useful for those situations in which a program MUST be run under DOS - it saves you having to close Windows, run the DOS program, and then type "WIN" to start Windows again.

If you are supplying a filename as a parameter to the DOS program, you need to supply the FULL pathname of that file.

See also;
CLOSE-ALL-AND-EXITRUN

GETCOLOR

GETCOLOR - Show the current color settings for TEXT-COLOR, and TEXT-BACKGROUND in WCL.INI. Alternative command is GETCOLORS.

HALT

HALT - Same effect as with "EXIT", except that this command will exit Windows whether or not WCL is your Windows Shell (alternative command is CLOSE).

See also;
EXIT
WMCLOSE

IF CONDITIONS

This command provides support for conditional execution of commands. If the specified condition exists, the specified command will be executed. Otherwise, the command will not be executed. You can specify an alternative course of action with the "$ELSE" (or "#ELSE") command.

The syntax is;

IF <condition> <command> [$else <command>]

The valid IF conditions are;

%x
CONFIRM
EXIST
GETFILE
GETSTRING
ISDIRECTORY
ISWINDOW
NOT_EXIST
ONPATH

These can be used from the command line or from within a .CBF batch file.

See also;
$ELSE
Batch_Files
CHGSTR
GOTO
RETURN

IF %x ==

This allows the user to check for parameters passed to the batch file from the command line, by the use of "%1" to "%7" to signify the first to the seventh parameters. If the command line parameter matches the quoted string, then the condition is true, and the command is executed. If the parameter does not match, then the condition is false, and the command is not executed. If a "$ELSE" (or "#ELSE") command is used, then the alternative command is executed instead. This feature is NOT case sensitive - so any mixture of upper and lower case letters will be okay.

The syntax is;

IF %x   == "[string]" <command>

Where "X" = any number from 1 to 7; "[string]" = any string to be matched (the string must be within quotation marks).

EXAMPLES:

IF %1 == " " SAY No Parameter! $ELSE Say Alright!

IF %1 == "WHY?" SAY Why NOT! #ELSE Say Good Question!

IF %1 ==   "C:" SAY I will not FORMAT drive %1!

IF %2 ==   "WINDIR" SAY I will not NUKE C:\WINDOWS!

IF CONFIRM

This presents the user with a dialog box asking a question. If the user clicks on "YES", the condition is true, and the command is executed. If the user clicks on "NO", then the condition is false, and the command is not executed. If a "$ELSE" (or "#ELSE) command is used, then the alternative command is executed instead.

The syntax is;

```
IF CONFIRM   <"Question"> <command>
```

EXAMPLE:

```
IF CONFIRM "Backup Files Now?" BACKUP.CBF $ELSE RETURN
```

IF EXIST

This looks for the file specified in "filename". If the file exists, its full name is stored in the "$variable", the condition is true, and the command is executed. If the file does not exist, then the condition is false, and the command is not executed. If a "$ELSE" (or "#ELSE) command is used, then the alternative command is executed instead.

The syntax is;

```
IF EXIST "<filename> <$variable>" <command>
```

EXAMPLE:

```
IF EXIST "TIN.DPC $i" COPY $i A:
```

IF GETFILE

This presents the user with a file dialog box, consisting of all the files that match the supplied "filespecs". If a file is selected, its full name is stored in the "$variable", the condition is true, and the command is executed. If the user does not select any file from the list, then the condition is false, and the command is not executed. If a "$ELSE" (or "#ELSE) command is used, then the alternative command is executed instead.

The syntax is;

```
IF GETFILE   "<filespecs> <$variable>" <command>
```

EXAMPLE:

```
IF GETFILE "*.PAS $i" copy $i A:
```

IF GETSTRING

This presents the user with an edit dialog box, with a prompt which you have supplied in "prompt".If the user enters a string, and presses ENTER, the string entered by the user is stored in the "$variable", the condition is true, and the command is executed. If the user does enter any string, or presses ESC, or clicks on CANCEL, then the condition is false, and the command is not executed. If a "$ELSE"   (or "#ELSE) command is used, then the alternative command is executed instead.

The syntax is;

```
IF GETSTRING "<prompt> <$variable>" <command>
```

EXAMPLE:

```
IF GETSTRING "Drive? $i" CD $i $else say No Drive!
```

IF ISDIRECTORY

This looks for the directory specified in "directory". If the directory exists, its full name is stored in the "$variable", the condition is true, and the command is executed. If the directory does not exist, then the condition is false, and the command is not executed. If a "$ELSE"   (or "#ELSE) command is used, then the alternative command is executed instead.

The syntax is;

```
IF ISDIRECTORY "<directory> <$variable>" <command>
```

EXAMPLE:

```
IF ISDIRECTORY "C:\UTILS $i" copy *.* $i
```

IF ISWINDOW

This looks whether there is any program whose main window matches the one specified in "title". If the window exists, its window handle is stored in the "$variable", the condition is true, and the command is executed. If the window does not exist, then the condition is false, and the command is not executed. If a "$ELSE"   (or "#ELSE) command is used, then the alternative command is executed instead.

The syntax is;

```
IF ISWINDOW "<title> <$variable>" <command>
```

EXAMPLE:

```
IF ISWINDOW "File Manager $i" KILLPROG $i
```

IF NOT EXIST

This looks for the file specified in "filename". If the file does NOT exist, the condition is true, and the command is executed. If the DOES exist, then the condition is false, and the command is not executed. If a "$ELSE"   (or "#ELSE) command is used, then the alternative command is executed instead.

The syntax is;

```
IF NOT EXIST <filename> <command>
```

EXAMPLE:

```
IF NOT EXIST TIN.DPC WS2 TIN.DPC
```

IF ONPATH

This looks for the file specified in "filename", and searches in all the directories which are in the DOS path. If the file is found in one of these, its full name is stored in the "$variable", the condition is true, and the command is executed. If the file is not found in the path, then the condition is false, and the command is not executed. If a "$ELSE" (or "#ELSE) command is used, then the alternative command is executed instead.

The syntax is;

```
IF ONPATH "<filename> <$variable>" <command>
```

EXAMPLE:

```
IF ONPATH "TIN.DPC $i" COPY $i A:
```

KEY ASSIGNMENTS

FUNCTION KEYS (F1=F9, and F11-F12)

CONTROL KEYS (CTRL-A to CTRL-Y)

WCL supports the assigning commands to FUNCTION KEYS (F1 to F9, and F11 to F12), and the CONTROL KEY + AN ALPHABET (CTRL-A to CTRL-Y, but MINUS CTRL-C, CTRL-G, CTRL-H, CTRL-M, CTRL-T, and CTRL-X - these are reserved). Note that this feature is disabled if DOSKEY EMULATION is turned off.

The commands have to be assigned manually in KEYS.WCL, which is a plain ASCII file that can be edited with Notepad. Any command can be assigned to any of these keys, and any entry is executed IMMEDIATELY, EXACTLY as it appears in KEYS.WCL.

Examples:

```
F1=HELP
F2=
F3=*
F4=
CTRL-A=WPWIN STATUS.WP
CTRL-B=
CTRL-D=
...... and so on .......
```

This is how you assign commands to the function keys and the control keys. I have pre-set F1 for HELP, and F3 to bring up the last command (the asterisk means "fetch the last command").You can change these settings, and you can assign any command to any of the others (except F10). The commands are executed IMMEDIATELY after the key is pressed (ie you do not need to press ENTER). None of the control keys is preset.

Example: `F2=COPY *.DOC A:\`

With this setting, when you press the F2 key, all the files with a .DOC extension are copied to drive A:

A known problem with this feature is that sometimes you may have to press a function key twice, before the command is executed. I have so far been unable to find a way of remedying this situation. The problem does not exist with the control keys.

KILLPROG

KILLPROG - you can use this command to terminate any currently active process, by its process ID number. Get a list of all the currently active processes, with their process ID numbers by typing **LISTWINS**. You then supply the process ID number of the program you wish to terminate, to KILLPROG. An alternative command is KP. This command has the same restrictions as when you use the **SENDMESSAGE** command to send a **CLOSE** message to a window.

Note that once you supply an ID number to KILLPROG, the program which is represented by that ID number is terminated WITHOUT WARNING. Also, note that LISTWINS does not list some programs because I consider that it will be foolish to terminate them. Thus, if you type KILLPROG and then supply an arbitrary number, it may actually represent the ID of a running process, and that process WILL be terminated.

Therefore, use this command with care, and use it only with process ID numbers supplied by LISTWINS. I cannot accept any responsibility for wrongful use of this command (or, indeed, any other WCL command).

Example: KILLPROG 4309
This will terminate the program which has the ID number 4309, without warning.
See also;
LISTWINS
SENDMESSAGE

LINE-EDITOR

LINE-EDITOR - Use this command to enable or disable the COMMAND-LINE EDITING functions during the current WCL session. Nothing is written to WCL.INI. This command takes one parameter (either "ON" or "OFF"). If the command is typed with no parameter, then you are just shown the current status of the LINE-EDITING function.

You can set the default value of this function in the "LINE-EDITOR=" line in WCL.INI. See more information in the "INI File Settings" section.

NOTE: if DOSKEY emulation is disabled, then the line-editing functions will not work.

LISTCOMMANDS

LISTCOMMANDS - produces a list of the first 20 command aliases.

LISTWINS

This command displays a list of the currently active processes, with their process ID numbers. You need this information in order to use the **RENAMEWIN, KILLPROG, and SENDMESSAGE** commands. An alternative command is **LW**.

See also;
KILLPROG
RENAMEWIN
SENDMESSAGE

MEM

This shows various items of information about your Windows setup, including free memory, resources, etc.

NEWCOMMAND

NEWCOMMAND - This command is for the purpose of creating a new command alias (see above), without having to edit WCL.INI manually. This command inserts the new alias into WCL.INI, and retains it in memory for use during the current session. Note that if this command is used to replace a command alias which already exists in WCL.INI, the new one will not take effect until when you next run WCL. Use it only to create NEW command aliases.

Syntax

```
NEWCOMMAND <Alias Name> <Command>
```

Examples;

```
NEWCOMMAND BACKIT COPY *.DOC A:\
NEWCOMMAND CT COPYTREE
```

To delete a particular entry from WCL.INI, supply "NIL" as the command. Note that when you do this, the command alias WILL be deleted from WCL.INI, but it will still be active in memory, until when next you run WCL. So if you type LISTCOMMANDS to see the currently active command aliases, the one that has just been deleted will still be there. Please note this point.

NEWLOGO

This command is for changing the Windows startup logo. The Windows startup logo is a bitmap which is encoded in "RLE 4" format, and is compiled into the Windows loader "WIN.COM" by the Windows SETUP program. The one that is normally displayed when Windows is started is a copy of the file called VGALOGO.RLE (which resides in the Windows SYSTEM directory). Any RLE 4 bitmap with the right specifications can be used instead.

The syntax is: NEWLOGO <RLE-FILE>

This command compiles the RLE 4 bitmap which is supplied as a parameter into a new Windows loader file called "WCLWIN.COM". You can then type "WCLWIN" from the DOS command prompt, instead of just "WIN". Note that you have to supply the full path name of the RLE file that you want to use.

Any bitmap file can be used for these purposes, subject to the following; [a] the bitmap MUST be converted to RLE 4 format (note: the correct format is "RLE 4", and not "RLE 8"); [b] the bitmap must not have more than 16 colours; [c] the RLE 4 file must not be larger than 47kb in size.

There are many shareware Windows graphics editors which can be used to convert standard Windows .BMP files into RLE 4 format. Examples of such programs are "WINGIF", "Paintshop Pro", and the "Graphics Workshop for Windows".

EXAMPLE:

```
NEWLOGO C:\MYLOGOS\CHIEF.RLE
```

NETWORKS

WCL provides support for users on a network through a number of devices. The main scheme centres around support for separate WCL.INI files, and KEYS.WCL files for each user on the network. This way, the WCL environment can be easily tailored to each individual user's taste.

WCL is fully compatible with the Windows for Workgroups, PC-NFS, and Novell Netware environments. Note however, that Netware users cannot dynamically MAP and unMAP drives from WCL. You will need to open a separate DOS session for this purpose, or mount your drives before loading Windows.

NOTE that using WCL on a network requires a network license. The license fee depends on the proposed number of users.

The first device for implementing separate INI files is the command-line parameter "/WCL$=<FileName>". "FileName" here refers to the FULL PATH NAME of the proposed INI file. This parameter will normally by added to the name of the WCL executable in the "Command Line" edit box in the Program Manager. However, you can also supply it as an argument to a WCL executable from any shell that you are using. This parameter can be supplied to WCL.EXE, BIGWCL.EXE, and the configuration program, WCLCFG.EXE.

e.g.;

U:\WCL\WCL.EXE /WCL$=F:\USR\FRED\FRED.FFF

This stipulates that F:\USR\FRED\FRED.FFF is to be the WCL INI file, instead of the WCL.INI file in the WCL directory. All read and write access to the INI file will be directed to this file.

The second device for implementing separate INI files is to stipulate a "Working Directory" in Program Manager, that is different from the directory where the WCL program files are. In such a case, WCL will first look in that directory for its WCL.INI file (note that the file must be called "WCL.INI" - unless you have used the "/WCL$=" parameter above).

There are a number of other things to note;

[a] Note the command "CFG" or "WCLCFG" - when this command is run, it loads the WCL configuration program, with access to the correct user INI file. If you just run WCLCFG.EXE from Program Manager, it may be reading/writing from/to the wrong INI file (if there are multiple INI files on a network). Thus it is better to use the CFG command.

[b] Note also the command "WCLMAP" - to MAP and UNMAP drives on a Novell Netware network only. If you are not on a Netware network, please do NOT try to use this command, or you WILL get a GPF. Also, this command might still be a bit buggy - so please use it with care!

[c] the SPAWN command creates its temporary batch file in the same directory as that in which the user INI file resides - to prevent one user from overwriting another user's temporary batch file.

[d] WCL will always look first for the WCL.INI file in the Working Directory (specified in Program Manager) before looking anywhere else.

[e] the file specified in the "/WCL$=" parameter will take priority over any other INI file. It is your responsibility to ensure that this parameter points to a valid INI file.

[f] the "/WCL$=" parameter can be used for WCL.EXE, BIGWCL.EXE, and WCLCFG.EXE. If

it is used for any of them, then it MUST be used ALL three of them. Otherwise you may get each of them pointing to a different INI file.

[g] WCL will first look for the KEYS.WCL file in the directory where the INI file resides, before looking elsewhere.

[h] If you wish to see which INI file is being used for the current WCL session, type "SET".

[i] the UNZIP, ENCODE, DECODE, and WCLMAP commands can only be used by one copy of WCL at a time - because of shared memory in the DLLs.

See also;
CFG
CHG2
WCLMAP

NOTOPMOST

If the WCL window is the topmost window, this restores it to the same state as any other window.

See also;
TOPMOST

OS2-YIELD-TIMER

This command is only relevant to those running WCL under OS/2. It causes the WCL timer counter to yield (basically, to stop running) when WCL is in the background, while running under OS/2. This is so as not to hamper OS/2's multi-tasking by hogging the CPU. However, this also means that if the on-screen clock is turned ON, it will only be updated when WCL has the input focus. It also means that programs scheduled with the TIMED-RUN command may not execute, since the timer counter is not being constantly updated.

When this command is run with the "OFF" parameter, then the yielding described above is over-ridden (until turned on again). You should do this if you want to schedule a command with "TIMED-RUN", and you are running WCL under OS/2.

The default behaviour of the timer counter under OS/2 can be specified in the "YIELD-TIMER-IN-OS2=" setting in WCL.INI.

EXAMPLE:

OS2-YIELD-TIMER

Turns ON the timer yielding under OS/2

OS2-YIELD-TIMER OFF

Turns OFF the timer yielding under OS/2

See also;
TIMED-RUN

PASTE

This command pastes the contents of the Windows clipboard into the WCL window. A maximum of 79 characters will be retrieved from the clipboard.   All carriage returns and line feeds in the pasted text are removed and converted to semi-colons (";"). You can then remove these semi-colons manually.

See also;
CLIP

PATH

PATH - Display the current DOS path settings

PROMPT

PROMPT - Show the current WCL prompt, or change it to whatever is typed after this command.

e.g. PROMPT %

This gives you the UNIX percentage prompt

If you want a space to appear after your prompt, add a hash ("#") to the end of the prompt. e.g. "PROMPT FRED>#"   ---- this will change the prompt to "FRED> ")

To return the prompt to one that shows the current directory, type "PROMPT PG". Note that you should NOT use the hash for "PG"

PRINTSCR

It is possible to output the contents of the WCL main windows to other devices. With BIGWCL, you use the command "PRINTSCR". If this command is run without any parameter, the contents of the window buffer are written into an ascii file called BIGWCL.SCR. You can however direct output to the printer by supplying "LPT1" or "PRN" as a parameter, and you can direct output to another ascii file, or to the Windows clipboard, by supplying another file name, or by supplying "CLIP" or "CLIPBOARD" as the parameter.

eg

PRINTSCR	- output to BIGWCL.SCR
PRINTSCR PRN	- ouput to the printer
PRINTSCR SCREEN.TXT	- output to SCREEN.TXT
PRINTSCR CLIPBOARD	- output to the clipboard

This command does not work with the small windowed version (WCL.EXE), because there is nothing to print there. However, when you run commands like "DIR" from the small windowed version, a "Print Window" menu item is inserted into the directory listing window. Clicking on that menu item brings up a dialog that allows you to direct output to the clipboard, or to an ascii file called WCL.SCR.

See also;
CLIP

RENAMEWIN

You can use this command to rename the main window of any active process, by its process ID number. Get a list of all the currently active processes, with their process ID numbers by typing **LISTWINS**. You then supply the process ID number of the program you wish to rename, to RENAMEWIN, and you also supply the new name that you wish to give to that window. The new name is active until you close down the program, or rename it again.

This command does NOT have a permanent effect, and so is quite safe. Alternative commands are **RENW**, and **RENWIN**.

EXAMPLE:

```
RENAMEWIN 4309 My Greatest Windows Hack
```

This will rename the main window of the program which has the ID number 4309, to "My Greatest Windows Hack", until you exit the program, or until you rename it again.

See also;
LISTWINS
SENDMESSAGE

RESTART

RESTART - Shut down Windows and restart Windows again (valid only for Windows 3.1).
Alternative command is **WIN**.

RUNDOS

RUNDOS - Sometimes, users experience some problems with some DOS programs. A typical situation is a case where the user is trying to pass some parameters to a DOS program (e.g., to redirect output from the DOS program to a file). If you have persistent problems with running a particular DOS program through WCL, and nothing else works, try the RUNDOS command.

This is used by typing "RUNDOS" before the name of the program to be executed. What this does is to call your DOS command interpreter (i.e., the "COMSPEC" environment variable) and then pass to it anything that appears after the "RUNDOS" command. This is in effect equal to running the program from the DOS prompt. If you would normally type;

```
PKUNZIP -VN WINCMD73.ZIP > THIS.TXT
```

you would now type;

```
RUNDOS PKUNZIP -VN WINCMD73.ZIP > THIS.TXT
```

NOTE: you should use this command for DOS programs ONLY. Do NOT use it to run Windows programs - it will certainly not work. See also the DO, and SPAWN commands.

See also;
CLOSE-ALL-AND-EXITRUN
DO
EXITRUN
SPAWN
DOS_Programs

RUNHIDDEN

RUNHIDDEN - this command allows you to run a program, but with its main window in a hidden state. In this condition, the relevant program will NOT show up in the task list. However, the program will show up when you use WCL's LISTWINS command. What this means is that you will NOT be able to do anything with that program (e.g., close it) unless you use WCL's SENDMESSAGE command to send a message to its main window.

The syntax is   RUNHIDDEN <progname> [parameters]

The main RESTRICTIONS with this command are, that you cannot use it with a command alias, and if you want to supply a filename as a parameter to the program you are running, you need to supply the FULL pathname of the file.

See also;
LISTWINS
SENDMESSAGE

RUN-MIN

This is for running a program in a minimized state. To use this command, just supply the program to be run as an argument to this command.

EXAMPLE:

```
RUN-MIN   FAXSERV.EXE
```

SAVE

SAVE - Save the current state of the Windows desktop (the desktop is saved into a file called WCL.DSK in the Windows directory. This command also and saves the current WCL window co-ordinates and system settings (e.g., the current WCL prompt, the current Windows Shell, etc.) into the WCL.INI file. IF WCL is your Windows Shell, the saved Windows desktop will be restored when next you run WCL.

SENDKEYS

This is part of the new scripting functionality in WCL. The command is used to send keystrokes and strings to any active application's main window. The window can be identified by its process ID number, or by its title. The process IDs are obtained by the "LISTWINS" command. The title of the main window should be put within quotation marks (" "). The title should be the one which appears on the title bar of the relevant window.

Syntax = SENDKEYS <window> <key(s)>

Normally, any character that is sent to the relevant window is treated as a literal key and sent as such. However, certain keys are RESERVED and have special meanings, and special keys can be sent by putting them within curly braces.

RESERVED KEYS:

"@" = Alt key

"~" = Enter (carriage return)

"+" = Shift Key

"^" = Ctrl Key

"/" = The next key should be treated as a literal key

"&" = Delay for 1 second

Note: Do NOT include the quotation marks!

SENDING SPECIAL KEYS (with curly braces):

Here are some examples of special keys which can be sent by putting their names within curly braces:

1. ENTER = Enter (carriage return)
2. TAB = Tab
3. UP = Up Arrow
4. DOWN = Down Arrow
5. LEFT = Left Arrow
6. RIGHT = Right Arrow
7. BKSP = Back Space
8. F1 = F1 function key

etc., etc.

Examples:

```
SENDKEYS "Program Manager" @FX     (Alt-FX)
SENDKEYS   7483 @FS&@FX               (Alt FS, Delay, Exit)
SENDKEYS   7483 @FS&&&&@FX            (Alt File, Save, 4 second delay, Exit)
SENDKEYS "Program Manager" @FRNotepad~ (File, Run, Notepad, Enter)
```

Type SENDKEYS at the WCL prompt to see other examples.

SENDMESSAGE

This command can be used to send "messages" to any active application through its process ID number. The process IDs are obtained by the **LISTWINS** command. You can also send a message to the application through the title of its main window. In this case, the title of the main window should be put within quotation marks (" "). The title should be the one which appears on the title bar of the relevant window.

The messages that can be sent are;

HIDE (tell the process's main window to hide itself);

MAX (tell the process's main window to maximise itself);

MIN (tell the process's main window to minimise itself);

SHOW or **NORMAL** (tell the process's main window to restore itself to the normal display);

CLOSE (tell the window to shut itself down - this is equal to telling the program that owns the window to terminate; **NOTE: do NOT use this message to close a DOS session** - it will cause Windows to become unstable);

TOPMOST (make the window always on top);

NOTOPMOST (restore a topmost window to normal);

NOTITLE (try to remove the title bar; this may result in a weird looking window; use at your own risk);

RETITLE (try to restore the title bar; this may result in a weird looking window; use at your own risk);

Note that **the CLOSE message by-passes the normal exit routines of the window being closed.** This means that whatever activities are normally carried out by the programs when you close them (e.g., prompting to save any unsaved work, writing to configuration files, or saving the session's settings, will **NOT** be carried out. If these things do not matter much to you, then you can freely use this message. It offers a quick-and-dirty way to close down misbehaving applications which are not responding to other attempts to close them down.

EXAMPLES:

SENDMESSAGE 5464 MAX - the window with the ID of 5464 will maximise itself.

SENDMESSAGE "Borland Pascal" HIDE - the window titled "Borland Pascal" will hide itself.

It is the user's responsibility to make sure that the correct process is being sent the message.

See also;
LISTWINS

SET

SET - Show selected DOS and Windows environment settings. Note that this only SHOWS you the settings. You cannot use the SET command to change any of the environment variables.

SETTIME

SETTIME - Set the system time. Format is hh:mm:ss.

```
    e.g. SETTIME 16:30:45
```

SETDATE

SETDATE - Set the system date. Format is dd:mm:yyyy.

```
    e.g. SETDATE 22:04:1993
```

SETFOCUS

This command is used to set the input focus. If it run without any parameter, it changes the input focus to the WCL window. It can take one parameter - the numeric window ID, or title (within quotes) of the window which should get the focus.

The Syntax is;

```
SETFOCUS <window>
```

EXAMPLES:

```
SETFOCUS "Program Manager"

SETFOCUS 5463
```

SPAWN

With regard to the way WCL runs DOS programs, some users have complained about the fact that WCL sometimes closes the DOS window before they can read the output from the window. This is actually a function of Windows, not WCL. However, I have provided an effective (if inelegant) way to bypass this.

This is implemented by prefixing the program with the "SPAWN" command. When the user types "SPAWN" before the name of a DOS program to be executed, WCL puts everything that appears after "SPAWN" into a batch file, and executes the file. When the file has finished executing, you are invited to press ENTER to return to WCL.

Example: SPAWN PKUNZIP -V *.ZIP | MORE

This will run PKUNZIP with a pipe into the MORE command. When PKUNZIP finishes the DOS window will still be open, and then you will be invited to press ENTER to return to WCL.

NOTE:

With this command, you can also get WCL to execute the COMMAND.COM version of internal DOS commands (eg DIR, PATH, SET, COPY) - instead of the WCL version.

e.g.

"DIR *.DOC" - this will execute WCL's own DIR command

but;

"SPAWN DIR *.DOC" - this will execute the COMMAND.COM version of DIR, in a separate DOS window.

NOTE: do NOT use this command to run Windows programs. It will most certainly NOT work. See also the DO and RUNDOS commands.

See also;
DO
RUNDOS
DOS_Programs

SYSDIR

SYSDIR - Changes to the Windows SYSTEM directory.

The WCL DOS Server

WCL provides (on a temporary basis) the facility to use WCL to run Windows programs from an MS-DOS prompt. This feature will be rendered largely obsolete by Windows 95 (which allows you to run Windows programs from a DOS prompt), but until Win95 is released, this may be of use to some people. This feature also has the added advantage that any command which can be run from the WCL prompt itself can now be run direct from a DOS window.

To activate this feature, you need to put this line very early in your WCL.INI file **ENABLE-DOS-SERVER=1**. When the entry has been made, you then have to close down and restart WCL. From then on, when you are in a DOS window, you can run commands from there by putting "WCL" and a space, before the command.

For example, typing **WCL write** from an MS-DOS prompt will cause WCL to run Windows the WRITE applet. Typing **WCL foreach i(*.txt); notepad $i;end** will cause WCL to run the FOREACH command with the arguments you passed. **WCL backup.cbf** will cause WCL to run the script file BACKUP.CBF.

The potential is thus tremendous. For example, if you have WCL running everytime you start Windows (either through your STARTUP group, or because WCL is your Windows SHELL), and you want WCL to run a command immediately Windows is started, you can type **WCL <command>** in DOS, before running WIN.COM. This way, you do not need to use an AUTOEXEC.CBF file, or to put the commands in your STARTUP lines in WCL.INI. This is useful for adhoc things (you will want to use the AUTOEXEC.CBF file for more long term situations).

Restrictions:

1. When this feature is enabled, it may slow down your system somewhat. On a 486dx/33, I noticed no speed degradation. However, on a 486sx/25, I noticed a slight slowdown.
2. This feature will work with BIGWCL.EXE, but it is **NOT** recommended to use it with BIGWCL (especially not to run WCL's "DIR" command). If you use BIGWCL rather than the "small" version, then it is advised that this feature be disabled.
3. Only one instance of WCL can use it.
4. The TEMP environment variable in your AUTOEXEC.BAT must point to a valid directory
5. The WCL directory MUST be in your PATH statement.
6. The WCL DOS server does **NOT** check whether it is running under Windows or not. This is why running it before loading Windows will still cause WCL to run the command when Windows is started. **Please be very careful about running the WCL DOS server outside of Windows**.
7. Sometimes you may get sharing violation errors.

See also;
ENABLE-DOS-SERVER

SYSINFO

This shows some information about your system. It is equivalent to running MEM and DRIVEINFO in succession.

See also;
DRIVEINFO
MEM

TIME

TIME - Display the current time and date.

TIMED-RUN

This command is used for timed execution (scheduling) of commands. You can schedule a maximum of 20 commands. As each one is executed, it is taken out of the list, and the count reduces. If the maximum is reached, any attempt to schedule another command will replace the last one on the list (you can see the list by just typing "TIMED-RUN" with no parameters). An alternative command is **SCHEDULE.**

When TIMED-RUN is run, WCL automatically activates the on-screen clock until the scheduled program is run. You can use this to check whether the scheduler is active or not.

Also, this feature will not work if WCL is in the background when running under OS/2. The only way to fix this is to stop the yielding of the WCL counter under OS/2 (see the command OS2-YIELD-TIMER).

The syntax is: TIMED-RUN [delete <[all][number]>] [<hh:mm>] [<command>]

You can remove an item from the list or clear the entire list by using the DELETE option. In this case, you should supply as a parameter to DELETE, "ALL" (to clear the entire list) or the number of the item on the list that you wish to delete.

EXAMPLES:

TIMED-RUN 08:30 BACKUP C:\WORK   D:\BACKUP\MORNING
This runs a backup program at 8.30 am. Note that hours less than 10 must be preceded by 0).   Note also that the time must be in 24 hour format. Note also there is no validation of the time that you specify. You should ensure that the time you specify is valid - otherwise, any command that you schedule with an invalid time will simply not run.

TIMED-RUN 23:00 BACKUP C:\WORK   D:\BACKUP\EVENING
This runs a backup program at 11.30 pm.

TIMED-RUN DELETE ALL
This deletes ALL the items on the scheduler's list

TIMED-RUN DELETE 5
This deletes item number 5 on the scheduler's list

See also;
OS2-YIELD-TIMER

TIMER

WCL now has a "TIMER" feature. This is enabled by the new "SHOW-TIMER=" setting in WCL.INI, or by typing "TIMER ON" at the WCL prompt.

What it does is to activate an on-screen clock. There are a number of things to note with respect to this clock;

[a] it is displayed inside the WCL window by default. This can cause a number of problems with untidy screen displays, e.g., when the PROMPT, or the commands being typed over-run the ticking clock,

The only other place I have managed to implement the display of the clock is on the Windows desktop (at the top right corner). You can use this option by setting the "TIMER-ON-DESKTOP=" line in WCL.INI to "ON" or "1", or by typing "TIMER-ON-DESKTOP ON" at the WCL command prompt.

The problems with displaying the clock on the Desktop are, [a] that the clock can be covered by other windows, and, [b] when WCL is closed, you can see still see the last clock display on the Windows desktop (you can get rid of this by maximising a window, and then restoring it to normal again).

[b] There are problems if you try to run FULL SCREEN DOS programs when the timer is ON - often you just get dumped unceremoniously to the DOS command prompt. Thus, if you are going to run FULL SCREEN DOS programs from WCL, then you should NOT enable the timer feature, or you should type "TIMER OFF" before running the program.

See also;
DESKTOP-TIMER-ALWAYS-VISIBLE
DESKTOP-TIMER.LEFT
DESKTOP-TIMER.TOP
TIMER-ON-DESKTOP

TIMER-ON-DESKTOP

This command can be used to state whether the on-screen clock (if enabled) is displayed on the Windows Desktop or not, and any changes made here last only for the current WCL session.   See the "TIMER" command for fuller information.

See also;

See also;
DESKTOP-TIMER-ALWAYS-VISIBLE
DESKTOP-TIMER.LEFT
DESKTOP-TIMER.TOP
TIMER

TOPMOST

This command makes the WCL window the topmost window. This will remain in effect until changed, or until you exit WCL.

See also;
NOTOPMOST

UNLOADLIB

This command unloads "orphan" DLLs from memory (i.e., DLLs which you KNOW shouldn't be in memory because all the programs using them have been closed, but which are still in memory). It should only be used by programmers or others who know what they are doing. **If the DLL you are trying to unload is still being used by another running program, you are CERTAIN to get a GPF. You have been warned!**

The Syntax is;

UNLOADLIB <filename[.dll]>

EXAMPLE:

```
UNLOADLIB CTL3D.DLL
UNLOADLIB VBRUN200.DLL
UNLOADLIB SPREAD.VBX
UNLOADLIB VBRUN300
```

UPDATE

UPDATE - Update the WCL system settings by reading the WCL.INI file again. It is to be used only when you have changed the contents of the WCL.INI file in the current session and you want the changes to take effect immediately without EXITing and restarting WCL. Running the command results in a total redrawing of the WCL window - so there will be some unsightly flashes.

This command is sent to the WCL window when the CFG command is used to run the WCL configuration program (each time you click on the buttons to save the configuration).

See also;
CFG
CHG2

USERMENU

WCL has support for dynamic user defined menus. These are kept in two menu definition files; MENU-S.WCL (for the "small" version of WCL); and MENU-B.WCL (for the "big" version). These files are plain text files that can be edited manually with a text editor, or changed with this command.

A maximum of 20 user menus is permitted in each file. The menu titles (the ones that appear in the WCL menu bars) can be up to 14 characters, and the command attached to each menu item can be up to 79 characters in length.

Single clicking on any menu item results in the command attached to it being sent to the WCL window. The command is immediately executed. The menus can also be executed by an Alt key combination. To define which key should be used for this, put the "&" character before that key.

You might have to resize the WCL window to accommodate the menu items. This is especially so in the "small" version. Note that you should debug your menu items carefully.

The Syntax is;
USERMENU <Menu Title> <command>

EXAMPLES:

```
USERMENU &Quit wmclosewindows

USERMENU E&xit wmclose

USERMENU B&ackup BACKUP.CBF C:\WINWORD\DOCS
```

If you want to delete an entry, supply "NIL" as the command (e.g., USERMENU B&ackup NIL).

WCLFORMAT

This command is for formatting **FLOPPY** diskettes only. In this respect, only drives "A:" and "B:" are supported.

Syntax = WCLFORMAT <drive> [/V[:label]] [/F:size]

The optional parameter /V[:label] specifies the volume label of the disk.

The optional parameter /F:size   specifies the size (in kilobytes) of the floppy disk to format, i.e., 360, 720, 1.2, or 1.44

The progress of the format is shown on the title bar of the WCL window.

Examples:

WCLFORMAT A: (formats disk in drive A: to the drive's capacity)

WCLFORMAT A: /F:720   (formats disk in drive A: to 720k)

WCLFORMAT A: /F:1.44   (formats disk in drive A: to 1.44M)

WCLFORMAT B: /F:1.2   (formats disk in drive B: to 1.2M)

WCLFORMAT A: /V:MYPROG (labels disk as "MYPROG" after format)

WCLFORMAT B: /F:360 /V:MYPROG (360k format, with label)

WCLLOGIN

This command is for logging into Novell Netware servers. Note that you need to attach to the server manually, before running this command.

The syntax is: WCLLOGIN <ServerName> <UserName> <[Password]>

Some types of services do not require passwords. In such cases, you can dispense with the password.

See also;
WCLLOGOUT

WCLLOGOUT

This command is for logging out from Novell Netware servers. This command can be used to logout from a specified server, or from all servers. It can also be used to detach from the servers.

The syntax is: WCLLOGOUT <Server> [/Detach]

Here, "Server" can be either the name of a server, or "ALL". If "ALL" is specified as the server name, the command will logout from all servers, AND also, it will detach from all servers except the current one. If a named server is specified, you can optionally also detach from that server by using the "/Detach" switch.

EXAMPLES:

```
WCLLOGOUT   Fred   /detach
WCLLOGOUT   Emily
WCLLOGOUT   ALL
```

See also;
WCLLOGIN

WCLMAP

WCLMAP - this command is used to MAP and UNMAP drives on a Novell Netware network only. If you are not on a Netware network, please do NOT try to use this command, or you are CERTAIN to get a General Protection Fault in Windows. Also, this command might still be a bit buggy - so please use it with care!

The WCLMAP command is in a WCL extension DLL (WCLMAP.WXX) so it will normally be very slow to load. To speed up loading of commands in .WXX files one should type "EXT" before the command (e.g., "EXT WCLMAP"). For this reason, I have defined a command alias ("WCLMAP=EXT WCLMAP") in WCL.INI. If you are not on a Netware network, you can delete that command alias.

The syntax for the command is;

WCLMAP <newdrive> <path>

e.g., WCLMAP U: SYS:\MAIL\27771 - this maps SYS:\MAIL\27771 (on the current server) to drive U:

To use another server, put the name of that other server WITHIN brackets, followed by a backslash, before supplying the path.

e.g., WCLMAP U: (MY_OTHER_SERVER)\SYS:\MAIL\27771

- this maps SYS:\MAIL\27771 (on the server called "MY_OTHER_SERVER") to drive U:

To UNMAP a drive, the syntax is;

WCLMAP DEL <drive>

e.g., WCLMAP DEL U: - this unmaps drive U:

To LIST all the mapped drives, the syntax is;

WCLMAP LIST

See also;
NETWORKS

WINDIR

WINDIR - Changes to the Windows directory.

WINLOAD

WINLOAD - Change the "LOAD=" setting in WIN.INI. The command can take more than one parameter, and operates exactly like the WINRUN command.

WINRUN

WINRUN -   Change the "RUN=" setting in WIN.INI. The command can take more than one parameter, each of them separated by spaces. The supplied parameters will replace the ones currently on the "RUN=" line. If all you want to do is to ADD extra programs to the "RUN=" line (as opposed to REPLACING the current one) then put a "+" sign BEFORE the first parameter

e.g., "WINRUN   + DRWATSON.EXE   WRITE.EXE"

this will ADD the two named programs to any one that is currently there.

(If no parameter is supplied, the current setting is presented).

To delete all the settings on the line, type   "WINRUN   NIL".

WINSHELL

WINSHELL - Change the "SHELL=" setting in SYSTEM.INI. The command takes one parameter (i.e., the new Windows Shell). If no parameter is supplied, the name of the current Windows Shell is presented.

e.g. WINSHELL WCL.EXE

This changes the Windows Shell to WCL.EXE

WMCLOSE

WMCLOSE - his command shuts WCL down, WITHOUT ANY QUESTION. Some users have complained about being asked whether they really want to quit (when they use other WCL exit commands (such as "HALT" and "EXIT"). This command is for such users. It is also useful for closing down WCL from a non-interactive batch file. When WCL is closed with this command, it still saves the window coordinates before closing.

If you want to replace any of the standard WCL exit commands with WMCLOSE, you may create a command alias for WMCLOSE, using the name of the particular WCL exit command that you wish to replace;

e.g.
```
NEWCOMMAND    ESC    WMCLOSE
```

with this command alias, anytime you press ESC, WCL will close without further reference to you.

See also;
EXIT
HALT

WMCLOSEWINDOWS

This command closes down Windows, WITHOUT WARNING. Please use carefully - and make sure you have saved ALL your work before running this command.

Disclaimer

Registration

WCL is distributed under the Shareware principle. The shareware version can be copied and distributed freely, as long as ALL the supplied files, including documentation (this file) are included, and NO ATTEMPT is made to modify any of the files. The program may not be supplied or bundled with any COMMERCIAL application without prior WRITTEN permission from me.

The Shareware principle means that you get a chance to EVALUATE the program free of charge for a reasonable period of time (in the case of WCL, a maximum of 30 days). It does not mean that you will NOT have to pay for the program. If you find WCL useful and would like to continue using it then you are requested to please REGISTER your copy with the author.

There are 4 LEVELS of registration;

1. STANDARD REGISTRATION

When you register your copy of WCL, you will be sent a SERIAL NUMBER and a code which you will apply to your copy to convert it into a registered version. This will;

[a] remove all registration reminders

[b] register your copy of WCL to you

[c] also enable you to freely register all releases up to the next major upgrade, without reference to me, and without any charge (minor upgrades are numbered in .01 increments, while major upgrades are number in .10 increments, and require a 50% upgrade fee). Basically, you should feel free to obtain and apply the code to all releases, until the code stops working.

[d] when you type "VER", you will see your SERIAL NUMBER, and your NAME (as the registeree) - this is the information that you must use to register subsequent releases.

[e] entitle you to on-line support (via e-mail) for the program

2. OPTIONAL EXTRA: ADD   £3 (Sterling), or $4.50 (US)

If you would prefer to be sent the most current version of WCL by UU-encoded e-mail (instead of just a serial number and code), this can be done - but you will need to pay an extra fee of £3.00 (or $4.50 (U.S.) - meaning a total of   £21 (Sterling), or $30.50 (US) With this optional extra you still get a serial number and a registration code which you can use as described above. NOTE: that the UU-encoded file is VERY large (about 680kb) - so please ensure that your mail system can handle files of that size. Note also that the most current version may also be (and probably is) the same as the version which you already have.

3. CARRIAGE AND HANDLING: ADD   £3 (Sterling), or $4.50 (US)

If you would prefer to be sent the most current version of WCL on a floppy disk, by POST, you will need to pay an extra fee of £3.00 (or $4.50 (U.S.) to cover handling and carriage - meaning a total of   £24 (Sterling), or $35 (US). With the floppy disk option, you still get a serial number and a registration code which you can use as described above. You will also get an unformatted copy of the WCL HELP file as a Windows Write .WRI file.

4. SITE or NETWORK LICENSE

If you are using WCL on a network, a you MUST obtain network license if the program is going to be made available to more than 1 user. Networks/Site licenses are priced for the STANDARD registration fee, multiplied by the number of uses (up to 40), but with generous discounts, which increase according to the number of users.

Purchasers of Network or Site licenses will receive a specially compiled version of WCL, which clearly states the number of people licensed to use the program. If you do not have this special version, you can be SURE that you do NOT have a network or a site license.

REGISTRATION FEE:

1. STANDARD FEE: (serial number and registration code only)

£18.00	**(U.K. STERLING)**
$26.00	**(U.S.)**
$30.00	**(Canadian): British Columbia residents should add 7% Sales Tax**
$40.00	**(Australian)**
Kr170.00	**(Danish)**

2. OPTIONAL EXTRA: (latest version by POST or UU-encoded e-mail)

ADD:

£3.00	**(U.K. STERLING)**
$4.50	**(U.S. and CANADIAN)**
$6.00	**(Australian)**
Kr25.00	**(Danish)**

3. CARRIAGE AND HANDLING (for OPTIONAL EXTRA *and* floppy disk by POST)

ADD:

£3.00	**(U.K. STERLING)**
$4.50	**(U.S. and CANADIAN)**
$6.00	**(Australian)**
Kr25.00	**(Danish)**

(transmission by uu-encoded e-mail is free).

4. SITE or NETWORK LICENSE (specially compiled version of WCL)

[a] 1-9 users: FULL price for each user

[b] 10 users : standard fee, times 10, with a 30% discount

[c] 20 users : standard fee, times 20, with a 50% discount

[d] 40 users : standard fee, times 40, with a 65% discount

[e] 40+ users: standard fee, times 20, with a 15% discount

All prices are subject to change without notice.

If you are outside the U.K. and Ireland, please look in the section titled "REGISTRATION SITES" (below) for the details of registration sites which are close to you, and in which you can register in your own currency, or by credit card. This will save you having to send foreign cheques to me, thereby incurring bank charges.

See also;
Registration_Sites

Users from outside the United Kingdom who wish to send their registration fee to me in the U.K. should please send or an International Money Order drawn out in STERLING only. Otherwise, if sending cheques drawn out in currencies other than Sterling, please add £3 (Three Pounds Sterling) to cover U.K. bank charges. Thus, for example, if you are sending a cheque that is drawn on a bank in the U.S.A, you need add $4.50 to the registration fee to cover U.K. bank charges (meaning $US30.50).

Please note that the extra £3 is COMPULSORY if you are sending me a *cheque* that is not drawn out in Sterling (Euro Cheques are OK, as long as they are drawn out in Sterling).

Registration provides the following benefits:

1. When you register your copy of WCL, you will be sent a SERIAL NUMBER and a registration code which you will apply to your copy to convert it into a registered version. This will;

   [a] remove all registration reminders - no more conscience-pricking nagging messages!

   [b] register your copy of WCL to you - you will become an officially registered user of WCL, and your registration details will be written into the executables.

   [c] enable you to freely register all subsequent releases up to the next major upgrade, without reference to me, and WITHOUT ANY CHARGE (minor upgrades are numbered in .01 increments, while major upgrades are number in .10 increments: MAJOR upgrades require a 50% upgrade fee).

   [d] Basically, you should feel free to obtain and apply the code to all subsequent shareware releases, until the registration code which you will receive stops working. That will be the signal that you now need to pay the upgrade fee.

   [e] when you type "VER", you will see your SERIAL NUMBER, and your NAME (as the registeree) - this is the information that you must use to register subsequent releases.

2. Support for the program

3. A clear conscience

4. The satisfaction of being an honest person

AND,

IF YOU REGISTER WITH THE OPTIONAL EXTRA;

1. If requested, a wordprocessed copy of the on-line help in Windows WRITE format. Note that this file is NOT well formatted. It is just a direct dump of the WCL.HLP help file into a format that can be read by AmiPro or Windows WRITE.

2. Complimentary copies of some of my Windows utilities, namely;

   [a] EXITWIN.EXE (click on the icon of this program, and you quit Windows - no fuss, no warnings, no questions)

   [b] RUNWIN.EXE   (click on the icon of this program, and Windows is shut down and restarted again - useful for those occasions when you need to reboot Windows, e.g. you have just installed a new device driver, or something has corruped Windows, and you want to restart it).

Please note that there is NO printed manual. What I can provide in lieu of this is the on-line help (WCL.HLP) in a wordprocessor file format, for those who take the OPTIONAL EXTRA registration, and request it (see above). Please note also that the most current version may be the same version as the version which you are registering.

If you wish to Register your copy, please send the payment to me at the address below, or to one of my registration sites;

Dr. A. Olowofoyeku,

c/o John Barton

57 Baddeley Green Lane

Baddeley Green

Stoke on Trent

Staffordshire, ST2 7JL

ENGLAND.

Internet:

laa12@keele.ac.uk

chief@mep.com

Compuserve: 100415,3414

Please specify floppy disk size (3.5" or 5.25").

See also;
Registration_Sites
Registration_FORM

Registration Sites

Below are the registration sites for WCL. Please fill the REGISTRATION FORM below.
Registration_FORM

YOU CAN SEND THE REGISTRATION FEE TO ANY OF THE FOLLOWING REGISTRATION SITES;

COMPUSERVE

On-line registration is available under the SWREG scheme. If you **GO SWREG**, the Registration ID for Standard Registration is **3445**; If you are registering with the Optional Extra (diskette by post) the Registration ID is **3429**.

CANADA, and the UNITED STATES

Minds Edge Productions Inc.
P. O. Box 211
3456 Dunbar Street
Vancouver, BC V6S 2C2
Canada

Internet: karsai@mep.com

Fidonet: 1:153/709

BBS: 1-604-737-7026   19.2 ZyXEL   Login: GUEST

WWW: http://www.wimsey.com/~karsai/

Fee: $26.00 (US funds)

or: $30.00 (Canadian funds)

OPTIONAL EXTRA:
add : $4.50
plus: $4.50 (for handling - not applicable to transmission by UUencoded E-mail)

Method of payment: Checks, Money Orders

Make cheques/money orders payable to: "Minds Edge Productions Inc.".

British Columbia residents should add 7% sales Tax.

UNITED STATES

TODD MERRIMAN
Software Toolz, Inc.
8030 Pooles Mill Dr.
Ball Ground,
GA 30107
U.S.A.

Fax: 404-887-5960

Internet: software@toolz.atl.ga.us

Fee: $26.00 (US funds)

OPTIONAL EXTRA:
add : $4.50
plus: $4.50 (for handling - not applicable to transmission by UUencoded E-mail)

Method of payment: Checks, Money Orders, Visa, Mastercard, American Express.

EUROPE

HENRIK MOERK
Survival BBS
P.O.Box 1538
DK-2700 Bronshoj
Denmark

FIDO: 2:231/306

Internet: Lene@vax.psl.ku.dk
Hmk@research.novo.dk

Fee: Kr170.00 (Danish funds)

OPTIONAL EXTRA:
add : Kr25.00
plus: Kr25.00 (for handling - not applicable to transmission by UUencoded E-mail)

Method of payment: Cheques, Eurocheques, Money Orders

GIRO: 1-207-4247

Make cheques/money orders payable to: "HENRIK MOERK".

AUSTRALIA, NEW ZEALAND, ASIA, AND THE FAR EAST

DAVID PERKOVIC
DP Computing
P.O.Box 712
Noarlunga Center
SA 5168
Australia

Internet: perkovic@cleese.apana.org.au
dpc@mep.com

Tel: +61 8 326 4364

Fee: $40.00   (Australian funds)

OPTIONAL EXTRA:
add : $6.00
plus: $6.00   (for handling - not applicable to transmission by UUencoded E-mail)

Method of payment: Cheques, Money Orders

Make cheques/money orders payable to: "DP Computing".

UNITED KINGDOM, IRELAND, EUROPE

John Barton
57 Baddeley Green Lane
Baddeley Green
Stoke on Trent
Staffs, ST2 7JL
ENGLAND.

NOTE: **Please write "WCL REGISTRATION" clearly on the envelope. Failure to do so will result in your order being delayed.**

Internet: laa12@keele.ac.uk
chief@mep.com

Compuserve: 100415,3414

Fee: £18.00 (U.K. funds; or equivalent)

plus: £3.00 (only if sending a foreign cheque)

OPTIONAL EXTRA:
plus: £3.00 (only if sending a foreign cheque)
plus: £3.00 (for handling - not applicable to transmission by UUencoded E-mail)

Method of payment: Cheques, Eurocheques, Money Orders

To register WCL, please PRINT and FILL IN the following Registration FORM.

NOTE: **Please write "WCL REGISTRATION" clearly on the envelope. Failure to do so will result in your order being delayed.**

Please specify your CURRENT version of WCL.

TO:

____

____

____

____

I wish to REGISTER my copy of "WCL". My current version is __________

I am paying the STANDARD REGISTRATION FEE of __________

OR

I am paying for a Site/Network license for [ ] users

____

-- OPTIONAL EXTRA ------------------------------------

NOT APPLICABLE TO SITE OR NETWORK LICENSES

I want the OPTIONAL EXTRA: YES [ ] NO [ ] (please tick)

If YES,

ADD £3 for sending latest version __________

ADD £3 for handling and carriage __________
(NO CHARGE for UUencoded E-mail)

I would like a copy of the latest version sent to me by;

[a] 5.25"[ ] 3.5"[ ] Floppy Disk. (please tick)

OR

[b] by UUencoded e-mail [ ] (internet addresses only)
(note that the UUencoded ZIP file is about 680kb in size)

I use the "big"[ ] "small"[ ] version (please tick)

----------------------------------- OPTIONAL EXTRA ENDS -------------------------------

ADD Tax (if applicable) __________

(See info on the registration sites to see if they collect tax)

Total FEE: ____________

I am paying by Cheque/Money Order/Credit Card (delete as inappropriate)

NAME ____

ADDRESS ____

____

____

POST/ZIP CODE ____

E-MAIL ____

How did you get your copy of WCL?

____

IF PAYING BY CREDIT CARD, PLEASE SEND THE FOLLOWING DETAILS;
(NOTE: Not all sites accept credit cards so please refer to the list of REGISTRATION SITES)

CARD ISSUER	______________________________
CARD NUMBER	______________________________
DATE OF ISSUE	______________________________
EXPIRY DATE	______________________________
SIGNATURE	______________________________
DATE	______________________________

CREDITS

Many thanks to;

1. **Claus Ziegler** of ZieglerSoft, Denmark - the greatest Windows guru that I have ever known - thanks for everything.