



Super Command Shell for Win32 version 6.8
Copyright © 1995,1996 by Lucien Cinc

▼ Expand

- [Information About WinOne](#)
- [Summary of WinOne](#)
- [The WinOne Command Line](#)
- [Additional WinOne Features](#)
- [WinOne System Menu Additions](#)
- [WinOne Command Reference](#)

WinOne[®]

Super Command Shell for Win32 version 6.8
Copyright © 1995,1996 by Lucien Cinc

 Collapse

- [Information About WinOne](#)
- [Introduction](#)
- [Shareware and Registration](#)

- [Windows NT, Windows 95 and Win32s](#)
- [On-line Help](#)
- [New in this Version](#)
- [Glossary](#)

- [Summary of WinOne](#)
- [Main Features](#)

- [The WinOne Command Line](#)
- [Command Execution and Precedence](#)
- [Command Line Macro's](#)
- [Command Line Edit Keys](#)
- [Special Characters](#)
- [Redirecting Command Input and Output](#)
- [Multiple Commands](#)
- [Command Grouping](#)
- [File Names and Wildcards](#)
- [Automatic Directory Changing](#)
- [File Extension Associations](#)
- [ANSI Graphics](#)

- [Batch Programs](#)

- [Additional WinOne Features](#)
- [Virus Protection](#)
- [Startup Batch Program](#)
- [WinOne Parameters](#)
- [Multiple Instances](#)
- [Multiple User](#)
- [Program Manger Replacement](#)
- [File Drag and Drop](#)

- [WinOne System Menu Additions](#)
- [Clipboard Manipulation](#)

- ▣ [Screen Size](#)
- ▣ [Variable Font Sizes](#)
- ▣ [Custom Colours and Colour Schemes](#)
- ▣ [Event Sounds](#)
- ▣ [Function Keys](#)
- ▣ [Program Run Options](#)
- ▣ [Program Manager Programs](#)
- ▣ [User Definable Buttons](#)
- ▣ [Drive Bar](#)
- ▣ [Status Bar and Alarm Clocks](#)
- ▣ [Unix Look and Feel](#)
- ▣ [Mail Notifications](#)
- ▣ [System Configuration Files](#)

- ▣ [WinOne Command Reference](#)
- ▣ [Command Syntax](#)

- ▣ [Commands by Name](#)
- ▣ [Commands by Category](#)

☐
Super Command Shell for Win32 version 6.8
Copyright © 1995,1996 by Lucien Cinc

☐

☐ [Information About WinOne](#)

☐ [Introduction](#)

☐ [Shareware and Registration](#)

☐ [Windows NT, Windows 95 and Win32s](#)

☐ [On-line Help](#)

☐ [New in this Version](#)

☐ [Glossary](#)

☐ [Summary of WinOne](#)

☐ [The WinOne Command Line](#)

☐ [Additional WinOne Features](#)

☐ [WinOne System Menu Additions](#)

☐ [WinOne Command Reference](#)

☐
Super Command Shell for Win32 version 6.8
Copyright © 1995,1996 by Lucien Cinc

☐

☐ [Information About WinOne](#)

☐ [Summary of WinOne](#)

☐ [Main Features](#)

☐ [The WinOne Command Line](#)

☐ [Additional WinOne Features](#)

☐ [WinOne System Menu Additions](#)

☐ [WinOne Command Reference](#)

☐
Super Command Shell for Win32 version 6.8
Copyright © 1995,1996 by Lucien Cinc

☐

☐ Information About WinOne

☐ Summary of WinOne

☐ The WinOne Command Line

☐ Command Execution and Precedence

☐ Command Line Macro's

☐ Command Line Edit Keys

☐ Special Characters

☐ Redirecting Command Input and Output

☐ Multiple Commands

☐ Command Grouping

☐ File Names and Wildcards

☐ Automatic Directory Changing

☐ File Extension Associations

☐ ANSI Graphics

☐ Batch Programs

☐ Additional WinOne Features

☐ WinOne System Menu Additions

☐ WinOne Command Reference

☐
Super Command Shell for Win32 version 6.8
Copyright © 1995,1996 by Lucien Cinc

☐

☐ [Information About WinOne](#)

☐ [Summary of WinOne](#)

☐ [The WinOne Command Line](#)

☐ [Additional WinOne Features](#)

☐ [Virus Protection](#)

☐ [Startup Batch Program](#)

☐ [WinOne Parameters](#)

☐ [Multiple Instances](#)

☐ [Multiple User](#)

☐ [Program Manger Replacement](#)

☐ [File Drag and Drop](#)

☐ [WinOne System Menu Additions](#)

☐ [WinOne Command Reference](#)

☐
Super Command Shell for Win32 version 6.8
Copyright © 1995,1996 by Lucien Cinc

☐

☐ [Information About WinOne](#)

☐ [Summary of WinOne](#)

☐ [The WinOne Command Line](#)

☐ [Additional WinOne Features](#)

☐ [WinOne System Menu Additions](#)

☐ [Clipboard Manipulation](#)

☐ [Screen Size](#)

☐ [Variable Font Sizes](#)

☐ [Custom Colours and Colour Schemes](#)

☐ [Event Sounds](#)

☐ [Function Keys](#)

☐ [Program Run Options](#)

☐ [Program Manager Programs](#)

☐ [User Definable Buttons](#)

☐ [Drive Bar](#)

☐ [Status Bar and Alarm Clocks](#)

☐ [Unix Look and Feel](#)

☐ [Mail Notifications](#)

☐ [System Configuration Files](#)

☐ [WinOne Command Reference](#)

☐
Super Command Shell for Win32 version 6.8
Copyright © 1995,1996 by Lucien Cinc

☐

☐ [Information About WinOne](#)

☐ [Summary of WinOne](#)

☐ [The WinOne Command Line](#)

☐ [Additional WinOne Features](#)

☐ [WinOne System Menu Additions](#)

☐ [WinOne Command Reference](#)

☐ [Command Syntax](#)

☐ [Commands by Name](#)

☐ [Commands by Category](#)

Commands by Name

A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

A

- [ABOUT](#)
- [ACS](#)
- [ASK](#)
- [ATTRIB](#)
- [ARCH](#)

B

- [BEEP](#)
- [BOX](#)

C

- [CALC](#)
- [CALL](#)
- [CAPTURE](#)
- [CDD](#)
- [CHDIR \(CD\)](#)
- [CLIP](#)
- [CLIPDISP](#)
- [CLS](#)
- [COMMA](#)
- [COPY](#)
- [COLOUR](#)

D

- [DATA](#)
- [DATE](#)
- [DATE2SER](#)
- [DDEEXEC](#)
- [DDEPOKE](#)
- [DDEREQ](#)
- [DELBUT](#)
- [DEL \(ERASE\)](#)
- [DESCRIBE](#)
- [DEVICES](#)
- [DIR](#)
- [DIRS](#)
- [DISK](#)
- [DISKFREE](#)
- [DISKUSED](#)
- [DISPBOX](#)
- [DOS](#)
- [DOS2UNIX](#)

- ▣ DRAG
- ▣ DUMP
- ▣ DUPLICAT

E

- ▣ ECHO
- ▣ EDITBOX
- ▣ EJECT
- ▣ END
- ▣ ENDLOCAL
- ▣ EXIT
- ▣ EXTENSION (EXT)

F

- ▣ FILEDATE
- ▣ FILESIZE
- ▣ FILETIME
- ▣ FILETYPE
- ▣ FIND
- ▣ FINDREG
- ▣ FOR
- ▣ FUNC

G

- ▣ GETINI
- ▣ GETKEY
- ▣ GETNUM
- ▣ GETREG
- ▣ GETSTR
- ▣ GO
- ▣ GOTHIC
- ▣ GOTO
- ▣ GOSUB
- ▣ GROUP

H

- ▣ HELP
- ▣ HISTORY (HIS)

I

- ▣ IF

J

No entries

K

No entries

L

- ▣ LABEL
- ▣ LET

- ▢ LOCATE
- ▢ LOGO
- ▢ LOWER

M

- ▢ MACRO
- ▢ MEM
- ▢ MERGE
- ▢ MKDIR (MD)
- ▢ MORE
- ▢ MOVE
- ▢ MSGBOX

N

No entries

O

- ▢ OPENBOX

P

- ▢ PAGE
- ▢ PATH
- ▢ PARSE
- ▢ PAUSE
- ▢ PLAY
- ▢ POPD
- ▢ POSTMSG
- ▢ PRINT
- ▢ PROMPT
- ▢ PUSHD

Q

No entries

R

- ▢ READ
- ▢ READLN
- ▢ REM
- ▢ RENAME (REN)
- ▢ REPLACE
- ▢ RETURN
- ▢ RMDIR (RD)

S

- ▢ SAY
- ▢ SAVEBOX
- ▢ SBANNER
- ▢ SCOPY
- ▢ SENDKEYS
- ▢ SER2DATE

- ▣ SET
- ▣ SETLOCAL
- ▣ SETINI
- ▣ SETREG
- ▣ SHELL
- ▣ SHIFT
- ▣ SLEEP
- ▣ SMOVE
- ▣ SPLIT
- ▣ START
- ▣ STOP
- ▣ STRFIND
- ▣ STRPAD
- ▣ STRREP
- ▣ STRREV
- ▣ STRRFIND
- ▣ STRSIZE
- ▣ STRTRIM
- ▣ SUBSTR

T

- ▣ TASKS
- ▣ TIME
- ▣ TIPS
- ▣ TOUCH
- ▣ TRACE
- ▣ TREE
- ▣ TYPE

U

- ▣ UNCOMP
- ▣ UNIX2DOS
- ▣ UNTAR
- ▣ UNZIP
- ▣ UPPER
- ▣ UUDECODE
- ▣ UUENCODE

V

- ▣ VER
- ▣ VIEWICON
- ▣ VOL
- ▣ VOLUME

W

- ▣ WALLPAPER
- ▣ WHERE
- ▣ WHICH

X

▢ XSET

Y

No entries

Z

No entries

Commands by Category



Commands

[Standard](#)

[Extra](#)

[Windows](#)

[External](#)

Batch Commands

[Standard](#)

[Enhanced](#)

Commands by Category



Commands

- ▣ Standard
- ▣ ATTRIB
- ▣ CHDIR (CD)
- ▣ CLS
- ▣ COPY
- ▣ DATE
- ▣ DEL (ERASE)
- ▣ DIR
- ▣ EXIT
- ▣ HELP
- ▣ LABEL
- ▣ MEM
- ▣ MKDIR (MD)
- ▣ PATH
- ▣ PROMPT
- ▣ SET
- ▣ RENAME (REN)
- ▣ RMDIR (RD)
- ▣ TIME
- ▣ TREE
- ▣ TYPE
- ▣ VER
- ▣ VOL

- ▣ Extra
- ▣ ACS
- ▣ ARCH
- ▣ CDD
- ▣ DELBUT
- ▣ DESCRIBE
- ▣ DISK
- ▣ DOS
- ▣ DUMP
- ▣ FIND
- ▣ FINDREG
- ▣ LET
- ▣ MORE
- ▣ PAGE
- ▣ GO
- ▣ HISTORY (HIS)
- ▣ MACRO

- [-] MOVE
- [-] TIPS
- [-] TRACE
- [-] REPLACE
- [-] SCOPY
- [-] SMOVE
- [-] START
- [-] WHERE
- [-] WHICH
- [-] XSET

- [-] Windows
- [-] ABOUT
- [-] CAPTURE
- [-] CLIP
- [-] CLIPDISP
- [-] DDEEXEC
- [-] DDEPOKE
- [-] DDEREQ
- [-] DEVICES
- [-] DISPBOX
- [-] DRAG
- [-] EDITBOX
- [-] EJECT
- [-] EXTENSION (EXT)
- [-] GROUP
- [-] MSGBOX
- [-] OPENBOX
- [-] POSTMSG
- [-] PRINT
- [-] SAVEBOX
- [-] SENDKEYS
- [-] SHELL
- [-] TASKS
- [-] VIEWICON
- [-] VOLUME
- [-] WALLPAPER

- [-] External
- [-] DOS2UNIX
- [-] DUPLICAT
- [-] FUNC
- [-] GOTHIC
- [-] LOGO
- [-] MERGE
- [-] SBANNER
- [-] SPLIT
- [-] TOUCH

- ▣ UNCOMP
- ▣ UNIX2DOS
- ▣ UNTAR
- ▣ UNZIP
- ▣ UUDECODE
- ▣ UUENCODE

Batch Commands

- ▣ Standard
- ▣ CALL
- ▣ ECHO
- ▣ ENDLOCAL
- ▣ FOR
- ▣ GOTO
- ▣ IF
- ▣ PAUSE
- ▣ REM
- ▣ SETLOCAL
- ▣ SHIFT

- ▣ Enhanced
- ▣ ASK
- ▣ BEEP
- ▣ BOX
- ▣ CALC
- ▣ COLOUR
- ▣ COMMA
- ▣ DATA
- ▣ DATE2SER
- ▣ DIRS
- ▣ DISKFREE
- ▣ DISKUSED
- ▣ END
- ▣ FILEDATE
- ▣ FILESIZE
- ▣ FILETIME
- ▣ FILETYPE
- ▣ GETINI
- ▣ GETKEY
- ▣ GETNUM
- ▣ GETREG
- ▣ GETSTR
- ▣ GOSUB
- ▣ LOCATE
- ▣ LOWER
- ▣ PARSE
- ▣ PLAY

- ▣ POPD
- ▣ PUSHD
- ▣ READ
- ▣ READLN
- ▣ RETURN
- ▣ SAY
- ▣ SER2DATE
- ▣ SETINI
- ▣ SETREG
- ▣ SLEEP
- ▣ STOP
- ▣ STRFIND
- ▣ STRPAD
- ▣ STRREP
- ▣ STRREV
- ▣ STRRFIND
- ▣ STRSIZE
- ▣ STRTRIM
- ▣ SUBSTR
- ▣ UPPER

Commands by Category



Commands

- ▣ Standard
- ▣ ATTRIB
- ▣ CHDIR (CD)
- ▣ CLS
- ▣ COPY
- ▣ DATE
- ▣ DEL (ERASE)
- ▣ DIR
- ▣ EXIT
- ▣ HELP
- ▣ LABEL
- ▣ MEM
- ▣ MKDIR (MD)
- ▣ PATH
- ▣ PROMPT
- ▣ RENAME (REN)
- ▣ RMDIR (RD)
- ▣ SET
- ▣ TIME
- ▣ TREE
- ▣ TYPE
- ▣ VER
- ▣ VOL

▣ Extra

▣ Windows

▣ External

Batch Commands

- ▣ Standard
- ▣ Enhanced

Commands by Category



Commands

- ▣ Standard
- ▣ Extra
- ▣ ACS
- ▣ ARCH
- ▣ CDD
- ▣ DELBUT
- ▣ DESCRIBE
- ▣ DISK
- ▣ DOS
- ▣ DUMP
- ▣ FIND
- ▣ FINDREG
- ▣ LET
- ▣ MORE
- ▣ PAGE
- ▣ GO
- ▣ HISTORY (HIS)
- ▣ MACRO
- ▣ MOVE
- ▣ TIPS
- ▣ TRACE
- ▣ REPLACE
- ▣ SCOPY
- ▣ SMOVE
- ▣ START
- ▣ WHERE
- ▣ WHICH
- ▣ XSET

- ▣ Windows

- ▣ External

Batch Commands

- ▣ Standard

- ▣ Enhanced

Commands by Category



Commands

- ▣ Standard
- ▣ Extra
- ▣ Windows
- ▣ ABOUT
- ▣ CAPTURE
- ▣ CLIP
- ▣ CLIPDISP
- ▣ DDEEXEC
- ▣ DDEPOKE
- ▣ DDEREQ
- ▣ DEVICES
- ▣ DISPBOX
- ▣ DRAG
- ▣ EDITBOX
- ▣ EJECT
- ▣ EXTENSION (EXT)
- ▣ GROUP
- ▣ MSGBOX
- ▣ OPENBOX
- ▣ POSTMSG
- ▣ PRINT
- ▣ SAVEBOX
- ▣ SENDKEYS
- ▣ SHELL
- ▣ TASKS
- ▣ VIEWWICON
- ▣ VOLUME
- ▣ WALLPAPER
- ▣ External

Batch Commands

- ▣ Standard
- ▣ Enhanced

Commands by Category



Commands

- [Standard](#)
- [Extra](#)
- [Windows](#)
- [External](#)
- [DOS2UNIX](#)
- [DUPLICAT](#)
- [FUNC](#)
- [GOTHIC](#)
- [LOGO](#)
- [MERGE](#)
- [SBANNER](#)
- [SPLIT](#)
- [TOUCH](#)
- [UNCOMP](#)
- [UNIX2DOS](#)
- [UNTAR](#)
- [UNZIP](#)
- [UUDECODE](#)
- [UUENCODE](#)

Batch Commands

- [Standard](#)
- [Enhanced](#)

Commands by Category



Commands

Standard

Extra

Windows

External

Batch Commands

Standard

CALL

ECHO

ENDLOCAL

FOR

GOTO

IF

PAUSE

REM

SETLOCAL

SHIFT

Enhanced

Commands by Category



Commands

[Standard](#)

[Extra](#)

[Windows](#)

[External](#)

Batch Commands

[Standard](#)

[Enhanced](#)

[ASK](#)

[BEEP](#)

[BOX](#)

[CALC](#)

[COLOUR](#)

[COMMA](#)

[DATA](#)

[DATE2SER](#)

[DIRS](#)

[DISKFREE](#)

[DISKUSED](#)

[END](#)

[FILEDATE](#)

[FILESIZE](#)

[FILETIME](#)

[FILETYPE](#)

[GETINI](#)

[GETKEY](#)

[GETNUM](#)

[GETREG](#)

[GETSTR](#)

[GOSUB](#)

[LOCATE](#)

[LOWER](#)

[PARSE](#)

[PLAY](#)

[POPD](#)

[PUSHD](#)

[READ](#)

- ▣ READLN
- ▣ RETURN
- ▣ SAY
- ▣ SER2DATE
- ▣ SETINI
- ▣ SETREG
- ▣ SLEEP
- ▣ STOP
- ▣ STRFIND
- ▣ STRPAD
- ▣ STRREP
- ▣ STRREV
- ▣ STRRFIND
- ▣ STRSIZE
- ▣ STRTRIM
- ▣ SUBSTR
- ▣ UPPER

Introduction

Welcome, and thanks for trying WinOne !

WinOne is a Command Language Interpreter, similar in concept to the shell CMD.EXE, except that WinOne has been designed to enable you get the most out of your Windows operating system.

WinOne attempts to make Windows easier to use and to make you more productive when there is a need to work at the command line level. WinOne provides a rich set of commands, including additional convenience functions accessed via the System Menu. Essentially, WinOne provides a very powerful working environment, without sacrificing the flexibility and control you get from working at the command line level.

WinOne is distributed as a Commercial Shareware Program. Please read the [Shareware Information](#) section, which describes the terms and conditions of use for WinOne.

Shareware and Registration

WinOne for Windows is **NOT** free software. WinOne is a Commercial Shareware program, which contains some annoyware. WinOne is protected by the Australian Copyright and International Copyright Laws.

WinOne can be evaluated for a trial period of **20 days**. After that period, if you wish to continue using WinOne, you are required to pay for the program, otherwise, you must discontinue to use WinOne.



To register WinOne, print out the Registration Form, supplied with WinOne in the file **REGISTER.FRM**, fill it out and send it along with the full registration fee to the address below :-

Lucien Cinc
56A Harbord Road
Harbord, NSW 2096
Australia



or for Credit Card orders use the Public Software Library (PsL). You can order with MasterCard, Visa, American express or Discover credit cards from PsL by contacting:

Toll-free: **1-800-2424-PSL**
Phone: **(713)524-6394**
Fax: **(713)524-6398**
CompuServe: 71355,470
Internet: 71355.470@compuserve.com

Mention the product number **14245**, your name, email address and how much you are paying. Note: the people at PsL do NOT know anything about WinOne except for the pricing. All enquires, comments and technical support should be directed to the author of WinOne.



or use the CompuServe Shareware Registration Forum. Just enter GO SWREG and follow the simple instructions. The following is the additional information needed to register WinOne using the Shareware Registration Forum :-

Program Title: WinOne for Windows NT/95
Registration ID: 4302

You will receive a **registration number** that will disable all the annoyware, along with the full **latest version** of WinOne. The full installation of WinOne includes the following additions :-

1. Printable WinOne and WOIO user manuals in Word for Windows format and Postscript format.
2. Full set of external commands. The shareware version only includes ARGS.EXC, UNZIP.EXC and LOGO.EXC.
3. Source code for many example external commands, including the source code for the WOIO library, that is used to write external commands. All source code is written using the C programming language and can be compiled with any C/C++ compiler (eg,

Borland C/C++, Microsoft Visual C++ etc). The supplied WOIO library (ie. WOIO.LIB) is currently compiled using the latest version of Borland C/C++.

A registered user will have a say in the future features that are included in WinOne. Suggestions from non-registered users will not be considered. A registered user is entitled to **product support** via the phone (ie. both FAX and voice), E-mail (ie. both Internet and CompuServe) and will also receive the next upgrade of WinOne free of charge.

Users of WinOne are encouraged to pass along the UNREGISTERED Shareware version of WinOne to other users on a trial, private non-commercial basis. WinOne may not be :-

1. Modified.
2. Distributed in a modified form.
3. Distributed in a registered state.
4. Distributed in connection with any other software, without written permission from the author.

The standard DISCLAIMER follows :-

THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

If you do not agree with the above terms and conditions then you do not have permission to use WinOne and you must stop using it and remove WinOne from your computer.

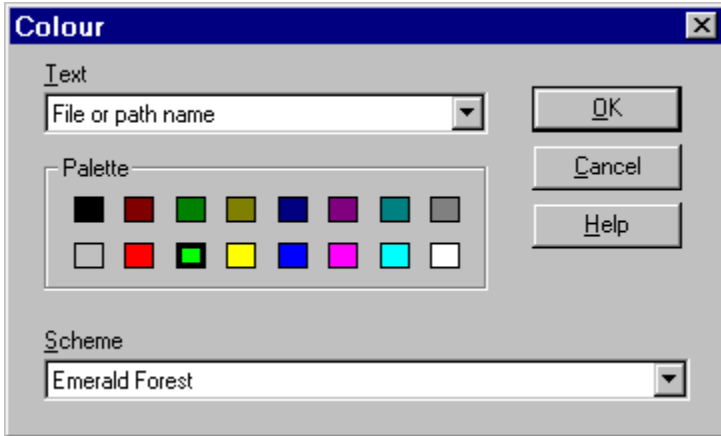
To contact the author, send E-mail to :-

Internet Address: luke@cia.com.au or lcinc@cs.newcastle.edu.au
Home Page: <http://www.cia.com.au/luke> or
http://www.cs.newcastle.edu.au/~lcinc/win_one.html
CompuServe ID: **100353,450**

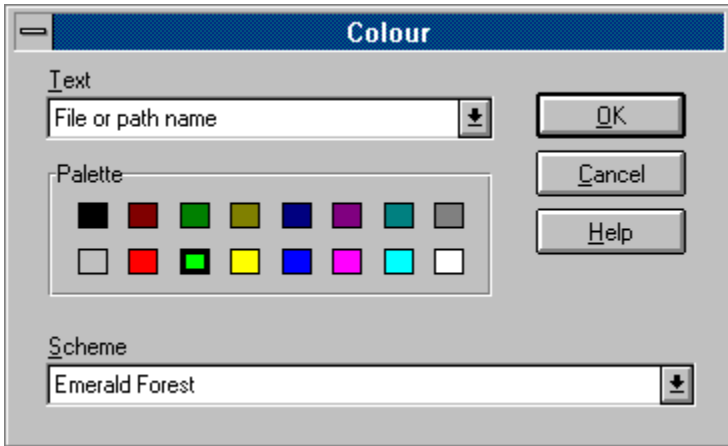
A special **thank you** to all the people that have supported and registered WinOne and if you have not then please consider doing so.

Windows NT, Windows 95 and Win32s

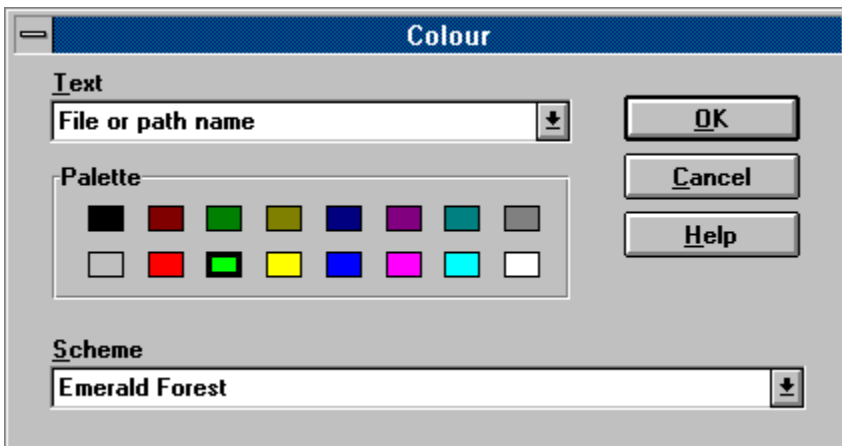
WinOne will run on a host of Windows operating systems. Depending on the operating system that WinOne is running on, the dialog boxes will look slightly different. For example, the colours dialog box will be displayed as follows :-



WinOne running in Windows 95

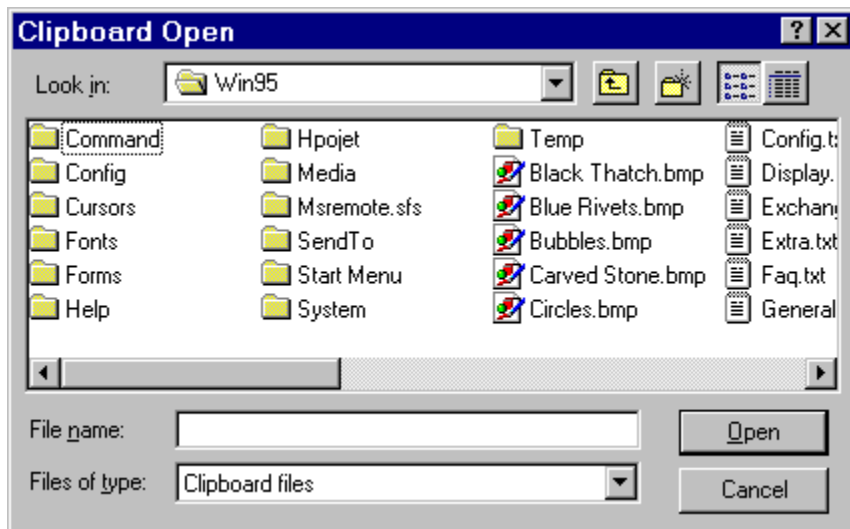


WinOne running in Windows NT



WinOne running in Win32s on Windows 3.1

Similarly, depending on the operating system, WinOne will use the common dialog boxes that are native to the particular operating system. For example the dialog displayed to load a file into the clipboard in Windows 95 will be displayed as follows :-



For simplicity reasons, all pictures included in this documentation will be of WinOne running in Windows NT.

On-line Help

The WinOne help file contains information on using WinOne commands, batch files and other features. There are several ways to locate and display a topic of interest in the WinOne help file, as described below.

The Green Help Button



Press the green help button to display the Contents for the WinOne help file. This button is located on the Button Bar at the top of the WinOne window.

The HELP Command

The HELP command is used to either display the Contents for the WinOne help file or to go directly to a topic contained in the WinOne help file. The WinOne help file contains key words for all features, including commands. The index in the printable documentation contains the complete list of key words that can be used with command HELP.

Context Sensitive Help Using the F1 Key

The F1 key can be pressed to display the help topic for a command that is currently being entered at the WinOne prompt. WinOne will use the first argument (ie. the command name) for the topic to display. When there is no command line, then the Commands by Name topic in the WinOne help file is displayed.

Context sensitive help is also available for the additional menu items contained in the system menu. When an item is highlighted and the F1 key is pressed, then the relevant help topic for the menu item is displayed. When a menu item does not contain a help topic, then the F1 key is simply ignored.

The /? Command Switch

When the question mark switch (ie. '/?') is the only switch specified for a WinOne command, the appropriate help topic is displayed for that command. For example :-

COPY /?

Since macro's can be used to set a commands default switches, then using the question mark switch may not result in the help topic being displayed, since it is a requirement that only the question mark switch be specified for a command. In this case insert a space character in front of the command to over ride the macro.

Start up Tip's

Tip's are a convenient way to draw attention to a particular feature that WinOne includes. The tip's displayed by WinOne include important features and short cuts that will generally simplify a commonly performed task. When a tip includes a reference to a command then to obtain more information about the tip use the HELP command and specify the command as the first parameter (ie. the topic).

Help Buttons on Pop Up Windows

Features that display a pop up window, such as the Colour window, include a help button. Press the help button to display the relevant help topic for the pop up window.

New in this Version

The following contains information about new features that is available in WinOne :-

- ▣ [New in Version 6.8](#)
- ▣ [New in Version 6.7](#)
- ▣ [New in Version 6.6](#) (Not Released)
- ▣ [New in Version 6.5](#) (Not Released)
- ▣ [New in Version 6.4](#)
- ▣ [New in Version 6.3](#) (Not Released)
- ▣ [New in Version 6.2](#)
- ▣ [New in Version 6.1](#)
- ▣ [New in Version 6.0](#) (Not Released)

Versions of WinOne marked as Not Released are distributed to registered users only.

New in Version 6.8

- WinOne, including the WOIO library, has been re-compiled with Borland C++ 4.52.
 - Additional system menu option to use small or large buttons at the top of the WinOne window. See [User Definable Buttons](#) for more information.
 - New [Function key](#) support. Text strings may be set to any of the function keys (excluding the F10 key) and modifiers used with function keys, such as all combinations of the Shift and Control keys.
 - A 256 colour editor which allows the red, green and blue (RGB) values to be set for the default 16 colours used by WinOne. See [Custom Colours and Colour Schemes](#) for more information.
 - New Truetype fonts, including Lucida Console (Windows NT 3.51 and above) and Courier New may be selected and used by WinOne. See [Variable Font Sizes](#) for more information.
- ☐ Comprehensive debugging support for [batch programs](#) using the [TRACE](#) command, including single step, running, stopping and environment variable manipulation while a batch program is being debugged.

☐ New internal windows commands, including :-

General pop up window commands :-

1. [MSGBOX](#) Display a message in a pop up window.
2. [EDITBOX](#) Display an edit field in a pop up window.
3. [DISPBOX](#) Display the contents of a plain text file in a pop up window.

File name pop up window commands :-

4. [OPENBOX](#) Display a file and directory selection pop up window suitable for retrieving an existing file name.
5. [SAVEBOX](#) Display a file and directory selection pop up window suitable for retrieving a file name that does not exist.

Windows message command :-

6. [POSTMSG](#) Post a message to a window.

Clipboard commands :-

7. [CLIP](#) Place a line of text into the clipboard.
8. [CLIPDISP](#) Display any text in the clipboard.

Dynamic data exchange (DDE) commands :-

9. [DDEEXEC](#) Send an execute command to an application.
10. [DDEREQ](#) Send data to an application.
11. [DDEPOKE](#) Request data from an application.

Device control commands :-

12. [DEVICES](#) List all the devices available on a system.
13. [EJECT](#) Eject a CD-ROM.
14. [VOLUME](#) Set or get the volume level for an audio device.

☐ New internal string batch commands, including :-

1. [STRFIND](#) Find the first occurrence of a string within the specified text.

2. STRRFIND Find the last occurrence of a string within the specified text.
3. STRREP Replace all occurrences of a string within the specified text with another string .
4. STRTRIM Remove leading and trailing space characters from a text string.
5. STRREV Reverse all the characters in a text string.

☐ Re-implemented the external command FUNC which was included in the 16 bit version of WinOne. Command FUNC allows Win32 API functions to be called at run time and is only included in the full registered version of WinOne. See the Shareware and Registration section for a list of external commands included in the shareware version of WinOne.

☐ New replaceable parameter %* which returns a batch programs original command line tail.

☐ New multi-dimensional environment variables. See batch programs for more information.

New in Version 6.7

■ Initial port for Windows 95. This version implement fixes and work arounds for problems with Windows 95.

New in Version 6.6

Support for multiple configurations. A user is now able to specify a configuration file as a parameter to WinOne (not to be confused with Multiple User). See WinOne Parameters for more information.

New internal batch commands, including :-

Data manipulation commands :-

1. DATA - Clear or add items to a global list.
2. READ - Read the next item from the global list.

Environment Manipulation commands :-

3. SETLOCAL - Set the localisation of environment variables.
4. ENDLOCAL - End the localisation of environment variables.

System registry commands :-

5. GETREG - Retrieve a value from the System Registry.
6. SETREG - Set or Delete a value from the System Registry.

Miscellaneous command :-

7. GETNUM - Wait for a sequence of characters forming a number.

New internal commands, including :-

1. FINDREG - Search the registry for a string.
2. XSET - Display, set or remove a persistent environment variable.

New dynamic environment variables including :-

1. DATE_FORMAT - International date format.
2. SCREEN_WIDTH - Current screen width.
3. SCREEN_HEIGHT - Current screen height.
4. WHERE_X - Horizontal cursor location.
5. WHERE_Y - Vertical cursor location.
6. DISK_DRIVE - Current disk drive letter.
7. DISK_PATH - Current working directory.

WinOne will now parse %# in a batch program to return the number of command line parameters passed to a batch program. See Batch Programs for more information.

Support for international date formats. All dates are either specified or displayed according to the default system date format. The default system date format is set via the International icon in the Control Panel.

The random desktop wallpaper changer has been transferred to a simple batch program which achieves the same result. See the example batch program for command DATA.

New support for running 16 bit non-text mode programs in a separate VDM. See Program Run Options and command START for more information.

New pop up directory window for the drive bar which enables the current directory to be set using the mouse instead of typing at the keyboard.

New in Version 6.5

▣ New escape character (ie. '^') which is used to over-ride the meaning of any special characters.

▣ The CALC key word is no longer needed for command IF. When using an expr with the IF command the brackets around the expr is sufficient for WinOne to recognise which syntax is being used. For example :-

IF (100 / 5 >= 30) { ECHO true } ELSE { ECHO false }

The above command will display the word "false". WinOne will still parse the IF CALC syntax for backward compatibility.

▣ Command CALC has been extended to include floating point numbers. Floating point numbers allow decimal points and exponents to be used in an expression. Also command CALC includes a number of scientific functions (eg. SIN, COS etc).

▣ Command FOR has been extended to accommodate floating point numbers.

▣ New command LET which evaluates an expression and sets the result to the specified environment variable.

▣ Can load or save the additional following file formats to and from the clipboard :-

1. TXT (plain text)
2. DIB (device independent bitmap)
3. RLE (run length encoded bitmap)

▣ Re-implemented the DRAG commands which was available in the 16 bit version of WinOne. The DRAG command can drag one or more files from WinOne to another window that can accept files.

▣ Improved the look of all the Load and Save windows. Also, all these windows will now remember the last path that a file was either opened from or saved to.

▣ New context sensitive help using the F1 key for the WinOne command line and for the added system menu additions.

▣ New sound effects for particular events associated with WinOne. Event sounds are manipulated via the system menu.

▣ Additional fonts for WinOne may be added via the Font Size window, accessed from the system menu.

▣ The screen size and/or the scroll back buffer may now be set via the system menu.

▣ Added two alarm clocks to the status bar. The alarm clocks are accessed via the system menu.

New in Version 6.4

☐ New dynamic environment variables that are instantiated at run time. See [Batch Programs](#) for more information.

☐ New internal batch commands, including :-

Text string commands :-

1. [COMMA](#) - insert comma's into a number.
2. [STRPAD](#) - pad a string with spaces.

File manipulation commands :-

1. [READLN](#) - get a line of text from a file.
2. [FILESIZE](#) - get the size of a file.
3. [FILEDATE](#) - get the date of a file.
4. [FILETIME](#) - get the time of a file.
5. [FILETYPE](#) - get the type of a file (ie. TEXT or BINARY).

Initialisation file commands :-

1. [SETINI](#) - set or delete a key value from an initialisation file.
2. [GETINI](#) - get a key value from an initialisation file.

Date commands :-

1. [DATE2SER](#) - convert a date to a serial date value.
2. [SER2DATE](#) - convert a serial date value to a date.

☐ Additional [FOR](#) syntax to loop through a range of numbers :-

FOR %variable IS num1 TO num2 [STEP num3] DO command

☐ New /N switch for command [COPY](#), [MOVE](#), [SCOPY](#) and [SMOVE](#) which will convert files with long file names to their equivalent short file names.

☐ Many new switches for command [DIR](#), including G, N, E, D, T, C, W, B, J, F, V.

☐ New [mail](#) support which displays "You have new mail" when enabled and there is new unread mail waiting in your incoming mail box.

☐ New [pop up history list](#) window which contains your complete command history. The history window is displayed using the [HISTORY](#) command.

☐ New /C switch for working with Common Program Manager groups. The default behaviour of command [GROUP](#) is to work with Private Program Manager groups.

☐ WinOne will automatically look for a [STARTUP.BAT](#) batch program in the same directory in which WinOne is installed and if found WinOne will execute it on start up.

☐ WinOne can now be set as the default startup shell replacing the Program Manager. The default startup shell can be set with the internal command [SHELL](#).

☐ WinOne can now be set to capture the input and output of Win32 console programs that use streamable input/output. See the [CAPTURE](#) command for more information.

☐ The Attributes option in the system menu contains a new Run option which can set or

reset the default behaviour of programs that are run from the WinOne prompt. See [Program Run Options](#) for more information

▣ Improved the help file which now includes information that was considered "assumed knowledge". These additions document features that have always been included in WinOne and they do not necessarily represent new features, unless stated above.

New in Version 6.3

☐ New insert or overwrite mode for the WinOne command line. Press the INSERT key to toggle insert or overwrite mode. When insert mode is on a vertical bar is used for the cursor, otherwise, a block is used for the cursor.

☐ Added a drive bar, running along the bottom of the WinOne window. The drive bar displays all your disk drives connected to your computer. This includes network drives. Click on the desired drive icon using the mouse pointer to change to that disk drive.

Also, pressing the left mouse button on the drive bar background is equivalent to pressing the enter key. Similarly, pressing the right or middle buttons on the drive bar background is equivalent to typing DIR at the WinOne prompt.

☐ The drive bar or the button bar can now be removed from the WinOne window.

☐ WinOne is now compiled so that it may run under Win32s installed on Windows 3.1. Win32s is a package from MicroSoft which allows some 32 bit programs to run under Windows 3.1. At the time of this release the latest version of Win32s is version 1.25.

☐ Improved the history list up and down arrows keys which can scroll though your history list for all command lines that start with the same sequence of characters entered at the WinOne prompt. When there is no characters entered at the WinOne prompt then using the up and down arrow keys will simply scroll though you command history list starting with the most recent command line to the oldest command line.

☐ New /B switch for the HISTORY command that can be used to clear the history list.

☐ New functions have been added to the WOIO Library, including :-

scroll()	- scroll a specified number of lines on the screen.
scrflush()	- update the screen.
insline()	- insert a line at the specified position.
delline()	- delete a line at the specified position.

☐ When running any non-text mode type programs WinOne will only launch the program and return the WinOne prompt immediately by default. To change this behaviour see Program Run Options for more information.

☐ Many internal optimisations and improvements.

New in Version 6.2

- WinOne is now compiled using Borland C++ for Windows version 4.5. Borland 4.5 consistently produces smaller and faster 32 bit executable's.
- New internal command START. Users are encouraged to use the ^Z feature included with WinOne instead of using the START command. When WinOne launches any program that creates its own window then switching back to WinOne and pressing ^Z (ie. CTL + Z keys), will return the WinOne prompt immediately.
- New internal command MORE. There is no need to pipe the output of any WinOne internal or external commands to MORE since WinOne automatically pages the output. However, this is not the case for text mode or console type programs where using the MORE command will pipe the output of such programs back to the main WinOne window.
- The internal DOS command can be used to set the behaviour of console windows. Either leaving a console window open and inactive or just closing the window after it has completed it's task. The default behaviour is to leave the window open and inactive.
- Re-implemented the internal command SENDKEYS, which can be used to send a sequence of key strokes to any window.
- The default multiple command separator is now the ampersand character (ie. '&'). See Multiple Commands for more information .
- Support for command grouping using brackets and braces. Using braces instead of brackets is highly recommended since braces effectively localise the instantiation of environment variables. See Command Grouping for more information.
- Improved the command CALC which now recognises many additional operators (eg. %, ~, ^, <, <=, <<, >, >=, >>, ==, !=, !, &, &&, |, ||) and types (eg. numbers and strings). The normal operator precedence is assumed unless the left and right brackets are used to over-ride the default precedence.
- Improved the IF command which now supports the same types of expressions as used with command CALC. Also, a new IF ELSE syntax has been introduced :-

IF [NOT] condition (command(s)) else command

and

IF [NOT] condition { command(s) } else command

New in Version 6.1

Support for long file names. All commands that use file and path names can be used with both long and short file names. Long file names are currently available on the following file systems :-

1. HPFS - High Performance File System.
2. NTFS - New Technology File System.

Use the improved VOL command to display information about a file system used for a disk drive.

Many internal commands have been re-written or improved to take advantage of the new functionality in Win32.

External commands now have a .EXC file extension. Previously, external commands had a .EXE file extension. A .EXC file extension allows external commands to be given a higher level of precedence over .COM, .EXE and .BAT files. For example, when both UNZIP.EXE and UNZIP.EXC exist and only UNZIP is entered at the WinOne prompt, WinOne will execute UNZIP.EXC.

Environment variables are now substituted on the command line. For example, enter at the WinOne prompt :-

```
ECHO %COMSPEC%
```

will display D:\WINDOWS\SYSTEM32\CMD.EXE

Improved the functionality and speed of pseudo file name completion using the TAB key. For example, quote marks are now automatically placed around file names that include space characters. See the TAB key example for more information.

New in Version 6.0

- ▣ Initial port of the 16 bit WinOne (ie. version 5.5 for Windows 3.1) to the 32 bit WinOne for Win32.
- ▣ Removed redundant and obsolete functionality from WinOne.

Glossary

The following definitions refer to commonly used terminology which is used though out the WinOne documentation.

Windows Operating System	Windows refers to either Windows NT, Windows 95 or Win32s. When it is necessary to qualify a particular operating system then either Windows NT, Windows 95 or Win32s is specified. Generally, there is no need to qualify an operating system. Typically, when a feature is not supported or operates differently under a particular operating system then a qualified name is specified.
Text Mode Type Programs	Text mode refers to all character based type programs, including DOS and console type programs. Console type programs are new to Windows NT and Windows 95. All other types of programs are referred to as non-text mode type programs. When it is necessary to qualify a particular type of text mode program then either DOS or console programs are specified. Typically, when a feature is not supported with a particular type of text mode program then a qualified name is specified.
File Names	File names refer to both long and short file names. This depends on whether an operating system supports long file names. For example, Win32s does not support long file names, where long file names are supported on Windows NT and Windows 95. Short file names are commonly known as 8.3 file names, since the file name allows at most eight characters and the file extension allows at most three characters.

Summary of Main Features

- ▣ Custom Colours and ANSI graphics.
- ▣ Full IBM graphics character set. Use the ACS command to display the full character set.
- ▣ Full edit key functionality, including the tab key (UNIX like), which expands incomplete path, file, command or macro names currently being typed at the WinOne prompt. See Command Line Edit Keys.
- ▣ More than one command can be entered on a single line. See Multiple Commands.
- ▣ Powerful command grouping using both braces and brackets..
- ▣ I/O redirection for WinOne commands and all text mode type programs. See Redirecting Command Input and Output.
- ▣ Archive file support for ZIP, LZH, ARJ, ARC. Use the ARCH command to view files inside archive files.
- ▣ Smart delete (eg. RMDIR /S and DEL /S). These commands will process sub-directories.
- ▣ Smart insertion of program files into Program Manager groups, by using the GROUP command.
- ▣ Many Extra Commands (eg. WHICH, WHERE etc.).
- ▣ Automatic directory changing. There is no need to use the CD or CDD commands, simply enter the directory at the WinOne prompt and press the return key. Also see Command Execution and Precedence.
- ▣ External Command support. External commands are simply programs that use WinOne for their input and output. External commands are written using either C or C++. See the WOIO.HLP file for more information on External Commands.
- ▣ Command line history buffer. Stores the last 64 command lines entered at the WinOne prompt. Use the HISTORY command to display a list. Also see Command Line Edit Keys.
- ▣ File Extension Associations. Associates a program(s) to a file extension. Use the EXTENSION command to manipulate file extension associations.
- ▣ Command Line Macro's. Enables sequences of commands to be grouped together. Use the MACRO command to manipulate macro's.
- ▣ Powerful enhanced Batch program support. Batch programs (ie. files having a file extension of .BAT) will be processed in the main WinOne window, unless the CALL command is used to run the batch program, then the batch program will be executed in its own separate window.
- ▣ File and directory descriptions, up to 128 characters. See the DESCRIBE command.
- ▣ User Definable Buttons. There are a maximum of 27 user definable buttons, that are fully

programmable. Use the **Buttons** option in the system menu to manipulate buttons.

- ▣ 43 predefined button images and unlimited custom images. Custom images are created from icons stored inside .EXE, .DLL and .ICO files.
- ▣ A screen buffer, which can store 100 to 1000 lines of scroll back in memory.
- ▣ Clipboard copy and paste.
- ▣ Variable Font Sizes. Set different font sizes for the WinOne Window. Use the **Fonts...** option in the system menu to change font sizes.
- ▣ Easy access to the configuration files, AUTOEXEC.BAT, CONFIG.SYS and all .INI files. Use the **System Edit...** option in the system menu. Also see System Configuration Files.
- ▣ Status Bar. Displays time, bytes free and a percentage done indicator for currently executing commands.
- ▣ Drive Bar, which displays all the disk drives along the bottom of the WinOne windows.
- ▣ WinOne can be set as the default start up shell, replacing the Program Manager.
- ▣ WinOne is mail aware. When this feature is enabled, the message "You have new mail" will be displayed when new unread mail is waiting your mail box.

Command Execution and Precedence

When a program or command is run from the WinOne prompt, there are a number of rules which resolve conflicts that can occur when there are several programs or commands that have the same file name but have a different file name extension. In order to resolve these conflicts, internal commands are given the highest precedence, otherwise when a command is entered with no file extension, then the extensions .EXC, .PIF, .COM, .EXE, .CMD and .BAT are tried, in that order.

The PATH environment variable is used to locate a command or program when it can not be found in the current directory. When a command or program can not be located, WinOne will assume a path has been specified and will attempt to change to the specified path. This process is referred to as Automatic Directory Changing. Should this fail then the error message **Bad command or file name** is displayed.

All text mode type programs and Windows programs can be executed from the WinOne prompt. When a program is executed, then the WinOne prompt will not appear until the program has completely finished executing. The user can place an executing program in the background before it has completed executing, and thereby displaying a new WinOne prompt, by pressing the control key and the z key together (ie. CTRL Z). By default, WinOne will not wait for non-text mode type programs to complete executing, before displaying a new WinOne prompt (eg. NOTEPAD.EXE etc). To change this behaviour see Program Run Options.

Many Windows or non-text mode type programs allow command line arguments to be past on the command line, similarly to text mode type programs and commands. For example, to edit a file called REPORT.TXT, enter at the WinOne prompt :-

NOTEPAD REPORT.TXT

Command Line Macro's

A Command line Macro enables a new command to be created from an original command or from a series of original commands. Command line macro's can also be given the same name as an original command. This allows a macro to be created to set an original commands default switches and parameters. For example, when WinOne is first installed there are a number of macro's that are automatically created by the install program, such as the DIR macro for command DIR. The DIR macro enables WinOne to display a sorted directory listing by default.

Command line Macro's can be created, deleted and listed using the MACRO command.

Command Line Edit Keys

The following list contains keys strokes that perform various cursor motions or actions, that can be used at the WinOne prompt to edit or modify a command line :-

Key	Function
left arrow	Move one character left.
right arrow	Move one character right.
up arrow	Display the previous command line.
down arrow	Display the next command line.
ctrl up arrow	Display the oldest command line.
ctrl down arrow	Display the most recent command line.
ctrl left arrow	Move one word left.
ctrl right arrow	Move one word right.
del	Delete one character forward.
backspace	Delete one character backward.
home	Move to beginning of line.
end	Move to end of line.
page up	Scroll up a page.
page down	Scroll down a page.
ctrl page up	Scroll up a line.
ctrl page down	Scroll down a line.
escape	Clear the command line.
shift escape	Switch to and from the <u>pop up history list</u> .
Insert	Toggle insert or overwrite mode.
ctrl insert	Copy to <u>clipboard</u> .
shift insert	Paste from clipboard.
tab	Expand an incomplete path, file, command or macro name that is currently being typed at the WinOne prompt.
ctrl tab	Display a list of names that can complete the current incomplete path, file, command or macro, that is being typed at the WinOne prompt.

Note:

Insert mode always defaults to on, when WinOne is started.

A flashing vertical bar is displayed, when insert mode is on and a flashing block is displayed when overwrite mode is on.

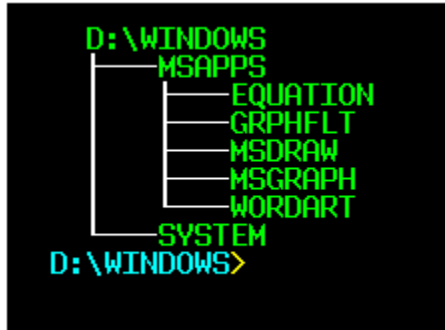
When the up or down arrow keys are used to scroll through the history list, and a partial command line has been entered at the WinOne prompt, then only the command lines in the history list that start with the same sequence of characters will be displayed.

When scrolling through the history list and there are no more commands to scroll through, then a beep will sound to signal that the end of the history list has been reached. See Event Sounds to associate a particular type of sound to this event.

Examples on using the TAB key.

TAB Key Example

The Tab key will expand an incomplete path, file, command or macro name, that is currently being entered at the WinOne prompt. For example, consider the following tree structure and default directory :-



To change to the MSAPPS sub-directory, enter at the WinOne prompt :-

```
CD MS<tab>
```

where <tab> is the tab key.

The new command line will be :-

```
CD MSAPPS\
```

When there is more than one name that can be substituted, WinOne will beep, unless the system beep has been switched off using the BEEP command, to let the user know. Press the Tab key again to select the next name. For example, assume the default directory is D:\WINDOWS\MSAPPS. To change to the sub-directory MSGGRAPH, enter at the WinOne prompt :-

```
CD MS<tab><tab>
```

After the first Tab is pressed, the command line will be :-

```
CD MSDRAW\
```

After the second Tab is pressed, the command line will be :-

```
CD MSGGRAPH\
```

Similarly, file, command and macro names can be expanding in the same way.

Wildcards can also be included in an incomplete path, file, command or macro names. For example, enter at the WinOne prompt :-

```
*.EXE<tab>
```

while in the windows directory, will be expanded to -

```
NOTEPAD.EXE
```

When a file or directory name contains space characters or any special characters then the complete file name, including the drive and directory, will automatically be enclosed in between

double quote mark characters (eg. ' " '). Typically, space characters or special characters occur in long file names.

Press the Control key along with the Tab key (ie. CTRL TAB) to display a listing of all the names found. The list is sorted alphabetically with command and macro names first, then directory names and finally file names. For example, consider :-

```
D:\WINDOWS>m
macro          mcd          mem
mkdir          more         move
[msapps]      main0.grp      make.pif
marble.bmp    moricons.dll     mountain.bmp
mouse.ini     mplayer.exe     mplayer.hlp
mplayer.ini   msd.exe         msd.ini
mtfonts.ini

D:\WINDOWS>macro
```

Command search paths

The PATH environment variable is searched for any command or program names that can be used to complete an incomplete name. When a program name is located, the file extension is removed from the file name before it is added to the list of commands.

Note:

Directory names will have a backslash character added to the end of the name, and all other names will have a space character added to the end of the name.

See Event Sounds to associate a particular type of sound to the beep emitted when there is more than one name that can be substituted.

Special Characters

There are a number of characters that have a special meaning and when encountered by WinOne the respective course of action will be taken. The special characters include :-

Character	Name	Meaning
(or)	Brackets	<u>Command Grouping</u>
{ or }	Braces	Command Grouping
&	Ampersand	<u>Command Separator</u>
"	Double Quote Marks	<u>Strings</u>
>	Greater Than Sign	<u>Redirection</u>
<	Less Than Sign	Redirection
	Vertical Bar	Redirection
^	Hat	Over-ride any special meaning attached to the next character

To over-ride any special meaning attached to any of the special characters in the above table precede the character with a hat character (ie. '^'). Similarly, to specify a hat character simply include two hat characters (eg. ECHO ^^).

When a special character appears in a string then there is no need to use the hat character, since any special meaning that may have been associated to the character is automatically ignored when the character appears in a string. Similarly, there is no need to use the hat character when specifying the expr parameter for command CALC, command LET or command IF.

Redirecting Command Input and Output

WinOne allows Standard Input and Standard Output to be redirected on the command line and assumes that Standard Input comes from the keyboard and that Standard Output goes to the screen. The default Standard Input and Standard Output can be redirected using the following characters on the command line :-

▣ The less than sign (ie. ' < ') to use the contents of a file as the input for a command. For example :-

`SORT < filename`

▣ The greater than sign (ie. ' > ') to send the output from a command to a file. When the specified file does not exist, it is created and when the file does exist, it is over written, and the previous contents are lost. For example :-

`SORT > filename`

▣ The double greater than sign (ie. ' >> ') appends the output from a command to the end of a file. . For example :-

`SORT >> filename`

▣ The bar character (ie. ' | ') allows the Standard Output from one command to be used as the Standard Input to another command. This is referred to as a pipe. For example :-

`TYPE filename | SORT | MORE`

The commands TYPE, SORT and MORE are executed in that order, that is from left to right.

Note:

When using redirection with text mode type programs, the executed task can not be placed in the background. For more information on placing a program in the background see [Command Execution and Precedence](#).

Multiple Commands

WinOne allows more than one command to be entered on a single command line. Simply separate each individual command with an ampersand character (ie. '&'). For example :-

CDD D:\WINDOWS & ECHO Current directory is D:\WINDOWS

The default ampersand separator character can be changed to any printable character in the following way :-

1. Enter at the WinOne prompt :-

SETINI WinOne CharSplit=value

where value is the ASCII character code.

2. Exit and re-run WinOne, so that the change can take effect.

Note:

For a complete list of ASCII character codes see the ACS command.

Command Grouping

Command grouping allows one or more commands to be specified in place where only a single command is allowed. To group command together enclose all the commands in brackets (ie. (and)) and separate them with the default multiple command character. There are three situations where command grouping is allowed, as follows :-

1. At the start of a command line. For example :-

```
( ECHO one&ECHO two&ECHO three )
```

2. Specifying the command parts of an IF statement. For example :-

```
IF "%VAR%"==" ( SAY nothing &ECHO found )
```

3. Specifying the command part of a FOR statement. For example :-

```
FOR %%J IN (*) DO ( SAY Found file &ECHO %%J )
```

When command grouping is used, the complete command can be split across several command lines. WinOne will concatenate command lines until the complete command is specified. WinOne determines that a command is incomplete, by checking for unbalanced brackets. Command grouping can be used either at the WinOne prompt or with in a batch program. When entering commands at the WinOne prompt and a command contains unbalanced brackets, the More? prompt is displayed, requesting more of a command. For example :-

```
D:\WINDOWS> IF "%VAR%"==" (
More? SAY nothing
More? ECHO found
More? )
```

When a command in a batch program contains unbalanced brackets, WinOne will simply continue to concatenate the next command line(s) in the batch program, until the complete command contains balanced brackets.

A command line can contain a maximum of 1023 characters. This includes any command lines that have been concatenated together.

When using environment variables with command grouping, they are instantiated as soon as the complete command is determined. This has a very undesirable effect. Consider the following batch program :-

```
SET VAR=1
IF %VAR%==1 (
    SET VAR=2
    ECHO %VAR%
)
ECHO %VAR%
```

Running the above batch program displays 1 on the first line and then displays 2 on the next line. This occurs since the first ECHO %VAR% is instantiated before the SET VAR=2 command is executed. This appears to be the standard behaviour when using brackets for command grouping (eg. CMD.EXE).

WinOne extends command grouping to allow for delayed environment variable instantiation, by grouping commands in braces (ie. { and }), instead of brackets. When braces are used in a

command line, then any environment variables that are specified inside the braces are not instantiated, until a command, inside the braces, is about to be executed. Consider the following batch program :-

```
SET VAR=1  
IF %VAR%==1 {  
    SET VAR=2  
    ECHO %VAR%  
}  
ECHO %VAR%
```

Running this batch program, produces the desired effect. That is, 2 is displays on both lines.

Note:

Command grouping using brackets and braces can be nested, repeatedly.

It is highly recommended to use braces instead of brackets to group commands together.

File Names and Wildcards

WinOne supports both long file names and wildcard characters. Long file names can only be used on a file system that supports long file names. In Windows NT this includes the NTFS and HPFS. Typically, long file names can include up to 255 characters. The VOL command displays particular information concerning a file system, such as the maximum number of characters allowed for a single file name.

Long file names can include space characters or special characters and more than one dot character. To specify a long file name that includes space characters or any special characters as a parameter to a command or program then the long file name, including the drive and directory, must be enclosed in between double quote marks (ie. ' " '). For example :-

"D:\WINDOWS\A LONG FILE NAME.TXT"

The use of double quote marks is necessary, otherwise a command would assume that many parameters have been specified when a file name includes space characters or WinOne would assume some other course of action depending on the special character encountered. Space characters and special characters are allowed for short file names or 8.3 file names, including when WinOne running on Win32s, and therefore whenever a file name contains space characters or special characters the complete file name **MUST** be enclosed in double quote marks. WinOne includes pseudo file name completion using the TAB key. When using the TAB key, double quote marks will automatically be inserted when an expanded file name contains space characters or special characters.

All long file names have an equivalent 8.3 file name, since DOS programs do NOT recognise file names that include more than 8.3 characters. For example, the short file name for "A LONG FILE NAME.TXT" is :-

ALONGF~1.TXT

There is no need to convert a long file name (as above) when the long file name conforms to the 8.3 file naming convention for DOS programs.

Wildcard characters allow a set of files that share a particular pattern to be grouped together and specified as a parameter to a command. Wildcard characters include both the star character (ie. '*') and the question mark character (ie. '?'). The star character will match zero or more characters in a file name and the question mark character will match a single character. Most users would be familiar with the use of these wildcards. For example :-

*.TXT	Specify all files that have a .TXT file extension.
.	Specify all file in the current directory.
FILE?.TXT	Specify all file such as FILE1.TXT, FILE2.TXT, FILE3.TXT etc.

Dot characters in file names are treated just like any other character when used in a file name and the file extension simply becomes the group of characters after the last dot character, that appears in a file name. Wildcards are not only matched with long file names, but they are also matched with the equivalent short file names. This can result in a long file name being accepted as matching a wildcard pattern even though it did not, since the short file name matched the wildcard pattern. Consider the following examples :-

*	Specifies all file in the current directory.
.	Specifies all file in the current directory.
..*	Specifies all file in the current directory.
.TXT.	Specifies all files that include .TXT. any where in a file name. Also includes any short file names that may match *.TXT

~ Specifies all files that include ~ any where in a file name (both long and short file names). Due to the way long file names are mapped to short file names, ***~*** may result in matching many file names that are not intended.

Similarly, wildcards can appear anywhere in a file name. For example :-

TXT Specifies files that includes the characters TXT any where in the file name, including a file names extension.

Just as long file names required short file names to work with DOS programs when a long file name exceeded 8.3 characters, it should be no surprise that certain wildcards should be avoided with DOS programs. The last two groups of wildcard examples can lead to unexpected results when used with DOS programs and should be avoided. WinOne commands, both internal and external, will function as expected, including when WinOne running on Win32s. However DOS programs do not recognise these wildcards under any operating system.

The limitations with long file names and wildcards, outlined in this section, are not unique to WinOne and they will arise with any Win32 program that use long file names and wildcards.

Automatic Directory Changing

When a path name is entered at the WinOne prompt as a command line, WinOne will attempt to change to the path specified. This process is referred to as Automatic Directory Changing, since it does not require the use of the commands CD and CDD to be used. The path name can include an optional drive and any number of nested directory names, including the use of dot characters to specify the current directory (.) and the parent directory (..). For example, assume the current directory is D:\WIN_ONE, then to change to the Windows directory, enter at the WinOne prompt :-

```
D:\WIN_ONE> ..\WINDOWS
```

the new prompt will then be :-

```
D:\WINDOWS>
```

Similarly, to change to the root directory of the current drive, enter at the WinOne prompt :-

```
D:\WINDOWS> \
```

the new prompt will be :-

```
D:\>
```

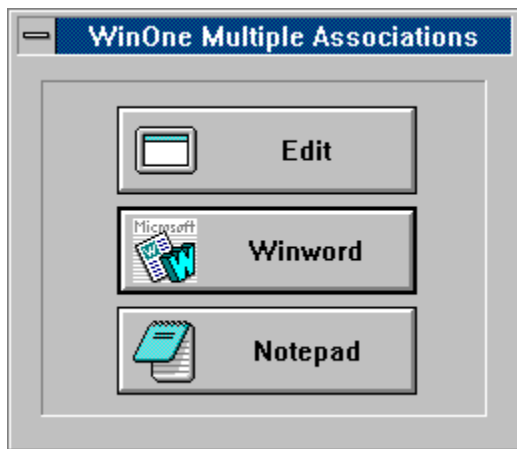
When a program has the same name as a directory, then WinOne will execute the program, instead of changing to the specified directory. To avoid this situation, simply enter an additional directory separator at the end of the path. When using the TAB key to expand a directory name, then there is no need to include an additional directory separator, since the expanded directory name will already include the extra directory separator. Similarly, expanding a directory name using the TAB key will also ensure that double quote marks are included when a directory includes the use of space characters or special characters. Generally, space characters or special characters appear in long file names.

File Extension Associations

File name associations allow the user to enter file names other than .EXC, .EXE, .PIF, .COM, .CMD or .BAT, at the WinOne prompt and provided that an association exists for the file extension, the respective program will be executed, with the filename past as a parameter.

For example, an association can be created for all .DOC files that will execute the word-processor WINWORD.EXE. To create or delete associations use the EXTENSION command.

Similarly, a single file extension may be associated with many programs. The user will be prompted to select one of the programs to execute. For example, assume an association exists between .DOC files and the three programs EDIT.COM, NOTEPAD.EXE and WINWORD.EXE. The following window is displayed :-



Simply, use the mouse and click on the button to execute the desired program.

Not all associations need to be created by the user. Many Windows programs automatically create associations for you, however text mode type programs do not, and they need to be created manually using the EXTENSION command.

ANSI Graphics Control Sequences

ANSI Control sequences are combinations of characters that can be used to control the cursor, screen and keyboard. The following ANSI Control sequences are supported by WinOne :-

Cursor Movement

ESC[H	Home cursor.
ESC[<a>;H	Move cursor to <a>, .
ESC[<a>;f	Same as for ESC[<a>;H
ESC[<a>A	Move cursor up <a> spaces.
ESC[<a>B	Move cursor down <a> spaces.
ESC[<a>C	Move cursor right <a> spaces.
ESC[<a>D	Move cursor left <a> spaces.
ESC[s	Save cursor position
ESC[u	Restore cursor position

Screen Control

ESC[2j	Clear screen
ESC[K	Erase to end of line
ESC[<n>;...;<n>m	Activate given attributes, where each <n> is
0 to 8	Sequence is Ignored
3x	Fixed Foreground colour x
4x	Fixed Background colour x
where x is	
0	Black
1	Red
2	Green
3	Yellow
4	Blue
5	Magenta
6	Cyan
7	White
5x	System Foreground colour x
6x	System Background colour x
where x is	
0	Background
1	Error message
2	Path and file name
3	Number
4	File Attribute
5	File Date
6	Bold Text
7	Text

ESC[=<n>h NOT Supported
ESC[=<n>l NOT Supported

Keyboard Control

ESC[<ns>;...;<ns>p NOT Supported

Note:

ANSI Control sequences can be entered via the PROMPT command, or by displaying a file that already contains the codes, using the TYPE command.

ANSI Control sequences can NOT be entered at the command line prompt.

The system foreground and background colours that are displayed depend on the Custom Colours and Colour Schemes set by the user.

Batch Programs

A batch program is a simple text file that contains a sequence of programs or commands to run. In order to distinguish a batch program from an ordinary text file, batch programs have a file name extension of .BAT. Any text editor (eg. NOTEPAD) can be used to create and write a batch program. Batch programs will be displayed inside the main WinOne window, unless the CALL command is used to run the batch program, then the batch program will be displayed in a separate window or if an appropriate PIF file exists then it may be run full screen. For more information on PIF files see the help file for the program PIFEDIT. When a batch program is run so that it is displayed in the main WinOne window, then the complete batch program is loaded into memory before the execution of the commands in the batch program starts. This greatly improves the performance and speed of batch programs.

There are a number of batch commands that can be used in a batch program to enhance the capabilities of batch programs. The batch commands, listed below, are separated into two categories, standard and/or enhanced. Standard batch commands are compatible with CMD.EXE, where enhanced batch commands are native to WinOne and therefore they are not supported by CMD.EXE. Typically, when a batch program is destined to run under different command shells then the enhanced batch commands should not be used. When this requirement is not necessary then the use of enhanced batch commands will greatly improve the usefulness of batch programs.

Standard Batch Commands

- ▣ CALL Run a second batch program, then return to the first batch program.
- ▣ ECHO Display a message or turns echo on or off.
- ▣ ENDLOCAL End the localisation of environment variables.
- ▣ FOR Perform a command for each file in the specified set of files or perform a command a given number of times.
- ▣ GOTO Switch to another part of the batch program, and continue executing the program from that point.
- ▣ IF Perform conditional processing in a batch program.
- ▣ PAUSE Suspend processing of a batch program and display a message.
- ▣ REM Allows comments inside a batch file.
- ▣ SETLOCAL Set the localisation of environment variables.
- ▣ SHIFT Change the position of replaceable parameters in a batch program.

Enhanced Batch Commands

- ▣ ASK Ask a yes/no question and set the errorlevel respectively.
- ▣ BEEP Send a beep to the system speaker.
- ▣ BOX Display a box in one of four pre-defined formats.
- ▣ CALC Perform basic arithmetic calculations in a batch program. The result is stored in an environment variable.
- ▣ COLOUR Change foreground and background screen colours.
- ▣ COMMA Insert comma's into a number. The comma delimited number is stored in an environment variable.
- ▣ DATA Clear or add items to a global list
- ▣ DATE2SER Convert a date to a serial date value. The serial date value is stored in an environment variable.
- ▣ DIRS Display the directory stack.
- ▣ DISKFREE Determine the amount of free disk space for a disk drive. The free disk space is stored in an environment variable.
- ▣ DISKUSED Determine the amount of used disk space for a disk drive. The used disk space is stored in an environment variable.

- ▣ END End a batch program.
- ▣ FILEDATE Get the last modified date of a file. The file date is stored in an environment variable.
- ▣ FILESIZE Get the size in bytes of a file. The file size is stored in an environment variable.
- ▣ FILETIME Get the last modified time of a file. The file time is stored in an environment variable.
- ▣ FILETYPE Determine whether a file contains text or binary data. The file type is stored in an environment variable.
- ▣ GETINI Get an initialisation file key value or enumerate all the key names in a section. The value is stored in an environment variable.
- ▣ GETKEY Wait for a single keypress from the user. The character is stored in an environment variable.
- ▣ GETNUM Wait for a sequence of keypresses from the user. The sequence of characters are stored in an environment variable as a string of characters.
- ▣ GETSTR Wait for a sequence of keypresses from the user. The sequence of characters are stored in an environment variable as a string of characters.
- ▣ GETREG Get a System Registry value or enumerate all the names for the specified hkey. The value is stored in an environment variable.
- ▣ GOSUB Jump to another part of a batch program, and continue executing from that point until RETURN is encountered.
- ▣ LOCATE Position the cursor anywhere on the screen.
- ▣ LOWER Convert a text string to lower case. The converted text string is stored in an environment variable.
- ▣ PARSE Allows a sentence to be broken into pieces. The pieces are stored in environment variables.
- ▣ PLAY Play a wave sound file.
- ▣ POPD Pop a directory from the directory stack and make this directory the current directory.
- ▣ PUSHD Push the current directory onto the directory stack and change to the specified directory.
- ▣ READ Read the next item from the global list. The item is stored in an environment variable.
- ▣ READLN Read a line of text from a file. The line of text is stored in an environment variable.
- ▣ RETURN Return execution to the next command following the GOSUB command.
- ▣ SAY Display a message. This command with not add a carriage return - line feed at the end of the message.
- ▣ SER2DATE Convert a serial date value to a date. The date is stored in an environment variable.
- ▣ SETINI Set or delete an initialisation file key value.
- ▣ SETREG Set or delete a System Registry value.
- ▣ SLEEP Do nothing for a time.
- ▣ STOP Stop processing a batch program and continue processing the batch program that called this one.
- ▣ STRFIND Find the first occurrence of a string within the specified text. The index where the string occurs is stored in an environment variable.
- ▣ STRPAD Pad a text string with space characters. The padded text string is stored in an environment variable.
- ▣ STRREP Replace all occurrences of a string within the specified text with another string. The resulting string is stored in an environment variable.
- ▣ STRREV Reverse all the characters in a text string. The reversed string is stored in an environment variable.

- ▣ STRRFIND Find the last occurrence of a string within the specified text. The index where the string occurs is stored in an environment variable.
- ▣ STRSIZE Determine the length of a string. The length is stored in an environment variable.
- ▣ STRTRIM Remove leading and trailing space characters from a text string. The resulting string is stored in an environment variable.
- ▣ SUBSTR Extract a section of text from a text string. The extracted text string is stored in an environment variable.
- ▣ UPPER Convert a text string to upper case. The converted text string is stored in an environment variable.

Environment Variables

Environment variables can also be used inside batch programs and have the following format %name%. For example, %COMSPEC% inside a batch program will be replaced with D:\WINDOWS\SYSTEM32\CMD.EXE. This value is system dependent and may vary from system to system. Environment variables can be manipulated using the SET, XSET or LET commands.

Multi-dimensional Environment variables

Multi-dimensional environment variables have the following syntax :-

NAME.INDEX1.INDEX2.INDEX3....

where NAME is the name of the environment variable and INDEX1, INDEX2 and so on specifies the index values, which may either be an integer value or the name of another environment variable which contains an integer value. Consider the following example, where the index value is specified by the contents of another environment variable :-

```
SET idx=0
SET array.idx=10
ECHO %array.idx%
```

When WinOne encounters the environment variable array.idx then the idx part is replaced with the contents of the idx environment variable, mapping the final name to array.0. The environment variable array.0 is then created and assigned the value 10. The ECHO command will display the value 10. Continuing from the above example :-

```
LET idx=%idx% + 1
SET array.idx=20
ECHO %array.idx%
```

Now the environment variable idx will contain the value 1 and the environment variable array.1 is created and assigned the value 20. The ECHO command will display the value 20. Continuing from the above example :-

```
FOR %%i is 0 to 1 DO ECHO %array.%%i%
```

the FOR loop is identical to the following :-

```
ECHO %array.0%
ECHO %array.1%
```

and will display the value 10 on the first line and the value 20 on the second line

Dynamic Environment Variables

Dynamic environment variables are environment variables that contain values that are determined dynamically, that is, at the time they are instantiated. The following dynamic environment variables are allowed :-

```
ERRORLEVEL          Last errorlevel set
```

DATE_FORMAT	System date format string
DATE_TODAY	Current date (formatted according to DATE_FORMAT)
DATE_YEAR	Current year (yyyy)
DATE_MONTH	Current month (mm)
DATE_DAY	Current day (dd)
DATE_DAYOFWEEK	Current day of the week (0=Sun)
TIME_TODAY	Current time (hh:mm:ss in 24 hour format)
TIME_HOUR	Current hour (hh as 24 hour)
TIME_MINUTE	Current minute (mm)
TIME_SECOND	Current second (ss)
DISK_DRIVE	Current disk drive letter (A, B, C ...)
DISK_PATH	Current working directory.
SCREEN_WIDTH	Current screen width
SCREEN_HEIGHT	Current screen height
WHERE_X	Horizontal cursor location (starting from 0)
WHERE_Y	Vertical cursor location (starting from 0)
WINDOWS_VERSION	Windows version number
WINONE_VERSION	WinOne version number
WINONE_PATH	WinOne home directory
WINONE_FILENAME	Full path of the WinOne executable
OPERATING_SYSTEM	Operating system (WinNT, Win32s or OTHER)
RANDOM_NUMBER	Random number (0 to 65534)

To use these dynamic environment variables, enclose the name in between percentage characters. For example to display the system date format, enter at the WinOne prompt :-

```
ECHO %DATE_FORMAT%
```

The system date format string displayed is system dependent and may vary from system to system.

Replaceable Parameters

Batch programs can be passed parameters on the command line. These parameters can be referenced by using replaceable parameters. There are a maximum of 10 replaceable parameters allowed, specified by %0 through to %9 inside a batch program and the special cases %# which returns the number of parameters passed to the batch program and %* which returns the batch programs original command line tail. For example, consider the batch program below called PARAMS.BAT. This batch program displays the total number of parameters, along with the first 10 parameters passed to the batch program :-

```
@ECHO OFF
ECHO %* is %*
ECHO %# is %#
ECHO.

LET num=0
FOR %%i in (%0,%1,%2,%3,%4,%5,%6,%7,%8,%9) do {
    ECHO %num% is %%i
    LET num=%num% + 1
}
```

Running the above batch program from the WinOne prompt :-

```
PARAMS abc 123
```

will display the following :-

```
%* is abc 123
```

%# is 2

%0 is PARAMS

%1 is abc

%2 is 123

Debugging a Batch Program

Debugging a batch program involves being able to control its execution in order to locate and fix problems that may arise while writing a batch program. WinOne supports single step, running, stopping and environment variable manipulation while a batch program is being debugged. See [Batch Program Debugging](#) and the [TRACE](#) command to debug a batch program.

Virus Protection

When WinOne is modified in any way, a window is displayed informing the user that WinOne has been modified and it could be a virus. WinOne will then terminate :-



Startup Batch Program

Every time WinOne is started, WinOne will look for a STARTUP.BAT batch program, in the same directory in which WinOne is installed and if found the batch program will be executed before the WinOne prompt is displayed. The STARTUP.BAT batch program can contain anything that can be included in a batch program, there are no special exceptions.

Note:

Also see [Batch Programs](#) for more information on writing a batch program.

WinOne Parameters

When WinOne is run from the Program Manager or any other program then any command line parameters specified will be displayed at the WinOne prompt, after WinOne has started, and executed. These parameters can include any thing that can be entered at the WinOne prompt, this includes File Extension Associations and Macro's. The syntax for WinOne is as follows :-

WIN_ONE [**USE=filename**] **/C** **/K** parameters ...]

parameters	Specifies the command(s) to run after WinOne has started.
filename	Specifies the file name of an initialisation file.
/C	Perform a command and exit.
/K	Perform a command and continue.

WinOne stores all its configuration information in the default initialisation file WIN_ONE.INI, which is automatically mapped to the System Registry when WinOne is running under Windows NT. To specify a different initialisation file include the "USE=" parameter. A new automatic mapping is created in the System Registry for the specified file name and the following System Registry key is created and used by WinOne :-

HKEY_CURRENT_USER
\Software
\filename

Specifying the switch /K is not necessary since this has the same effect as not specifying any switches and is included for compatibility with CMD.EXE.

Generally, WinOne will be started from the WinOne icon in the Program Manager. To specify any parameters for the WinOne icon in the Program Manager, firstly, select the **Properties...** option from the File menu in the Program Manager. Make sure that the WinOne icon is selected and highlighted before selecting the **Properties...** option. Secondly, after the Properties window is displayed for the WinOne icon, simply added any parameters to the end of the command line that the Program Manager will run to start WinOne.

For example, to set WinOne to have a high process priority class when started, then use the TASKS command and run WinOne as follows :-

WIN_ONE TASKS HIGH

Note:

Also see the Startup Batch Program.

Multiple Instances

WinOne allows more than one copy of itself to be run at any one time. Running more than one copy of WinOne at any one time is supported on Windows NT and Windows 95, but not when WinOne is running on Win32s, since Borlands compiler does not support the loading of a Dynamic Link Library (ie. a .DLL file) more than once in Win32s.

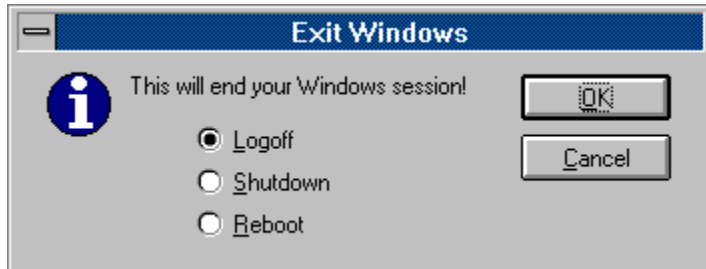
Multiple User

WinOne supports multiple users on Windows NT, that is, running a single installed version of WinOne on a computer for all users on that computer. Each user is able to configure WinOne to suit their own individual needs without conflicting with any other WinOne configurations that may already be established for other users. This is possible since WinOne uses the System Registry to store all of its configuration information.

This feature is not supported in Windows 95 and Win32s since both do not support multiple users. Also, when WinOne is running on Windows 95 or Win32s then WinOne will use the initialisation file WIN_ONE.INI, located in the windows directory.

Program Manager Replacement

WinOne can replace the default Program Manager supplied with Windows. When WinOne is set as the default start up shell then WinOne will run all the programs that would normally be run by the Program Manager, every time Windows is started. Exiting from WinOne will exit Windows after the following message is displayed :-



Simply use the mouse to select one of the three exit options (ie. Logoff, Shutdown or Reboot). A small dot will appear next to the option text when selected. Press the OK button to perform the selected option.

Setting WinOne as the default Windows shell

WinOne can be set as the default start up shell as follows :-

1. Enter at the WinOne prompt :-

SHELL WIN_ONE,WOWEXEC

2. Logout and then log back into Windows so that the change to take effect.

Before setting WinOne as the default start up shell make a note of the current shell setting by entering at the WinOne prompt :-

SHELL

Warning

The Program Manager must NOT be replaced :-

1. unless there is sufficient memory to run other programs. At least 16 Megabytes of on-board memory is recommended.
2. on an operating system other than Windows NT. Under NO circumstances should WinOne be set as the default start up shell in Windows 95, Win32s or any other operating system, other than Windows NT.

Note:

Also see command SHELL and Program Manager Programs.

File Drag and Drop

WinOne supports Drag and Drop for any program that can drag files (eg. the File Manager). One or more file or path names can be dragged into the main WinOne window, and when dropped, the file or path names will be added to the end of the WinOne command line.



For example, to drag a file from the File Manager, simply position the mouse cursor over the file to drag, press and hold down the left mouse button, drag the file until the mouse cursor is inside the main WinOne window and release the left mouse button to drop the file. During the dragging process the cursor will change to one of the following, as it is moved around the desktop :-



the window can accept the file.



the window can accept the files.



the window can not accept the file(s).

File names that are dropped into the main WinOne window, will have a space character added to the end of the file name and path names will have a backslash character added to the end of the path name.

To drag one or more files from WinOne to another program that accepts a file drop (eg. the Program Manager) use the DRAG command.

Clipboard Manipulation

The system menu contains the **Edit** option, which when selected will display a sub-menu containing the several options, as below :-

A tributes		
E dit	C opy	C trl+ I ns
B uttons	P aste	S hift+ I ns
D rives	L oad...	
S tatus b ar	S ave a s...	
P rograms	C lear	
S ystem edit...		

Marking a region



Use the mouse cursor and position it over the first character to mark. Press and hold down the left mouse button. While holding down the left mouse button drag the cursor over the last character to mark and release the left mouse button. The marked region is displayed in reverse colours.

Similarly, to mark a word, simply position the mouse cursor over the word to mark and double click the left mouse button.

To clear a region that is marked on the screen, position the mouse cursor any where on the screen and press the left mouse button once.

WinOne only allows a region to be marked and copied to the clipboard when WinOne is waiting for a key stroke or a series of key strokes to be entered.

Copying to the Clipboard

Having marked a region on the screen, there are several ways to copy it to the clipboard :-

1. Select the **Edit** option in the system menu, and then select **Copy**.
2. Press the Control key and the Insert key together (ie. CTRL INS).
3. Press the right mouse button.

When a region is successfully copied to the clipboard the highlight is cleared from the screen.

Pasting from the Clipboard

Text that has been copied to the Clipboard can be pasted to the command line or any command that is waiting for a input. There are several ways to paste from the clipboard :-

1. Select the **Edit** option in the system menu, and then select **Paste**.
2. Press the Shift key and the Insert key together (ie. SHIFT INS).
3. Press the middle mouse button.
4. Double click the right mouse button. When a region is marked on the screen, the region is first copied to the clipboard, then the clipboard contents are pasted to the command line, otherwise, when there is no region marked on the screen, only the clipboard contents are pasted to the command line.

Saving and Loading the clipboard contents

The contents of the clipboard can be saved to or loaded from any of the following standard file types :-

1. Clipboard file (.CLP)
2. Bitmap file (.BMP or .DIB or .RLE)
3. Plain text file (.TXT)

Select the **Edit** option in the System menu, and then select **Save...**, to save the contents of the clipboard to a file, or select **Load...**, to load the contents of a file into the clipboard.

Clearing the Clipboard Contents

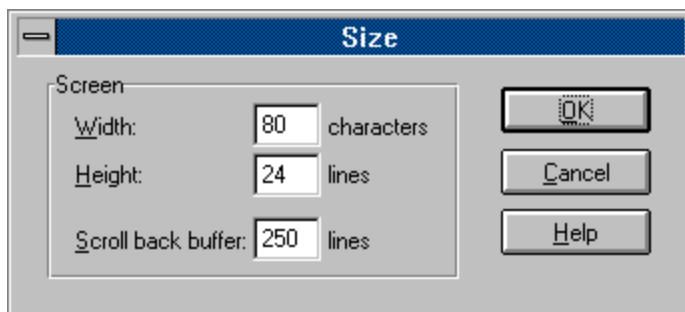
Select the **Edit** option in the System menu, and then select **Clear**, to clear the contents of the clipboard, and release all the memory that was used for the contents in the clipboard.

Screen Size

The system menu contains the **Attributes** option, which when selected will display a sub-menu containing the **Size...** option, as below :-

A tttributes	S ize...
E dit	F onts...
B uttons	C olours...
D rives	S ounds...
S tatus b ar	K eys...
P rograms	U nix mode
S ystem edit...	M ail
	R un...

When selected the Size window is displayed, where the screen width, height and the number of lines for the scroll back buffer can be set :-



Screen Width and Height

Simply enter the desired width and/or height for the screen and press the OK button. The minimum allowable size for the screen is 60 characters across (width) by 5 lines down (height). The maximum screen size is dependent on the size of the desktop and the size of the fonts used to display text.



Alternatively, WinOne also allows the screen to be resized by dragging any of the borders of the WinOne window to the desired size.

Scroll Back Buffer

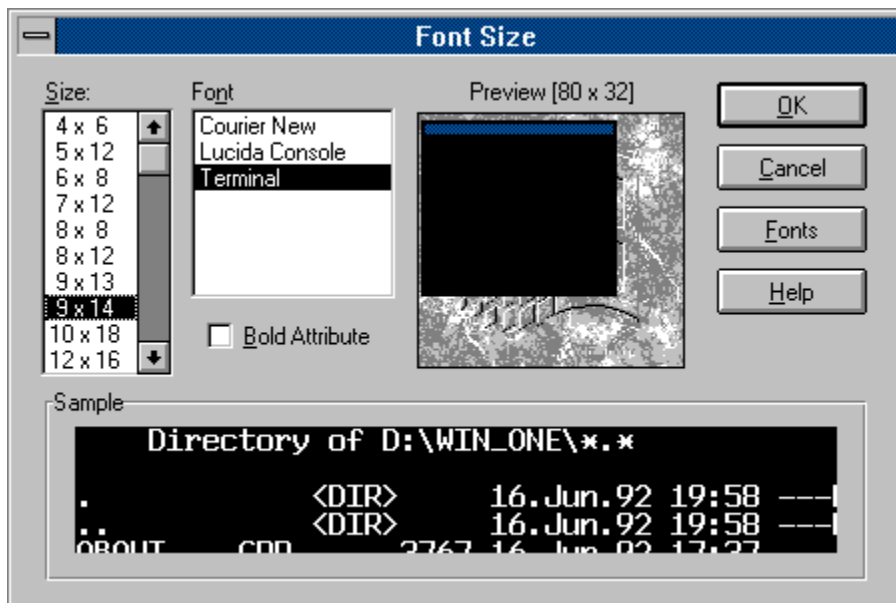
The scroll back buffer stores any lines of output that may scroll out of view. Simply enter the desired number of lines for the scroll back buffer and press the OK button. By default WinOne stores 250 lines of output. The minimum scroll back buffer is 100 lines and the maximum is 1000 lines.

Variable Font Sizes

The system menu contains the **Attributes** option, which when selected will display a sub-menu containing the **Fonts...** option, as below :-

A tttributes	S ize...
E dit	F onts...
B uttons	C olours...
D rives	S ounds...
S tatus b ar	K eys...
P rograms	U nix mode
S ystem edit...	M ail
	R un...

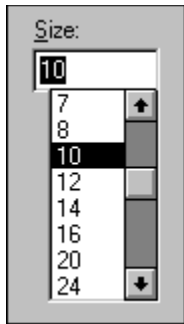
When selected the Font Size window is displayed, containing a list of font sizes, font types, bold attribute, a window preview, showing the WinOne window size with respect to the desktop, and a sample of the selected font size :-



There are 3 font types, that are supplied with Windows :-

1. Courier New
2. Lucida Console (Windows NT 3.51 or higher)
3. Terminal

Both Courier New and Lucida Console are TrueType fonts. TrueType font sizes are specified using point sizes, where one point represents 1/72 of an inch in height :-



TrueType fonts do not include the full ASCII graphics character set and when used with commands such as BOX then the results will be less than desirable.

Terminal fonts are commonly referred to as Raster fonts. Raster fonts are not scalable, unlike TrueType fonts, but Raster fonts do include the full ASCII graphics character set. There are 10 standard Terminal font sizes that are supplied with Windows :-

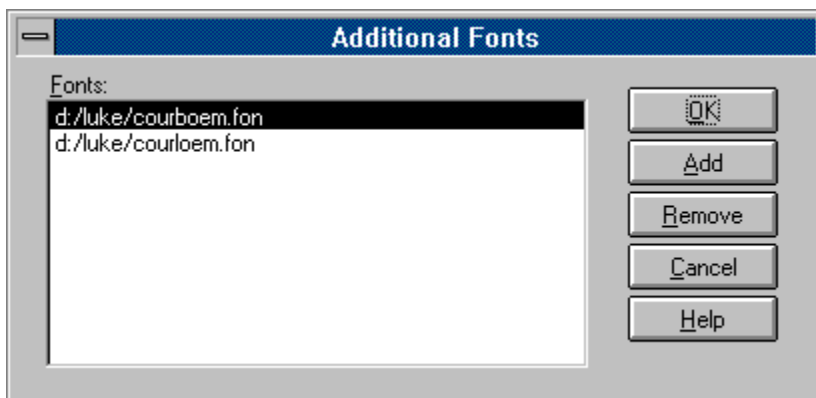
4 x 6
5 x 12
6 x 8
7 x 12
8 x 8
8 x 12
10 x 18
12 x 16
16 x 8
16 x 12



Simply use the mouse to select the font in the Font list box and the desired font size in the Size list box and press the OK button.

Additional Fonts:

WinOne allows a maximum of 8 additional fonts to be added to the above lists of fonts. Only fonts that are for Windows DOS applications (ie. Terminal font types) can be added. Press the Font button to display the Additional Fonts window, as below :-



To add a new font to the list, simply press the Add button to display the Font Select window, where an individual font file may be located. Continue and repeat this process until all the desired font files appear in the list and then press the OK button to accept the new fonts. Exit and restart

WinOne so that the changes can take effect. Similarly, to remove a font from the list, simply select the desired font file in the list and press the Remove button.

The first picture shows two addition font sizes (ie. 9 x 13 and 9 x 14).

Custom Colours and Colour Schemes

The system menu contains the **Attributes** option, which when selected will display a sub-menu containing the **Colours...** option, as below :-

A tributes	S ize...
E dit	F onts...
B uttons	C olours...
D rives	S ounds...
S tatus b ar	K eys...
P rograms	U nix mode
S ystem edit...	M ail
	R un...

When selected the Colour window is displayed, showing the Text drop down list, containing all the customizable attributes, the colour palette, and the predefined colour schemes drop down list :-



Changing Colours

The Text drop down list contains all the attributes for which the colours can be set and include :-

- File or path names
- Highlighted file or path names
- Numbers
- Plain text
- Highlighted text
- Bold text
- Environment names
- Environment strings
- Error messages
- Left hand side of equal sign
- Highlighted left hand side of equal sign
- Right hand side of equal sign
- File dates
- File times
- File attributes
- File descriptions
- Background



Simply select the attribute to change, then select the colour to associate with the attribute. Continue this process until all the desired colours have been set. Then press the OK button to accept the new colours. WinOne will automatically save the new colours and all output from all WinOne commands will then use these colours.

Colour Schemes

The Scheme drop down list contains a number of predefined colour schemes, as follows :-

- Black on White
- Cloudy Day
- Default
- Emerald Forest
- Gray on Black

Ice
Metallic
Ocean
Sun Set
White on Black



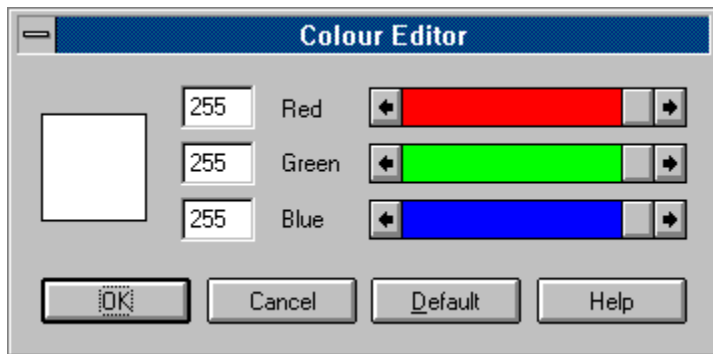
Simply select the desired colour scheme and press the OK button to accept the new colour scheme.

Colour Editor

When the screen is capable of displaying 256 colours or more then the Red, Green and Blue values for the default 16 colours used by WinOne may be individually set.



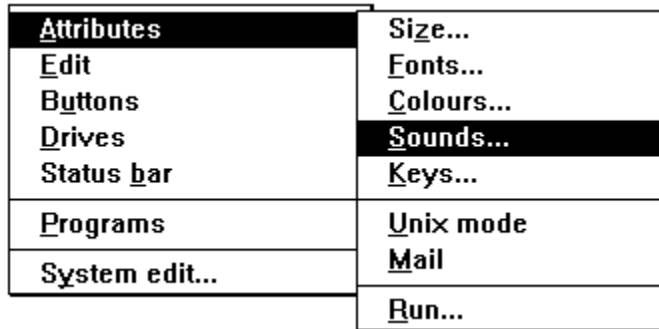
From the Colour window simply position the mouse pointer over the colour in the Palette to edit and while holding down the Control key press the left mouse button. This will display the Colour Editor window for the respective colour, as follows :-



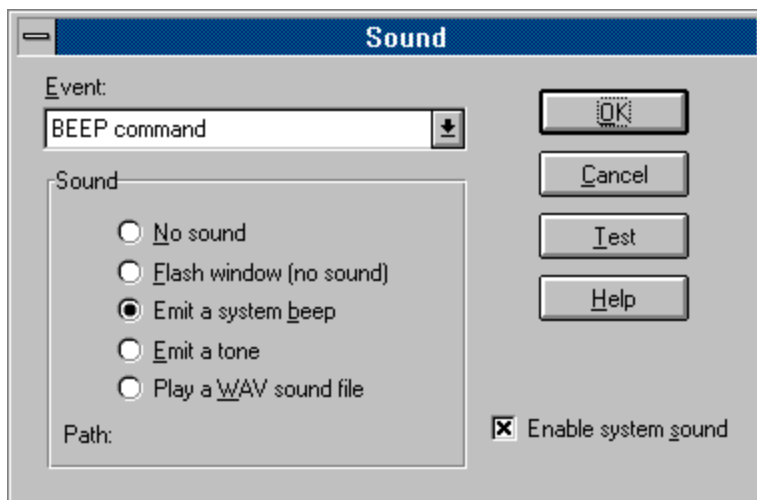
Simply set the desired Red, Green or Blue colour values and press the OK button to accept the changes. To reset the colour to the default system colour press the Default key.

Event Sounds

The system menu contains the **Attributes** option, which when selected will display a sub-menu containing the **Sounds...** option, as below :-



When selected the Sound window is displayed containing the Event drop down list and a group of all the different sounds that can be selected for an event :-



The Event drop down list contains all the events that can be associated to a particular type of sound and includes :-

1. BEEP command.
2. Beep hourly
3. Beep half hourly
4. 1st alarm
5. 2nd alarm
3. File expansion TAB key.
4. History UP/DOWN key.

There are a number of sounds that can be associated to any of the above events, as follows :-

1. Play a WAV sound file.
2. Emit a tone.
3. Emit a system beep.
4. Flash window.
5. No sound.

When associating the "Play a WAV sound file" option to a particular event a window will be displayed in which a wave sound file may be chosen. Generally, a wave sound file has a file extension of .WAV. There are a number of wave sounds files that are included with Windows and these are located in the Windows directory. When no sound board is installed, then playing a wave sound file will result in no sound being played or heard.

Setting Sound Events



Simply select the desired event from the Event drop down list and select the sound to associate to the event. Continue this process until all the desired events have been set and then press the OK button to accept the changes.

Enabling the system sound

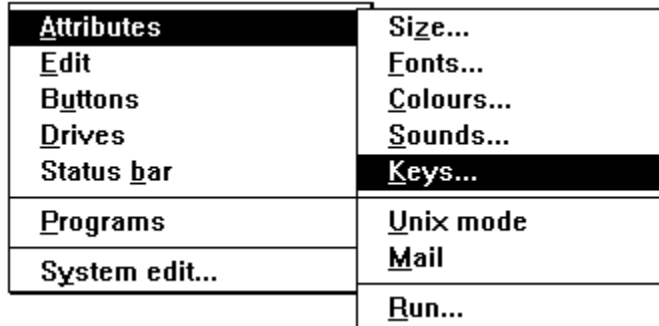
Enabling or disabling the system sound only affects the "Emit a system beep" option. When the system sound is disabled then no sound will be played or heard for the system beep. The system beep is used by Windows as a audio warning signal and is generally sounded when a message window is displayed. When the system sound is enabled then a small tick mark will appear next to the option text, similarly, when the system sound is disabled then no tick mark is displayed next to the option text. Changes to the system sound are reflected system wide.

The Test button

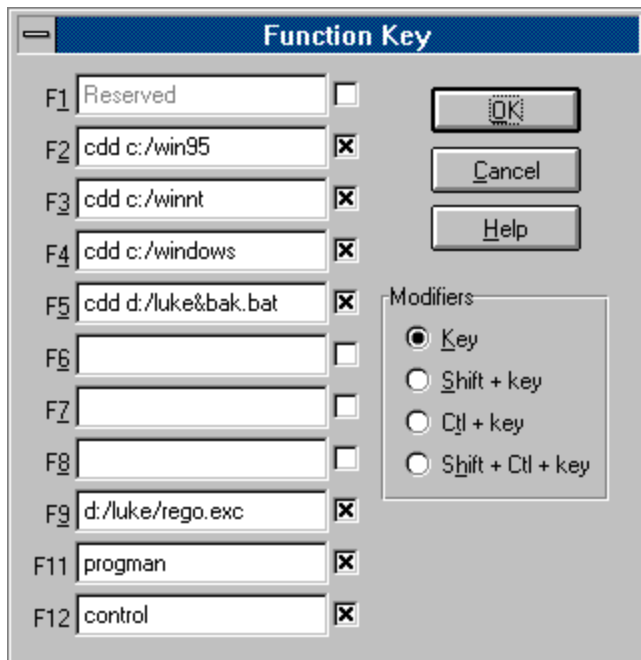
Press the Test button to listen to a sample of the actual sound that will be heard when the event occurs.

Function Keys

The system menu contains the **Attributes** option, which when selected will display a sub-menu containing the **Keys...** option, as below :-



When selected the Function Key window is displayed in which the text for the function keys F1 to F12 (excluding F10) may be set, as below :-



The F10 function key is reserved by Windows and pressing the F10 function key will activate the system menu. Similarly, the F1 function key (no modifiers) is reserved by WinOne for Context Sensitive Help.

Function Keys

The function keys can include any text string. When a function key is pressed the respective text string is inserted at the WinOne prompt.

Modifiers

Allows the text for functions keys to be set for the respective modifier keys (ie. the Shift and Control keys).

Appending a Carriage Return

To append a carriage return character to the end of the text string associated with a function key

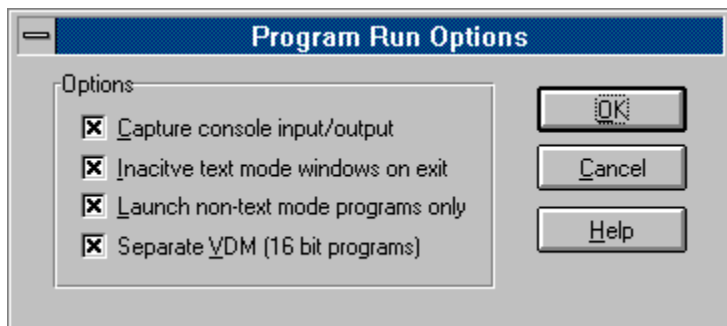
simply select the small check box next to the respective function key edit field. When a cross is displayed, then the carriage return character will be appended to the end of the text string, otherwise when no cross is displayed then no carriage return character is appended to the end of the text string.

Program Run Options

The system menu contains the **Attributes** option, which when selected will display a sub-menu containing the **Run...** option, as below :-



When selected the Program Run Options window is displayed :-



When a tick mark appears next to the options text (as above), then the option is switched on, otherwise, the option is switched off. Simply position the mouse pointer over the option text and press the left mouse button to toggle the option either on or off. The tick mark will either be shown or removed, respectively.

Capture console input/output

Console programs refer to text mode type programs that use the Windows console Application Programming Interface (API). Most standard DOS commands have been converted to console programs in Windows NT. For example FORMAT.COM under Windows NT is a console program.

This option, when switched on, allows console programs to use the main WinOne window for their input and output, instead of creating their own windows.

Not all console programs can use the main WinOne window for their input and output. See command CAPTURE for more information.

Inactive text mode windows on Exit

When a text mode type program is run in its own separate window, either the window will be left open or the window will be automatically closed when the program has finished executing.

This option, when switched on, will leave the window open. The windows title will be changed to include the word "Inactive", signalling that the program has completed executing and may be closed manually at any time.

To set this option from the command line see the DOS command.

Launch non-text mode programs only

Non-text mode programs refer to programs that use the Windows Graphical User Interface (GUI). These programs include NOTEPAD.EXE, WIN_ONE.EXE etc.

This option, when switched on, will only launch non-text mode programs and return the WinOne prompt immediately, ready for the next command. Otherwise, when the option is switched off, WinOne will wait for non-text mode programs to finish and exit before the WinOne prompt is returned. A program can still be placed into the background manually. See Command Execution and Precedence for more information on placing a program into the background manually.

Separate VDM (16 bit programs)

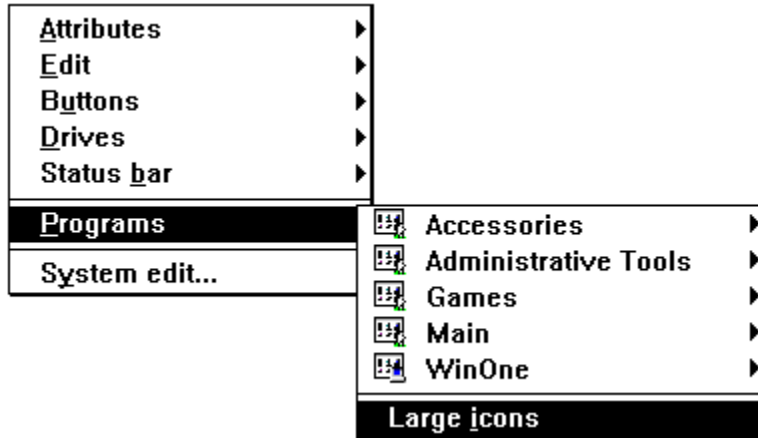
This option, when switched on, will allow all 16 bit non-text mode programs to run in their own virtual environment, so that these programs will preemptively multitask. Otherwise, when this option is switched off, then all 16 bit non-text mode programs will run in a single environment using a cooperative multitasking mechanism.

To individually run programs in a separate VDM see the START command.

This option is only available on Windows NT 3.5 or higher.

Program Manager Programs

The system menu contains the **Programs** option, which when selected will display a sub-menu containing the Program Manager groups, as below :-



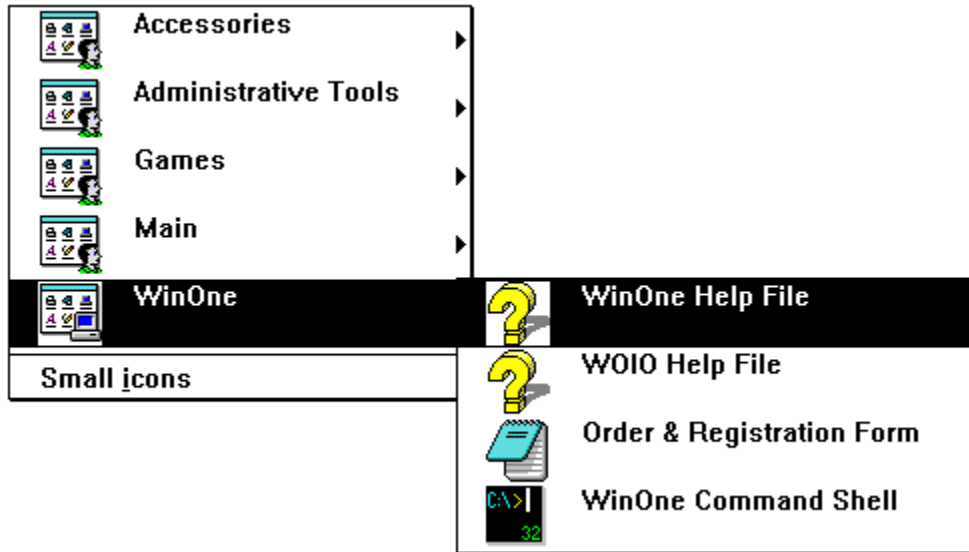
Using large or small icons

The option Small icons will display item icons half size. Similarly, the option Large icons will display item icons actual size :-



Running a program

Select the desired Program Manager group to display a new sub-menu containing all the program items in the group. Simply select one of the items to run the respective program.



Group Icons

The following Program Manager group icons are used for the respective operating systems :-



Windows NT (Common groups)



Windows NT (Personal groups)



Win32s



Windows 95

Note:

When a group does not contain any program items, then that group will be excluded from the menu.

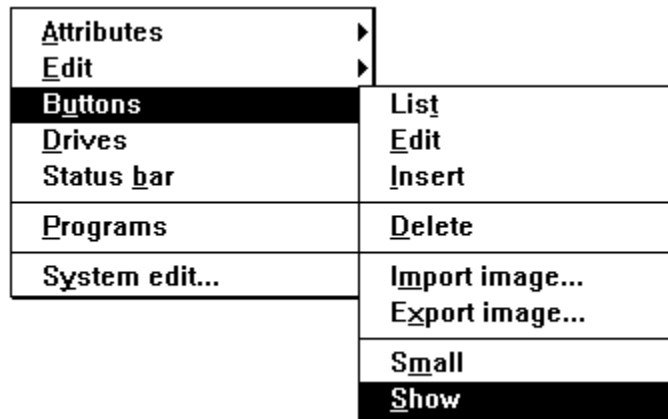
To manipulate Program Manager groups and programs, see the command GROUP.

User Definable Buttons

User Definable Buttons enable the user to associate a command line with a button. When the button is pressed then the specified button command line is executed. There are three system buttons that can NOT be modified in any way (ie. the red Exit button, the blue Shell button and the green Help button), and 27 additional user buttons that can be modified (ie. the yellow buttons) :-



The system menu contains the **Buttons** option, which when selected will bring up a sub-menu, as below :-



Displaying the Buttons

The option **Show** will display the buttons at the top of the WinOne window.

Hiding the Buttons

The option **Hide** will remove the buttons from the top of the WinOne window.



Displaying Small Buttons

The option **Small** will display small buttons at the top of the WinOne window.

When the mouse pointer is positioned over a small button for 2 seconds the button text will be displayed in a small yellow window as below :-



Displaying Large Buttons

The option **Large** will display large buttons at the top of the WinOne window.



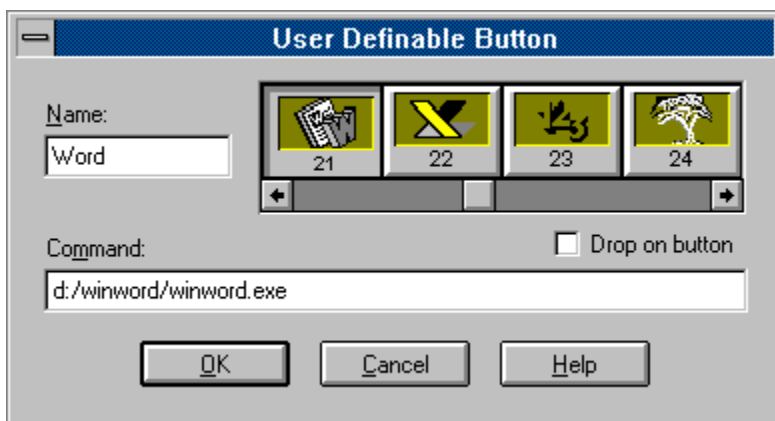
The sub-menu also contains the following additional options :-

1. List - Displays a list of the commands associated with each button.
2. Edit - Modify an existing button.
3. Delete - Remove an existing button.
4. Insert - Add a new button.
5. Import image - Add or import a new button image.
6. Export image - Delete an imported button image.



When editing, inserting or deleting a button, a message is displayed, informing the user that the buttons have been marked to perform the specified function. Also, the cursor will be changed to a hand cursor. Then the next time a button is pressed, using the mouse, the specified function will be carried out. Type ^C at the WinOne prompt to cancel the operation. For example, after the buttons have been marked to insert, press the red Exit button to insert a new button before it.

When editing or inserting a button, a window will appear, where the button name, command line and the image for the new button is specified. The images displayed includes all the default images, as well as the imported images. The button command line can include anything that can be entered at the WinOne prompt, this includes File Extension Associations and Macro's.



Drop On Buttons

Drop On Buttons allows a file(s) or a directory to be dropped on to them (eg. from the File Manager) and the respective button command will be executed, after the file(s) or directory has been added to the end of the button command. Use the Drop On Button check box to set or reset drop on capabilities. The button name will be displayed in italics in the main WinOne window to enable them to be distinguished from non-drop on buttons :-

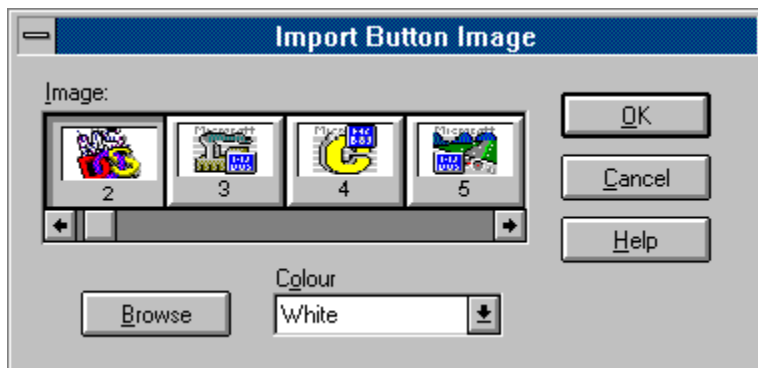


Button Images

There are 43 default images that can be displayed inside a button, as follows :-

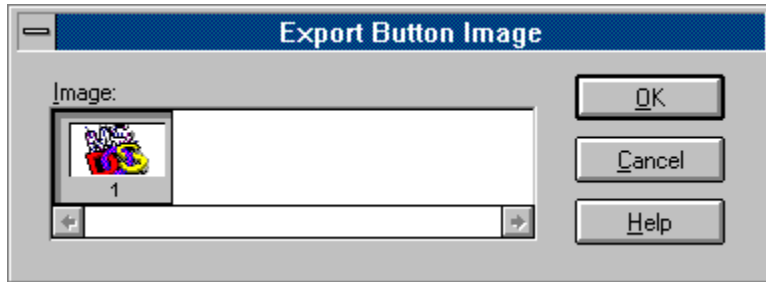


Importing button images allows custom images to be displayed inside a button. Images are created from icons that are stored inside files that have an extension of .EXE, .DLL or .ICO. To import a button image select the **Import image...** option from the button menu. The following window is displayed when importing a button image :-



Press the Browse button to select a file to extract the images from. After a file has been selected then all the images inside the file will be displayed inside the list box. Simply select the image to import from this list box and press the OK button. The image will then be added to the end of all the default button images. The Colour drop down list allows the background colour of the buttons, contained in the list box, to be set. By default the buttons will have a yellow background colour.

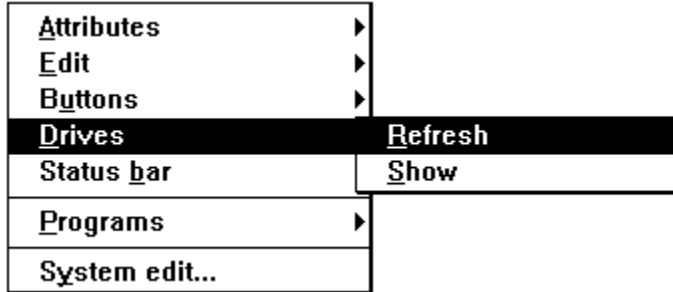
To remove an imported button image select the **Export image...** option from the button menu. The following window is displayed ;-



Simply select the image to export from the list box and press the OK button.

Drive Bar

The system menu contains the **Drives** option, which when selected will display a sub-menu containing several options, as below :-








Displaying the Drive Bar

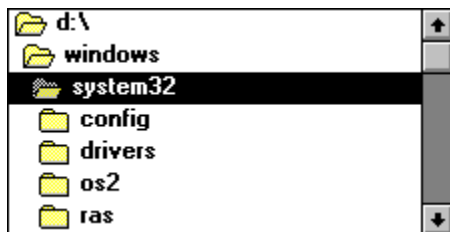
The option **Show** will display the drive bar at the bottom of the WinOne window, showing all the known disk drives.




The following images are displayed on the drive bar to indicate the type of disk drive :-

-  Removable or Floppy drive
-  Non-removable or Hard drive
-  CD-ROM drive
-  Network drive
-  RAM drive

To change drives, simply position the mouse cursor over the desired disk drive, and press the left mouse button. A black rectangle is then drawn around the newly selected disk drive. Similarly, position the mouse cursor over the desired disk drive and press the right mouse button to display a pop up window which contains a list of the directories for the selected disk drive, as below :-

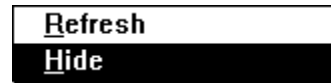


Simply double click the left mouse button on a directory to change into that directory.  is displayed for the currently selected directory. Continue this process until the desired directory is found and then press the left mouse button anywhere inside the main WinOne window to return to the WinOne prompt.

Also, pressing the left mouse button, anywhere on the drive bar background, is equivalent to pressing the enter key at the WinOne prompt. Similarly, pressing the right or middle buttons, anywhere on the drive bar background, is equivalent to typing DIR, at the WinOne prompt.

Hiding the Drive Bar

The option **Hide** will remove the drive bar from the bottom of the WinOne window.

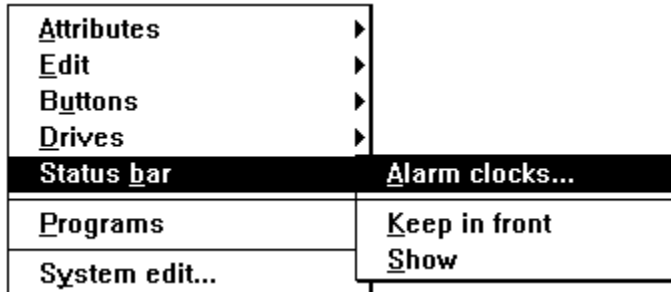


Refreshing the Drive Bar

Update the drive bar, by re-sampling the disk drives. Changes to network disk drives, either adding or removing them, after WinOne has started, will not be automatically reflected by the drive bar. The drive bar needs to be updated manually, using the **Refresh** option.

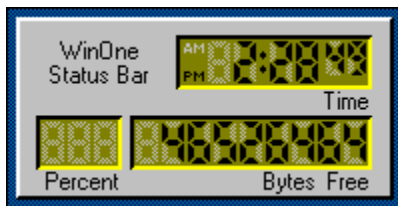
Status Bar and Alarm Clocks

The system menu contains the **Status Bar** option, which when selected will display a sub-menu containing several options, as below :-



Displaying the Status Bar

The option **Show** will display the Status Bar, showing the current time, the amount of free bytes in the global memory heap, and a percent indicator, which displays how much of a command is done. When no command is executing the percent display is blank.

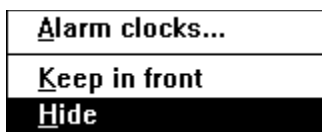


To move the Status Bar, simply position the mouse cursor inside the Status Bar window, hold down the left mouse button, then drag the white rectangle to its new position and release the left mouse button.

Not all the WinOne Commands use the Percent indicator, For example, CLS does not.

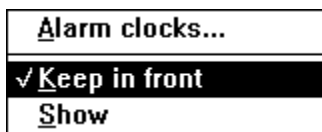
Hiding the Status Bar

The option **Hide** removes the Status Bar :-



Keeping the Status Bar in front

The option **Keep in front** will keep the Status Bar in front of all other Windows, even when the Status Bar is not active. This option is used to toggle this feature on and off. A tick mark will be placed next to the option text when it is on :-

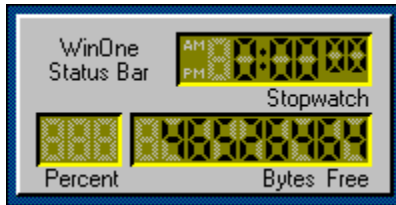


Stopwatch



To either display or remove the Stopwatch simply click the right mouse button, while the mouse

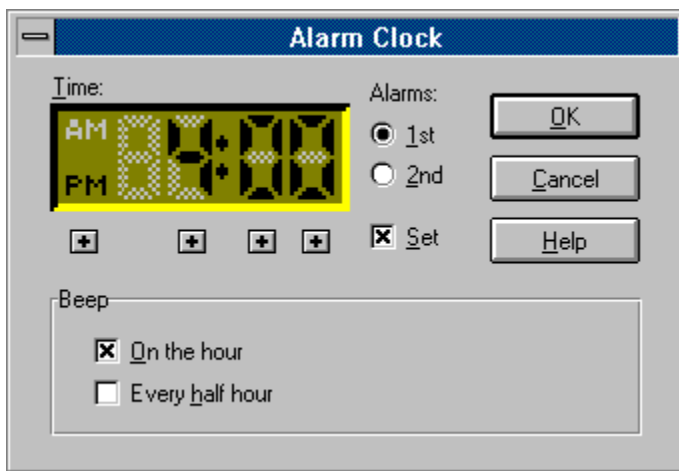
cursor is inside the Status Bar. Clicking the right mouse button twice, while the time is displayed, will automatically start the Stopwatch.



Pressing the SPACE key will either start or stop the Stopwatch and pressing the ESC key will reset the Stopwatch back to 0:00 00.

Alarm Clock

When the **Alarm Clocks...** option is selected the Alarm Clock window is displayed, as below :-



The Alarm Clock window displays the time for the selected alarm clock (ie. either the 1st or 2nd alarm clock), a set option which either enables or disables the respective alarm clock, four plus sign buttons to increment the respective digits (including toggling the AM or PM for the time) directly above the buttons on the LCD display and the beep hourly and half hourly options.



Setting an alarm clock involves the following steps :-

1. Select either the 1st or the 2nd alarm clock by positioning the mouse pointer over the desired alarm and pressing the left mouse button once. A dot is displayed next to the selected alarm clock.
2. Set the desired time by positioning the mouse pointer over the plus sign buttons, one at a time, and press the left mouse button to increment the digits directly above the plus sign buttons. Repeat this process until the desired time is displayed in the LCD display. The first plus sign button toggles the time from AM to PM and visa versa. The remaining three plus sign buttons increment the digits directly above the buttons.
3. Switch the alarm clock on by positioning the mouse pointer over the Set option and pressing the left mouse button. When the respective alarm clock is set then a small tick mark is displayed next the option text, otherwise no tick mark is displayed.
4. Press the OK button to accept the changes.



The beep hourly and half hourly options will emit a sound either every hour or every half hour. Simply position the mouse pointer over the respective option and press the left mouse button to either switch the option on, signalled by a small tick mark being displayed next to the option text

or off, signalled by no tick mark being displayed next to the option text.

See Event Sounds to associate a particular type of sound to each of these events.

Unix Look and Feel

The system menu contains the **Attributes** option, which when selected will display a sub-menu containing the **Unix mode** option, as below :-

A tttributes	S <u>i</u> ze...
<u>E</u> dit	F <u>o</u> nts...
B uttons	<u>C</u> olours...
<u>D</u> rives	<u>S</u> ounds...
S <u>t</u> atus <u>b</u> ar	<u>K</u> eys...
P rograms	<u>U</u> nix mode
S ystem edit...	<u>M</u> ail
	<u>R</u> un...

Selecting the **Unix mode** option will either switch this mode off, signified by no tick mark next to the option text, or switch this mode on, signified by a tick mark next to the option text, which will then give WinOne a look and feel similar to the Unix Operating System.

Unix Paths

Path names under Unix use front slash characters (ie '/') instead of back slash characters (ie. '\') to separate the directory names. For example, the Windows system directory is specified in the following way :-

D:/WINDOWS/SYSTEM

Unix Command Line Switches

Command line switches under Unix are signalled using a minus sign character (ie. '-') instead of a front slash character (ie. '/'). Also, a space character must proceed the minus sign character for the command line switch to be correctly interpreted. For example, to display a wide sorted directory listing of all the files in the Windows system directory, enter at the WinOne prompt :-

DIR D:/WINDOWS/SYSTEM/*. * -OW

There are some special cases where a space character followed by a minus sign character should NOT be interpreted as a command line switch, but instead be interpreted simply as a space character and a minus sign character. For example, to remove the Read Only attribute for all the files in the current directory :-

ATTRIB -R *. *

under Unix mode, will be incorrectly interpreted as :-

ATTRIB /R *. *

In this case, simply add an extra minus sign character. WinOne will interpret a space character followed by two minus sign characters as simply a space character and a minus sign character :-

ATTRIB --R *. *

Similarly, text mode type programs that already use minus sign characters to specify command line switches, will also need an extra minus sign character inserted to be correctly interpreted under Unix mode. For example, to test the contents of a ZIP file, under Unix mode, enter at the WinOne prompt :-

PKUNZIP --T SOMEFILE.ZIP

Command line strings

There is no difference between strings specified under Unix mode or DOS mode.

Unix Output

All output is displayed according to the above rules under Unix mode. Additionally, Unix path names will be displayed in lower case characters, where ever possible. The lower case mapping of path names can, however, be over-rided, in the following way :-

1. Enter at the WinOne prompt :-

SETINI UnixMode Case=type

where type is either upper (ie. for upper case characters), or lower (ie. for lower case characters).

2. Exit and re-run WinOne, so that the change can take effect.

Displaying DOS equivalent commands

DOS equivalent of Unix commands can be set to display after each command is entered, in the following way :-

1. Enter at the WinOne prompt :-

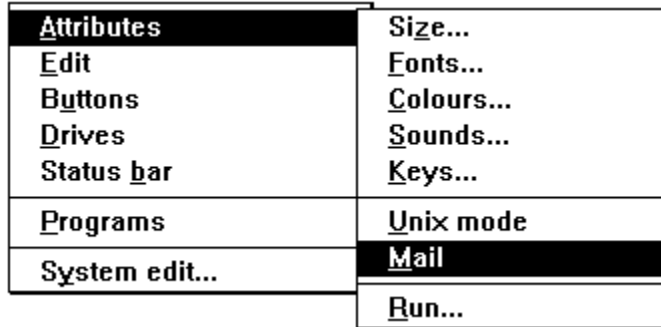
SETINI UnixMode Display=value

where value is either 1 (ie. display DOS equivalent commands), or 0 (ie. do not display DOS equivalent commands)

2. Exit and re-run WinOne, so that the change can take effect.

Mail Notifications

The system menu contains the **Attributes** option, which when selected will display a sub-menu containing the **Mail** option, as below :-



Selecting the **Mail** option will either switch mail notifications off, signified by no tick mark next to the option text, or switch mail notifications on, signified by a tick mark next to the option text. When this feature is enabled then WinOne will display the message "You have new mail" when new unread mail has arrived and is waiting in your incoming mail box.

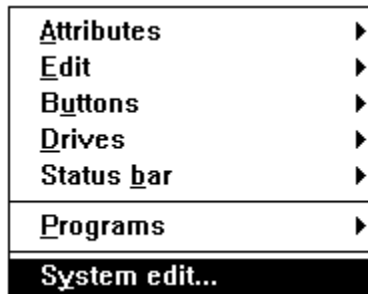
Also, when mail notifications is enabled, WinOne will display the familiar Mail Sign In window, after WinOne has started :-



Simply enter your name and password to sign into mail and press the OK button.

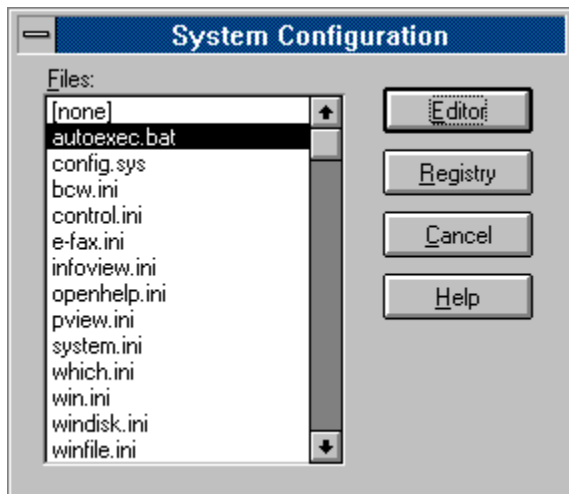
System Configuration Files

The system menu contains the **System Edit...** option, as below :-



When selected a window is displayed, showing a list of all the configuration files. These files include :-

1. AUTOEXEC.BAT
2. CONFIG.SYS
3. all .INI files



Use the mouse, to select the file to edit and press the Editor button.

Registry Button

Run the Windows System Registry Editor.

Note:

Do NOT alter any of the configuration files, when in doubt.

The default editor run by pressing the Editor button may be changed in the following way :-

1. Enter at the WinOne prompt :-

SETINI WinOne Editor=program.exe

where program.exe specifies the editor which will be used when the Editor button is pressed.

Command Syntax

Consider the following command syntax :-

NAME [drive:][path][filename] [...] [ON | OFF] variables /ABC

Words in blue upper case letters are keywords, and must be entered to specify the command.

Words and characters in magenta make up the rest of the command line. Usually referred to as the command line tail. Lower case words represent variables.

The variable drive: specifies the disk drive. For example A:, B: or C: etc.

The variable path specifies either a relative or absolute path name. An absolute path name has the path fully qualified (eg: \WINDOWS\SYSTEM32), where a relative path name does not and generally includes the use of dot characters to specify the parent directory (eg. ..\WINDOWS\SYSTEM32)

The variable filename specifies a file name. A file name can either conform to the 8.3 convention or be a long file name. When a file name contains one or more space characters then the complete drive, path and file name must be enclosed in quote character, regardless of whether the file name conforms to 8.3 or long file names. For example :-

"C:\MY PROGRAM.EXE"

Words or letters in square brackets (ie. ' [' and '] ') represent optional parameters and they do not need to be specified to execute the command. Generally these optional parameters change the behaviour of the command.

Three dots (ie. ' ... ') suggests that additional parameters are allowed be not necessary.

Words of letters separated by a vertical bar (ie. ' | ') represents one selection from a list.

A group of letters preceded by a front slash (ie. ' / ') specifies one or more switches . Switches represent command options that can be specified. Generally command switches will provide extra functionality for a command.

Command Argument Strings

A String is a sequence of characters enclosed in double-quote marks. For example, consider the string *"Hello, world"*.

To specify a double-quote mark inside the string, enter two double-quote marks. For example, the string *"Some ""quote marks"" inside a string"* will be interpreted as *Some "quote marks" inside a string*.

Command Argument Numbers

When specified in the documentation a number may either be :-

1. a whole number (ie. integer) or
2. a floating point number (ie. real)

Integers

Integer numbers are comprised of digits and may include an optional sign :-

[+ | -] dddd

For example, 255, -2147483648 and 2147483647 are all valid integer values.

Similarly, integers may also be specified as a hexadecimal (base 16) number by preceding the number with the characters "0x" :-

[+ | -] 0xdddd

or as an octal (base 8) number by preceding the number with the character "0" :-

[+ | -] 0dddd

Reals

Real numbers are comprised of digits and a decimal point. Real numbers may also include an optional sign and an exponent :-

[+ | -] dddd.dddd [e [+ | -] [ddd]]

For example, 4294967296.0, 3.14159265359 and 1.71122452428e+98 are all valid real numbers.

Standard Commands

Command ATTRIB

Function: Displays or changes file attributes.

Syntax: **ATTRIB** [+R | -R] [+A | -A] [+S | -S] [+H | -H] [[drive:][path]filename] [/S] [/O]

+	Sets an attribute.
-	Clears an attribute.
R	Read-only file attribute.
A	Archive file attribute.
S	System file attribute.
H	Hidden file attribute.
/S	Process sub-directories.
/O	Take ownership of files

Note: Sufficient privilege is required to take ownership of a file or file(s).

Command CHDIR or CD

Function: Displays the name of or changes the current directory.

Syntax:
CHDIR [drive:][path]
CD [drive:][path]

Note: To easily move between nested sub-directories see the command GO.
Type CD drive: to display the current directory for the specified drive.
Type CD without any parameters to display the current directory.
Also see command GO, CDD and Automatic Directory Changing.

Command CLS

Function: Clears the screen.

Syntax: **CLS** [/B]
/B Clear the internal screen buffer.

Command COPY

Function:

Copies one or more files to another location.

Syntax:

COPY [/A | /B] source [destination] [/V] [/N] [/Q]

source	Specifies the file(s) to be copied.
destination	Specifies the directory and/or filename for the new file(s).
/A	Ignored.
/B	Ignored.
/Q	Quiet Mode. Only Error Messages are displayed.
/N	Use short file name equivalents instead of long file names.
/V	Verifies that new files are written correctly.

Note:

All files are opened in binary mode.

The current directory is used when the parameter destination is not specified.

When copying files that have long file names from an NTFS partition to a FAT partition, the /N switch must be specified, to convert the long file names to the equivalent short file names, otherwise the copy operation will fail.

COPY does not support appending of files.

Also see the commands MOVE, SMOVE and SCOPY.

Command DATE

Function:

Displays or sets the date.

Syntax:

DATE [date]

date Specified according to the international format.

Note:

Type DATE without any parameters to display the current date setting, formatted according to the international setting in the Control Panel.

Parameter date is also specified according to the international format set in the Control Panel. For example, the USA date format is typically mm-dd-yyyy and the Australian date format is typically dd-mm-yyyy.

Also see command [TIME](#).

Command DEL or ERASE

Function:

Deletes one or more files.

Syntax:

DEL [drive:][path]filename [/P] [/S] [/F] [/Y] [/Q]
ERASE [drive:][path]filename [/P] [/S] [/F] [/Y] [/Q]

[drive:][path]filename Specifies the file(s) to delete.
/P Prompts for confirmation before deleting each file.
/S Process sub-directories.
/F Force deleting of read-only files.
/Q or /Y Quite mode or assume YES for all questions. Questions are not displayed.

Note:

When the /S switch and either *.* or a directory name is specified for parameter filename, then command DEL will also remove all the directories, including sub-directories, after deleting all the files, otherwise only the specified file or files are deleted and there is no attempt made to remove any directories or sub-directories.

Also see command [DELBT](#).

Command DIR

Function:

Displays a sorted list of files and sub-directories in a directory.

Syntax:

DIR [drive:][path][filename] [/P] [/W] [/A[:]attributes] [/O[:]order] [/T[:]timefield] [/V] [/N] [/F] [/B] [/J] [/S] [/L]

[drive:][path][filename]	Specifies drive, directory, and/or files to list.
/P	Ignored.
/W	Uses wide list format.
/A	Displays files with specified attributes, where attributes are D Directories R Read-only files H Hidden files A Archive files S System files - Prefix meaning NOT
/O	Display files sorted in the specified order, where order is :- G Group directories first N By name (alphabetic) E By file extensions D By file date and time (earliest first) S By file size (smallest first) - Prefix to reverse the order
/T	Display or sort on this timefield. where timefield is :- C Creation A Last access W Write
/V	Display Volume Name and Serial Number.
/N	Uses new long file name directory format.
/F	Uses 8.3 file name directory format, when possible.
/B	Uses bear format (only display file names).
/J	Right justified file name extensions for 8.3 file names.
/S	Process sub-directories.
/L	Display file names in lower case.

Note:

When the /F switch is specified and a directory contains files that conform to the 8.3 file naming convention, then regardless of the file system, the directory listing will be formatted with file names on the left hand side of the listing, otherwise the new long file name directory format is used. In the new long file name directory format the file names appear on the right hand side of the listing.

Switches that are preset in the DIRCMD environment variable are ignored.

Also see command [DEVICES](#).

Command EXIT

Function: Exit WinOne or Windows.

Syntax: EXIT [/L] [/S] [/B]

/L	Logout of Windows.
/S	Shutdown Windows.
/B	Reboot the computer.

Note: Type EXIT without any parameters to exit WinOne. The current state is automatically saved when exiting WinOne.

Command HELP

Function:

Display help information for a WinOne command.

Syntax:

HELP [command] [/L]

command Specifies the command or topic to display.

/L Use the WOIO.HLP file instead of the default WIN_ONE.HLP file.

Note:

WinOne is supplied with two main help files :-

WIN_ONE.HLP

The main WinOne help file.

WOIO.HLP

The WinOne Input and Output Library (WOIO) programmers reference. The WOIO Library package is used to write external commands.

Type HELP without any parameters to display the Contents for the WIN_ONE.HLP file.

Also see [On-line Help](#).

Command LABEL

Function: Creates, changes, or deletes the volume label of a disk.

Syntax: LABEL [drive:][label]

Note: Use the following syntax to delete a volume label :-

LABEL [drive:]

Command MEM

Function:

Display free memory that is available in the system heap.

Syntax:

MEM

Command MKDIR or MD

Function: Creates a directory.

Syntax:
MKDIR [drive:]path
MD [drive:]path

Note: Also see command [RMDIR](#).

Command PATH

Function:

Displays or sets the search path environment variable for executable files.

Syntax:

PATH [[drive:]path[;...]]

Note:

Type PATH ; to clear all search-path settings and direct WinOne to search only in the current directory.

Type PATH without any parameters to display the current path.

Command PROMPT

Function:

Changes the command prompt.

Syntax:

PROMPT [text]

text Specifies a new command prompt.

Prompt can be made up of normal characters and the following special codes :-

\$A	& (ampersand)
\$B	(pipe)
\$C	((left parenthesis)
\$D	Current date
\$E	Escape code (ASCII code 27), See ANSI Graphics
\$F) (right parenthesis)
\$G	> (greater-than sign)
\$H	Backspace (erases previous character)
\$I	{ (left brace)
\$J	} (right brace)
\$L	< (less-than sign)
\$N	Current drive
\$P	Current drive and path
\$Q	= (equal sign)
\$S	(space)
\$T	Current time
\$V	Windows version number
\$Z	Current command line number
\$\$	\$ (dollar sign)
\$_	Carriage return and linefeed

Note:

Type PROMPT without any parameters to reset the prompt to the default setting.

Command RENAME or REN

Function:

Renames a file or files.

Syntax:

```
RENAME [drive:][path]filename1 filename2  
REN [drive:][path]filename1 filename2
```

filename1 Specifies the source file(s).
filename2 Specifies the new file name(s).

Note:

A new drive or path can not be specified for the destination file (filename2).

Command REN can be used to rename directories. When renaming directories Wildcard characters can not be used.

Command RMDIR or RD

Function: Removes (deletes) a directory.

Syntax:
RMDIR [drive:]path [/S]
RD [drive:]path [/S]

/S Process sub-directories.

Note: Also see command [MKDIR](#).

Command SET

Function:

Display, set or remove environment variables.

Syntax:

SET [variable=[chars]]

variable Environment variable name.

chars A series of characters to assign to the variable.

Note:

Type SET with no parameters to display a list of all environment variables.

When parameter chars is not specified then the environment variable will be removed from the environment space,

Also see command [XSET](#), [LET](#) and [Batch Programs](#).

Command TIME

Function: Displays or sets the system time.

Syntax: **TIME** [time]

time hh:mm:ss

Note: Type TIME with no parameters to display the current time setting.

The parameter time is specified in 24 hour format.

Also see command DATE.

Command TREE

Function: Graphically displays the directory structure.

Syntax: **TREE** [drive:][path]

Command TYPE

Function:

Displays the contents of a text file.

Syntax:

TYPE [drive:][path]filename [/L] [/tabstop]

/tabstop Number between 1 to 8 inclusive. Specifies the number of space characters to use to expand tabstop characters.

/L Display line numbers.

Note:

The default number of space characters used to expand tabstop characters is 8.

A file can contains ANSI escape sequences, see [ANSI Graphics](#).

Also see command [DUMP](#).

Command VER

Function:

Display the version numbers for Windows and WinOne.

Syntax:

VER

Command VOL

Function: Display the disk volume label.

Syntax: **VOL**

Extra Commands

Command ACS

Function:

Display the full ASCII Character Set or return the character for the specified value. The character returned is stored in an environment variable called ACS.

Syntax:

ACS [value]

value Return the character for the specified value.

Note:

When no parameters are specified then the full ASCII character set is displayed.

Command ARCH

Function:

Displays files inside most popular archive formats, including ZIP, LZH, ARJ, ARC formats.

Syntax:

ARCH [[drive:][path]filename] [/V]

/V Verbose mode display.

Note:

Type ARCH with no parameters, to display files inside any archive format, in the current directory.

Verbose mode will display the filename, original file size, compressed file size, percentage, date and time for each file inside the archive.

Also see command [UNZIP](#).

Command CDD

Function: Changes the current directory and disk.

Syntax: **CDD** [drive:][path]

Note: Also see command CD, GO and Automatic Directory Changing.

Command DELBUT

Function:

Deletes all files except the files specified.

Syntax:

DELBUT [drive:][path]filename1 [filename2] [...] [/P]

[drive][path] Specifies the drive and path where the file(s) can be found.

filename1, ... Specifies the file(s) NOT to delete.

/P Prompts for confirmation before deleting each file.

Note:

Also see command DEL.

Example on using DELBUT.

DELBUT Example

To delete all files except the files that have a file extension of .EXE or .COM, in the current directory, enter at the prompt :-

```
DELBUT *.EXE *.COM
```

Command DESCRIBE

Function:

Add, modify or delete a file or directory description of up to 128 characters.

Syntax

DESCRIBE [drive:][path]filename ["description"]

[drive:][path]filename Specifies the file name or the directory name to be described.

description Specifies a sequence of up to 128 characters.

Note:

When the parameter description is not specified then the file or directory description will be deleted. The file or directory itself will not be deleted.

Wildcard characters can not be used with parameter filename.

File and directory descriptions are stored in a hidden file called DESCRIPT.ION, and can be displayed by using the DIR command. Descriptions are automatically maintained when using the commands COPY, DEL, DELBUT, MOVE, RENAME, RMDIR, SCOPY and SMOVE.

Command DISK

Function:

Display bytes used for all sub-directories (including nested sub-directories), of the specified directory.

Syntax:

DISK [[drive:]path]

Note:

Type DISK without any parameters to display the number of bytes used for all sub-directories and nested sub-directories, contained in the current directory.

Command DOS

Function:

Shell to DOS or change the behaviour of all text mode windows.

Syntax:

DOS [CLOSE | INACTIVE]

CLOSE - Close text mode windows on exit.

INACTIVE - Do not close text mode windows on exit, instead leave all text mode windows open. The windows title will be changed to include the word "Inactive", signalling that the program has completed executing and may be close manually at any time.

Note:

Type DOS without any parameters to run the default shell, specified by the COMSPEC environment variable.

Also see [Program Run Options](#).

Command DUMP

Function:

Displays the contents of any file.

Syntax:

DUMP [drive:][path]filename [offset] [/D] [/H] [/A] [/Z]

filename	Specifies the file to display.
offset	Specifies the starting offset in the file.
/D	Decimal output.
/H	Hexadecimal output.
/A	Display characters only.
/Z	Display offsets starting from zero.

Colours:

The output is colour coded to make it easier to recognise the different character types :-

Magenta	Letter characters
Cyan	Punctuation characters
Yellow	Dot character
Green	Number characters
Red	Null (zero) character
Blue	Carriage return or Linefeed characters
White	Unprintable characters

Note:

The default output is in hexadecimal values.

Parameter offset can be specified as either a decimal or a hexadecimal value. A hexadecimal value must be preceded with the characters '0x'. For example, 255 in hexadecimal is 0xFF.

Also see command TYPE.

Command FIND

Function:

Search for a text string in the specified files.

Syntax:

FIND [drive:][path]filename "textstring" [/S] [/M]

[drive:][path]filename Specifies the text file(s) to search.

textstring Specifies the text string to find.

/M Match case.

/S Process sub-directories.

Note:

Command FIND uses the fast Boyer-Moore Algorithm and will only search text files. Non-text file are simply skipped.

Command FIND is not compatible with the DOS command FIND.

Unlike the DOS command FIND, Wildcards are allowed for parameter filename.

Also see [macro example 3](#).

Command FINDREG

Function:

Search the System Registry for a string.

Syntax:

FINDREG [hkey] "textstring"

hkey Specifies the name of the registry to search.
textstring Specifies the text string to find.

Note:

When parameter hkey is not specified then all the registries are searched, otherwise parameter hkey must be one of the following System Registry names :-

HKEY_USERS
HKEY_CURRENT_USER
HKEY_LOCAL_MACHINE
HKEY_CLASSES_ROOT

Command FINDREG will only search the following types and all other types are ignored :-

REG_SZ	A null terminated string.
REG_DWORD	A 32 bit number.
REG_DWORD_LITTLE_ENDIAN	Same as REG_DWORD.
REG_DWORD_BIG_ENDIAN	A 32 bit number which is automatically converted to REG_DWORD format.

The search is not case sensitive.

Command LET

Function:

Evaluate an expression and set the specified environment variable with the result.

Syntax:

LET variable=expr

variable Environment variable name where the result is stored.
expr Expression to evaluate.

Note:

Parameter expr follows the same rules as parameter expr for command CALC. Also, when parameter expr includes characters that have a special meaning (eg. <, >, & etc) then enclose the complete expr in brackets, as follows :-

LET variable=(expr)

Also see command SET, XSET, CALC and Batch Programs.

Command MORE

Function: Page the output of a command.

Syntax: **MORE**

Note: Command MORE is used as a pipe. When a programs output has been piped through command MORE then -- More -- will be displayed at the bottom of the screen, after each screen full of information has been displayed. Simply press any key to continue on to the next screen full of information.

There is no need to pipe the output of any WinOne internal or external commands to MORE, since WinOne automatically pages their output. However, this is not the case for text mode type programs which create their own windows. In this case, using the MORE command will pipe the output of such programs back to the main WinOne window.

Also see [Redirecting Command Input and Output](#) and command [PAGE](#).

Command PAGE

Function: Sets paging on or off.

Syntax: **PAGE [ON | OFF]**

Note: When paging is set to on, then -- MORE -- will be displayed at the bottom of the screen, after each screen full of information has been displayed. Simply press any key to continue on to the next screen full of information.

Paging always defaults to ON when WinOne is executed.

Also see command MORE.

Command GO

Function:

Changes the current directory to another sub-directory.

Syntax:

GO [drive:]path

path The target sub-directory to change to.

Note:

This command searches the directory structure to find the first matching sub-directory to change to.

Type GO \path to start searching from the root directory.

Also see command [CD](#), [CDD](#) and [Automatic Directory Changing](#).

[Example](#) on using GO.

GO Example

Assume you are in the root directory of D: with the following prompt :-

```
D:\>
```

To move to the directory D:\WINDOWS\SYSTEM, enter at the WinOne prompt :-

```
GO SYSTEM
```

The NEW prompt will be :-

```
D:\WINDOWS\SYSTEM>
```

Command HISTORY or HIS

Function:

Displays the history buffer, which consists of commands entered at the WinOne prompt.

Syntax:

HISTORY [LIST] [/B]
HIS [LIST] [/B]

/B Clear the history list.
LIST Display the pop up history list.

Note:

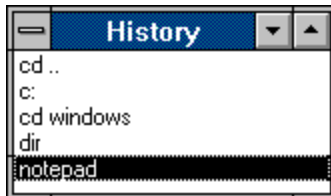
Also see the [Pop Up History List](#) and [Command Line Edit Keys](#).

Pop Up History List

WinOne contains a pop up history list which is displayed using the HISTORY command and entering at the WinOne prompt :-

HISTORY LIST

The pop up history list contains the complete command history, as below :-

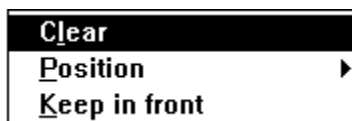


To select a command to run from the pop up history list, simply position the mouse pointer over the desired command line and double click the left mouse button. The selected command line will then be displayed at the WinOne prompt and executed.

A command from the pop up history list can be located a number of different ways using key strokes instead of using the mouse. The following keys strokes perform various functions :-

char	Locate the previous command line that begins with the character char. Repeatedly pressing the key char will move backward through the history list.
ctrl char	Locate the next command line that begins with the character char. Repeatedly pressing the control key and the char key together will move forward through the history list.
up arrow	Move back one command line.
down arrow	Move forward one command line.
page up	Move back one page full of command lines.
page down	Move forward one page full of command lines.
home	Locate the oldest command line.
end	Locate the most recent command line.
shift escape	Switch to and from the WinOne prompt.
enter	Select the command line that is highlighted. The command line is displayed at the WinOne prompt and executed.

The pop up history list system menu contains several options, as below :-



Clearing the History List

The option **Clear** will clear the contents of the history list. This option has the same effect as using the /B switch for command HISTORY.

Positioning the History List

The option **Position** will display a sub-menu containing several options, as below :-



Each option in the sub-menu will position the pop up history list along the selected edge of the desktop.

Keeping the History List in front

The option **Keep in front** will keep the pop up history list in front of all other Windows, even when the history list is not active. This option is used to toggle this feature on and off. A tick mark will be placed next to the option text when it is on.

Command MACRO

Function: Provides command line macro's.

Syntax: **MACRO** [macroname=text]

or

MACRO [/D macroname]

or

MACRO [ON | OFF]

macroname Specifies the name of the new command.

text Specifies the commands to be executed.

/D Delete a macro.

ON Displays expanded macro's.

OFF Does not display expanded macro's.

There are some special character combinations that have a special meaning and that can be included in parameter text :-

\$\$	- Dollar sign.
\$1 to \$9	- Replaceable parameters similar to %1 to %9 parameters used in batch programs.
\$*	- Replace with the whole command tail.
\$B	- Bar character. Pipe commands.
\$G	- Greater than sign character. Redirect output.
\$L	- Less than sign character. Redirect input.
\$T	- Command separator.

Note: Type MACRO without any parameters to display a list of all the macro's defined.

Macro's can have the same name as an original command, to execute the original command, type a space character before the command is entered.

Only the first 12 characters in parameter macroname are significant when executing a macro at the WinOne prompt. However, all the characters in parameter macroname are significant when deleting a macro.

Macro's are saved as they are created.

Examples on using MACRO.

MACRO Examples

Example 1:

To create a command LS, which will display a sorted directory listing in lower case letters and in wide format, enter at the WinOne prompt :-

```
MACRO LS=DIR /OWL $*
```

To display all .EXE files, enter at the WinOne prompt :-

```
LS *.EXE
```

Example 2:

To alter the DIR command to display a sorted directory listing, enter at the WinOne prompt :-

```
MACRO DIR=DIR /O $*
```

Example 3:

To create a command to locate a file by its file description only, enter at the WinOne prompt :-

```
MACRO WHEREIS=FIND \*.ION "$*" /S
```

To locate all the External Commands, enter at the WinOne Prompt :-

```
WHEREIS EXTERNAL COMMAND
```

Example 4:

To create a command to execute an original internal CMD.EXE command, enter at the WinOne prompt :-

```
MACRO ORIG=CMD.EXE /C $*
```

To redirect the CMD.EXE DIR command into a file called TEMP.TXT, enter at the WinOne prompt :-

```
ORIG DIR > TEMP.TXT
```

Example 5:

To delete the macro ORIG, enter at the WinOne prompt :-

```
MACRO /D ORIG
```

Command MOVE

Function:

Moves one or more files to another location.

Syntax:

MOVE [/A | /B] source [destination] [/V] [/N] [/Q]

source	Specifies the file(s) to be copied.
destination	Specifies the directory and/or filename for the new file(s).
/A	Ignored.
/B	Ignored.
/Q	Quiet Mode. Only Error Messages are displayed.
/N	Use short file name equivalents instead of long file names.
/V	Verifies that new files are written correctly.

Note:

All files are opened in binary mode.

The current directory is used when the parameter destination is not specified.

When copying files that have long file names from an NTFS partition to a FAT partition, the /N switch must be specified, to convert the long file names to the equivalent short file names, otherwise the copy operation will fail.

Also see the commands COPY, SCOPY and SMOVE.

Command TIPS

Function:

Switch the displayed start up tip on or off.

Syntax:

TIPS ON | OFF

ON Enable the displaying of a tip on start up.

OFF Disable the displaying of a tip on start up.

Note:

Tip's are enable, when WinOne is first installed.

Command TRACE

Function:

Provide debugging support for [batch programs](#).

Syntax:

TRACE [[drive:][path]filename [batch-parameters]]

[drive:][path]filename Batch program name and location.

batch-parameters Any command line information that is used by the batch program.

Note:

When the TRACE command is used to start a specified batch program, then the debugging process begins from the beginning of the batch program.

To specify a point (ie. a break point) within a batch program where the debugging process is to begin, use the TRACE command without any parameters. For example, consider the following batch program :-

```
@ECHO off  
LET total=0  
  
TRACE  
FOR %%i is 1 to 10 do {  
    LET total=%total% + %%i  
}  
ECHO sum is %total%
```

When using the TRACE command, as in the above batch program, there is no need to start the batch program using the TRACE command, instead run the batch program by entering its name at the WinOne prompt. When WinOne encounters the TRACE command included in the above batch program, then the debugging process will begin from that point.

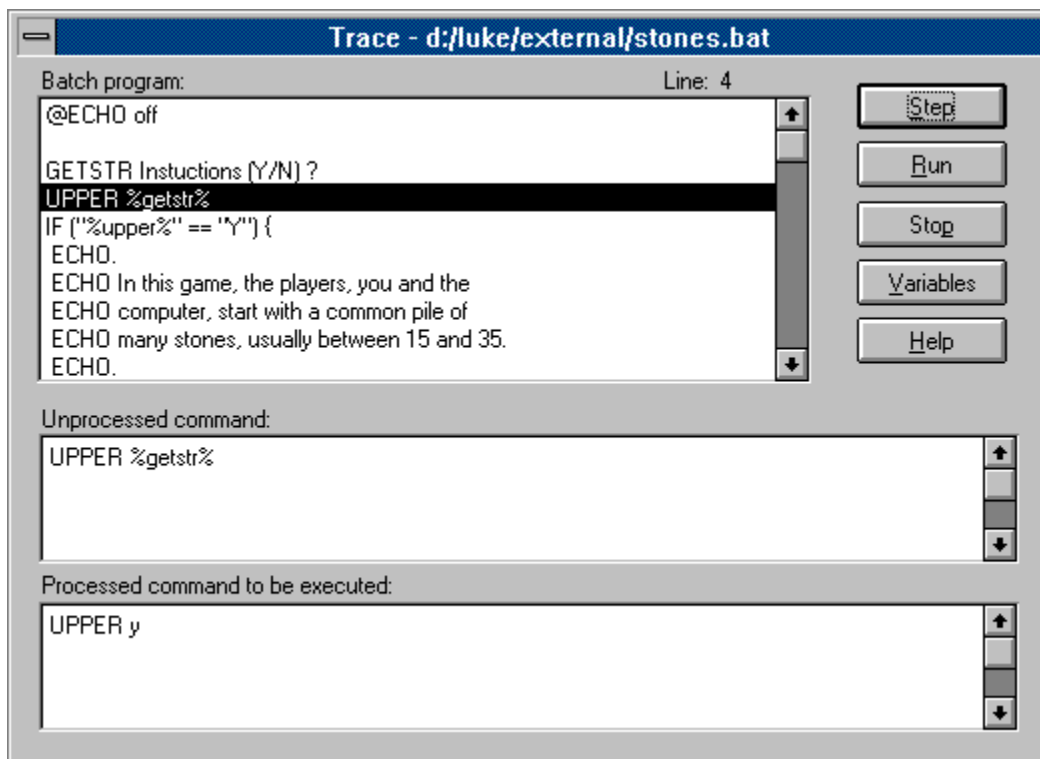
The use of the TRACE command without any parameters to specify break points is valid only when used within a batch program.

Also see [Batch Program Debugging](#).

Batch Program Debugging

Debugging is a process where by the execution of a program is tightly controlled in order to help locate and fix problems (commonly referred to as bugs) that may arise while writing a program. WinOne provides the TRACE command to help in the debugging of a batch program. Traditionally, in command shells such as CMD.EXE, debugging support for batch programs has been limited to using the ECHO batch command (ie. to display a batch program while running or to display the contents of an environment variable), however, WinOne provides a far greater level of control, including single step, running, stopping and environment variable manipulation while a batch program is being debugged

When using the TRACE command to debug a batch program the following window is displayed, which includes the Batch Program list box, which contains a listing of the complete batch program, the Unprocessed Command box, which contains the full unprocessed command line that is being interpreted and the Processed Command to be Executed box, which contains the actual command (after relevant environment variables have been instantiated) that is next in line for execution :-



Single Step

Single step allows one command in a batch program to be executed at a time. Press the Step button to execute the command currently displayed in the Processed Command to be Execute box and to process (but not execute) the next command.

Running the Batch Program

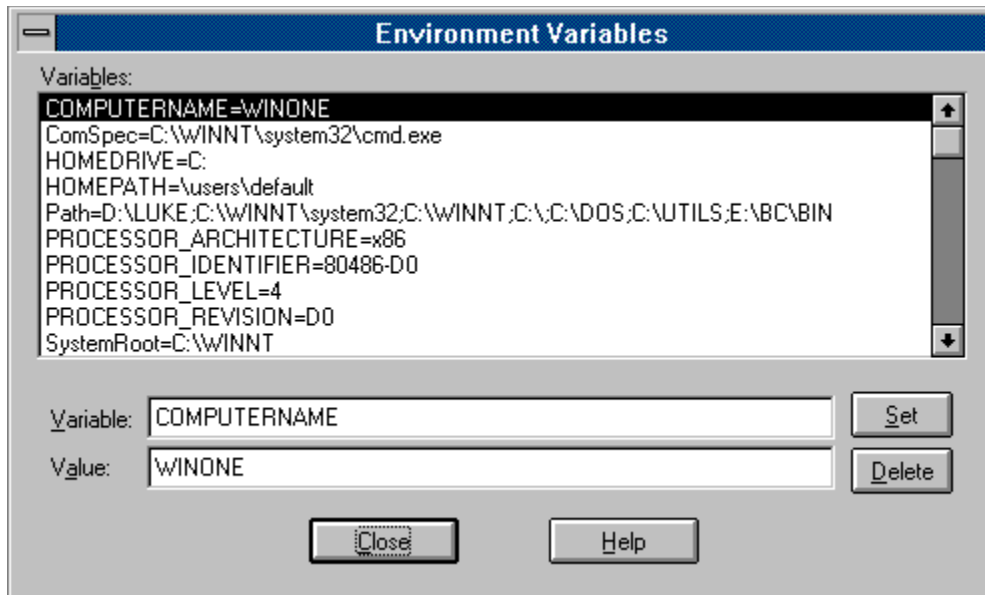
Press the Run button to close the Trace window and continue running the remainder of the batch program.

Stopping the Batch Program

Press the Stop button to stop the batch program, close the Trace window and return to the WinOne prompt..

Variables

Press the Variables button to view and/or modify the current environment variables for the batch program. The Environment Variables window is displayed, as follows :-



To move an environment variable from the Variables list into the Variable and Value fields respectively (where the environment variable may be set or deleted), simply double click on the desired environment variable in the Variables list.

Command REPLACE

Function:

Search and replace a text string in the specified files.

Syntax:

REPLACE [drive:][path]filename "searchfor" "replacewith" [/M] [/Y] [/S]

[drive:][path]filename Specifies the text file(s) to search.

searchfor Specifies the text string to replace.

replacewith Specifies the replacement text string.

/M Match case.

/Y Assume yes for all questions.

/S Process sub-directories.

Note:

Command REPLACE uses the fast Boyer-Moore Algorithm and will only search text files. Non-text file are simply skipped.

The parameter replacewith is case sensitive. The parameter searchfor is not, unless the /M switch is specified.

Command SCOPY

Function:

Safely copies one or more files to another location. Will display a warning when a file that already exists is about to be over-written.

Syntax:

SCOPY [/A | /B] source [destination] [/V] [/N]

source	Specifies the file(s) to be copied.
destination	Specifies the directory and/or filename for the new file(s).
/A	Ignored.
/B	Ignored.
/N	Use short file name equivalents instead of long file names.
/V	Verifies that new files are written correctly.

Note:

All files are opened in binary mode.

The current directory is used when the parameter destination is not specified.

SCOPY does not support appending of files.

When the destination file exists then the user is prompted with the message **Overwrite (Y/N/A/S) ?**. Where entering :-

Y	Yes. Overwrite the file.
N	No. Do not overwrite the file. A new filename is then requested.
A	Always overwrite the files. No further warnings are displayed.
S	Skip this file only.

When copying files that have long file names from an NTFS partition to a FAT partition, the /N switch must be specified, to convert the long file names to the equivalent short file names, otherwise the copy operation will fail.

Also see the commands COPY, MOVE and SMOVE.

Command SMOVE

Function:

Safely moves one or more files to another location. Will display a warning when a file that already exists is about to be over-written.

Syntax:

SMOVE [/A | /B] source [destination] [/V] [/N]

source	Specifies the file(s) to be copied.
destination	Specifies the directory and/or filename for the new file(s).
/A	Ignored.
/B	Ignored.
/N	Use short file name equivalents instead of long file names.
/V	Verifies that new files are written correctly.

Note:

All files are opened in binary mode.

The current directory is used when the parameter destination is not specified.

When the destination file exists then the user is prompted with the message **Overwrite (Y/N/A/S) ?**. Where entering :-

Y	Yes. Overwrite the file.
N	No. Do not overwrite the file. A new filename is then requested.
A	Always overwrite the files. No further warnings are displayed.
S	Skip this file only.

When copying files that have long file names from an NTFS partition to a FAT partition, the /N switch must be specified, to convert the long file names to the equivalent short file names, otherwise the copy operation will fail.

Also see the commands MOVE, COPY and SCOPY.

Command START

Function:

Start a program and return the WinOne prompt immediately.

Syntax:

START ["title"] [/V] program [parameters]

"title"	Ignored.
/V	Run the program in a separate VDM.
program	Specifies the program to execute.
parameters	Specifies the parameters, for the program that is to be executed.

Note:

Users are encouraged to use the following method, to place a task into the background, instead of using the START command. When WinOne executes any program, that creates its own window, then switching back to the main WinOne window and pressing the control key and the z key together (ie. CTRL Z)., will return the WinOne prompt, immediately.

Only 16 bit non-text mode programs can be run in a separate VDM. A separate VDM allows these programs to preemptively multitask. This option is only effective on Windows NT 3.5 or higher and is ignored in all other cases.

Command WHERE

Function:

Locates one or more files.

Syntax:

WHERE [drive:][path]filename

drive	Specifies the drive to search.
path	Specifies the directory to start searching in.
filename	Specifies the file or files to be found.

Note:

Type WHERE filename, without a drive or path to search the current drive and all non-removable drives on the hard disk.

Examples on using WHERE.

Also see command WHICH.

WHERE Examples

Example 1:

To search all drives on the hard disk for all files that have the extension .DOC, enter at the WinOne prompt :-

```
WHERE *.DOC
```

Example 2:

To search for files that have the extension .TXT on A: only, starting from the root directory, enter at the WinOne prompt :-

```
WHERE A:\*.TXT
```

Command WHICH

Function:

Displays the full path of a program.

Syntax:

WHICH filename[.ext]

filename Specifies the file name of the program to be located.
.ext Optional program file name extension to search for.

Note:

Command WHICH will search the PATH environment variable, in order to locate the specified file name.

When no file name extension is specified, then the extensions .EXC, .PIF, .COM, .EXE, .CMD and .BAT are tried.

The full path displayed, when the file name is found, reflects the program that WinOne would execute, unless the program is located in the current directory. Command WHICH does not search the current directory.

Also see command WHERE.

Command XSET

Function:

Display, set or remove persistent environment variables.

Syntax:

XSET [variable=[chars]]

variable Persistent environment variable name.
chars A series of characters to assign to the variable.

Note:

The persistent environment variables are stored in the system registry and Windows NT will automatically include these environment variables in the environment space for all programs. Also, setting or deleting a persistent environment variable will cause all interested programs to update their environment space immediately (eg. Program Manager, WinOne etc).

Type XSET with no parameters to display a list of all the persistent environment variables.

When parameter chars is not specified then the persistent environment variable will be removed from the environment space,

The PATH environment variable is a special case and is made up of 3 parts, as below :-

system_path ; persistent_path ; autoexec_path

Command XSET will only manipulate the persistent_path part of the complete PATH environment variable. For example, to add C:\TEST enter at the WinOne prompt :-

XSET path=c:\test

and the resulting PATH environment variable will be :-

system_path ; c:\test ; autoexec_path

Command XSET is not available under Windows 95 or Win32s.

Example on using XSET.

Also see command SET, LET and Batch Programs.

XSET Example

Consider the following batch program :-

```
@ECHO off

IF ("%update%"=="") {
    ECHO No last access.
} ELSE {
    DATE2SER %date_today%
    SET today=%date2ser%

    DATE2SER %update%
    LET days=%today%-%date2ser%

    ECHO Last access on %update%, %days% days ago.
}

ECHO.

GETKEY Update last access (Y/N) ?
UPPER %getkey%
IF ("%upper%"=="Y") XSET update=%date_today%
```

The above batch program demonstrates the use of command XSET and uses a persistent environment variable to determine the last date an access occurred. An access can refer to anything, for example, dialing into a BBS etc.

Window Commands

Command ABOUT

Function: Displays the About window.

Syntax: **ABOUT**

Note: The registered version of WinOne displays the following About window :-



The date October 23, 1994 displayed in the About window reflects the first day that development commenced on the 32 bit version of WinOne. The About window also displays various other information, such as the registered owner of WinOne, the current WinOne version number and various memory statistics.

Also see the [Shareware Information](#) section for more information on how to register WinOne and for the benefits of registering WinOne.

Command CAPTURE

Function:

Switch capturing of input and output for console programs on or off. Also allows the No Capture list to be manipulated.

Syntax:

CAPTURE [[/D] path]

or

CAPTURE [ON | OFF]

path	Specifies the full path of the console program to include in the No Capture list. Programs included in this list will run in their own separate window.
ON OFF	Enable or disable capturing of input and output for console programs.
/D	Remove the specified console program from the No Capture list.

Note:

WinOne uses anonymous pipes to receive and send characters to and from a console program. This process is referred to as "capturing a programs input and output". Console programs are a sub-set of text mode type programs. Most standard DOS commands have been converted to console programs in Windows NT.

The No Capture list contains console programs that should not be run under capture. This includes console programs that do not use streamable input and output. The console programs contained in the No Capture list will be given their own window when executed from the WinOne prompt.

When an attempt is made to capture the input and output of a console program that does not use streamable input and output then the program will be blocked until ^C is pressed to end the program. The program should then be added to the No Capture list, so that, WinOne will know NOT to capture the programs input or output next time it is run. There is no way of knowing in advance whether or not a program will perform correctly under capture, hence the need for a No Capture list. This list is similar in concept to PIF files.

All other types of text mode programs (other then console programs) will be run in their own window (eg. DOS programs) when run from the WinOne prompt. There is no need to include these types of programs in the No Capture list.

The feature is not available when WinOne is running on Win32s, since Win32s is not capable of running console programs.

When no parameters are specified for command CAPTURE then the current No Capture list is displayed.

Also see [Program Run Options](#).

Command CLIP

Function:

Place a line of text into the clipboard.

Syntax:

CLIP [text]

text Sequence of character.

Note:

When parameter text is not specified then the clipboard is cleared.

When the clipboard already contains some text then command CLIP will append the specified line of text to the clipboard.

To place a blank line into the clipboard type CLIP followed by a dot character :-

CLIP.

Also see command [CLIPDISP](#).

Command CLIPDISP

Function: Display any text in the clipboard.

Syntax: **CLIPDISP**

Note: When the clipboard does not contain any text then nothing is displayed.
Also see command CLIP.

Command DDEEXEC

Function:

Send an execute command to an application.

Syntax:

DDEEXEC app topic "item"

app	Specifies the application name to communicate with.
topic	Specifies the subject of the conversation.
item	A <u>string</u> containing the command to execute.

Note:

Dynamic data exchange (DDE) is a protocol for interprocess communication.

Command DDEEXEC will wait, up to 60 seconds, for the specified command to be executed before displaying an error message.

Examples on using DDEEXEC.

Also see commands DDEPOKE and DDEREQ.

Command DDEPOKE

Function:

Send data to an application.

Syntax:

DDEPOKE app topic "item1" "item2"

app	Specifies the application name to communicate with.
topic	Specifies the subject of the conversation.
item1	A <u>string</u> that specifies the subject of the data.
item2	A <u>string</u> that specifies the data to send.

Note:

Dynamic data exchange (DDE) is a protocol for interprocess communication.

Command DDEPOKE will wait, up to 60 seconds, for the specified data to be accepted before displaying an error message.

Example on using DDEPOKE.

Also see commands DDEREQ and DDEEXEC.

Command DDEREQ

Function:

Request data from an application.

Syntax:

DDEREQ app topic "item"

app	Specifies the application name to communicate with.
topic	Specifies the subject of the conversation.
item	A <u>string</u> containing the request.

Note:

Dynamic data exchange (DDE) is a protocol for interprocess communication.

Command DDEREQ will wait, up to 60 seconds, for the requested data before displaying an error message.

The data returned by the specified application is placed in the following environment variables :-

- | | |
|----------------------------|---|
| 1. DDEREQ | Contains the index value of the last environment variable that contains the data. |
| 2. DDEREQ.0, DDEREQ.1, ... | Contains the requested data. |

Example on using DDEREQ.

Also see commands DDEEXEC and DDEPOKE.

DDE Examples

The following examples illustrates the use of DDE with Word for Windows and assumes that Word is running and the currently open document contains a bookmark called "mark" some where within the document.

Example 1:

To insert the text "hello world" at the bookmark, enter the following at the WinOne prompt :-

DDEPOKE WinWord Document1 mark "hello world"

Example 2:

To goto the text in the bookmark and highlight it, enter the following at the WinOne prompt :-

DDEEXEC WinWord Document1 "[EditGoTo ""mark""]"

Example 3:

To format the text in the bookmark to Arial, with a point size of 20 and using the bold attribute, enter at the WinOne prompt :-

DDEEXEC WinWord Document1 "[FormatCharacter ""Arial"", ""20"", 1]"

This example assumes that the bookmark has been highlighted as in example 2.

Example 4:

To retrieve the text in the bookmark, enter at the WinOne prompt :-

DDEREQ WinWord Document1 mark

The environment variable DDEREQ will contain the value 0 and DDEREQ.0 will contain the text retrieved from the bookmark.

Command **DEVICES**

Function: List all the devices available on a system.

Syntax: **DEVICES**

Note: Command DEVICES is only supported on Windows NT.
Also see command DIR.

Command DISPBOX

Function:

Display the contents of a plain text file in a pop up window. The button pressed to close the window is stored in an environment variable called DISPBOX.

Syntax:

DISPBOX "title" filename

title A string which specifies the window caption.

filename Specifies the plain text file to display in the pop up window.

Note:

A fixed width font is used to display the plain text file.

Also see command MSGBOX and EDITBOX.

Command DRAG

Function:

Drag one or more files to another program.

Syntax:


DRAG [drive:][path]filename

filename Specifies the file(s) to drag.

Note:



After entering the DRAG command, click the left mouse button in the main WinOne window to pick up the specified file(s), then while holding down the left mouse button, drag the mouse cursor into the window or icon in which to drop the file(s) and release the left mouse button.

Not all programs accept files that are dragged over their window or icon, the  cursor is displayed when a window can not accept the file drop and similarly



or



cursor is displayed when a window can accept the file drop.

Also see [File Drag and Drop](#).

Command EDITBOX

Function:

Display a pop up window containing an edit field. The button pressed to close the window is stored in an environment variable called EDITBOX and the characters entered in the edit field is stored in an environment variable called EDITBOXA.

Syntax:

EDITBOX "title" "prompt" ["default"]

title	A <u>string</u> which specifies the window caption.
prompt	A <u>string</u> which is displayed above the edit field.
default	A <u>string</u> which specifies the default sequence of characters for the edit field.

Note:

Also see command MSGBOX and DISPBOX.

Command EJECT

Function:

Eject a CD-ROM.

Syntax:

EJECT

Note:

Command EJECT uses the Media Control Interface (MCI) to eject a CD-ROM from the first CD-ROM drive installed.

When the MCI drivers are not installed or not supported by an operating system then the CD-ROM will not be ejected.

Command EXT or EXTENSION

Function:

Set filename associations for file extensions.

Syntax:

EXTENSION [extension=association]
EXT [extension=association]

or

EXTENSION [/D extension]
EXT [/D extension]

extension Specifies the filename extension.
association Specifies the program name and parameters.
/D Delete an extension.

Note:

Do NOT include a dot in the parameter extension.

Type EXT without any parameters to display a list of all existing associations.

When an association is established for a file extension, then file names (which MUST including the extension) can be entered at the prompt without the need to specify the program name.

Multiple file extension associations are created and deleted in the same way as single file extension associations. When a single association already exists then WinOne will prompt the user with the message **Multiple XXX extension (Y/N)?**, in which case typing 'Y' will append the association, or typing 'N' will over write the existing association. Also when deleting Multiple associations, WinOne will confirm each association before deleting it.

File extension associations are saved as they are created.

Also see [File Extension Association](#).

[Examples](#) on using EXTENSION.

EXTENSION Examples

Example 1:

To associate all file names with an extension of TXT to the program NOTEPAD.EXE, enter at the WinOne prompt :-

```
EXT TXT=NOTEPAD.EXE ^^ .TXT
```

Two hat characters (ie. ' ^^ ') are needed since a single hat character has a special meaning. The resulting hat character (ie. ' ^ '), will be replaced with the file name excluding the extension, hence the .TXT extension must be specified.

To execute the program NOTEPAD and pass the file REPORT.TXT to the program, enter at the WinOne prompt :-

```
REPORT.TXT
```

Example 2:

Assume that the drive and directory for the program WINGIF.EXE is located in D:\PICS. Then to create an association to view GIF files, enter at the WinOne prompt :-

```
EXT GIF=D:\PICS\WINGIF.EXE ^^ .GIF
```

To view GIF files, located in D:\CARS, enter at the WinOne prompt :-

```
D:\CARS\*.GIF
```

Example 3:

To delete the file extension association for .TXT files, enter at the WinOne prompt :-

```
EXT /D TXT
```

Special notes on associations:

Wildcards can be entered at the WinOne prompt, such as in Example 2 above, but the filename extension must not contain Wildcards, since the filename extension is used to determine which association is to be used.

When an association is determined, then WinOne will try to locate the program. WinOne will search the current directory first, and if it is not found in the current directory then the PATH environment variable is searched.

Command GROUP

Function:

Insert program files into Program Manager groups.

Syntax:

GROUP [{"groupname"} filename [/D] [/S] [/C]]

or

GROUP ["groupname" /D [/C]]

groupname Program Manager group name string.
filename Program file(s) to add to the group.
/D Delete a group.
/S Process sub-directories.
/C Indicates that parameter groupname refers to a common group.

program files:

Only valid program files can be inserted into the Program Manager groups, and all other file types are ignored. A valid program file must have an extension of either .EXE, .COM, .BAT, .PIF or any filename that has a File Extension Association.

error message:

The command GROUP uses Dynamic Data Exchange to communicate with the Program Manager. A request is sent to the Program Manager, and the Program Manager returns either a positive or negative acknowledgment, depending on whether the request is carried out or not.

When a request fails then the error message **Request Denied** is displayed. The most common reasons for this error occurring is :-

1. WinOne can not link to the Program Manager.
2. an attempt to delete a Program Manager group that does not exist.
3. an attempt to insert a program file into a Program Manager group that is full. A Program Manager group can only contain a fixed number of items or programs. Typically, only 50 items or programs are allowed in any one Program Manager group.
4. an attempt to manipulate a common group when the currently logged in user does not have system administrator privileges .

Note:

When the /C switch is not specified then parameter groupname name refers to a private group. When the /C switch is specified do not include the "[Common]" part to specify a common group name. For example, when WinOne is installed on Windows NT it is placed in a common Program Manager group. The parameter groupname is specified as "WinOne", not "WinOne [Common]", and the /C switch must also be specified.

The /C switch is ignored when WinOne is running on Windows 95 or Win32s, since there are no common Program Manager groups in Windows 95 or Win32s.

System administrator privileges are required to manipulate common groups or items contained in a common group.

Examples on using GROUP.

GROUP Example

Example 1:

Assume the current directory is D:\WIN_ONE. To insert all .EXE files into a Program Manager group **WinOne Command Shell**, enter at the WinOne prompt :-

```
GROUP "WinOne Command Shell" *.EXE
```

Example 2:

To insert all program files into the Program Manager group **Windows App**, that is, files contained in the directory D:\WINDOWS, including all sub-directories, enter at the WinOne prompt :-

```
GROUP "Windows App" D:\WINDOWS\*.* /S
```

Example 3:

To delete the Program Manager group **Windows App**, and insert the program SYSEDIT.EXE into the group **Windows App**, enter at the WinOne prompt :-

```
GROUP "Windows App" D:\WINDOWS\SYSEDIT.EXE /DS
```

Example 4:

To delete the Program Manager group **Windows App**, enter at the WinOne prompt :-

```
GROUP "Windows App" /D
```

Command MSGBOX

Function:

Display a message in a pop up window. The button pressed to close the window is stored in an environment variable called MSGBOX.

Syntax:

MSGBOX "title" "text" [flags]

title A string which specifies the window caption.
text A string containing a sequence of characters which specifies the message to display
flags A number which specifies the following attributes :-

Button :-

- 0 OK button (default).
- 1 OK and Cancel buttons.
- 2 Abort, Retry, and Ignore buttons.
- 3 Yes, No, and Cancel buttons.
- 4 Yes and No buttons.
- 5 Retry and Cancel buttons.

Icons :-

- 0 No icon (default).
- 16 Stop icon.
- 32 Question icon.
- 48 Attention icon.
- 64 Information icon.

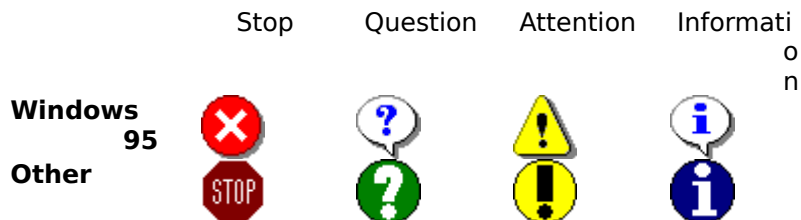
Button action :-

- 0 First button is the default.
- 256 Second button is the default.
- 512 Third button is the default.

To specify more than one attribute simply add the values together

Note:

The following icons are displayed under the respective operating systems :-



To include a new line character in parameter text use a tilda character (ie. '~').

Example on using MSGBOX.

Also see command EDITBOX and DISPBOX.

MSGBOX Example

Example:

To display a message box that contains an information icon, OK and Cancel buttons where the Cancel button is the default, enter at the WinOne prompt :-

MSGBOX "Example" "A sample message box." 321

where the value 321 is calculated as 1 (OK and Cancel buttons) + 64 (Information icon) + 256 (Second button is the default).

Command OPENBOX

Function:

Display a file and directory selection pop up window suitable for retrieving an existing file name. The full path of the file is stored in an environment variable called OPENBOX.

Syntax:

OPENBOX "title" spec1;spec2;spec3;...

title A string which specifies the window caption.

spec1;spec2;spec3;... Specifies the file names which will be listed.

Note:

Parameters spec1;spec2;spec3;... can contain wildcard characters.

Also see command SAVEBOX.

Command **POSTMSG**

Function:

Post a message to a window.

Syntax:

POSTMSG [hwnd | "title" | "class"] msg wparam lparam

hwnd	Specifies the window handle.
title	A <u>string</u> that specifies the window caption or part of the window caption.
class	A <u>string</u> that specifies the window class name or part of the window class name.
msg	Number that represents the message to send
wparam	Number that represents the wParam value.
lparam	Number that represents the lParam value.

Note:

Parameters msg, wparam and lparam are integer numbers.

Also see command TASKS.

Command PRINT

Function:

Print a text file via the Printer Manager.

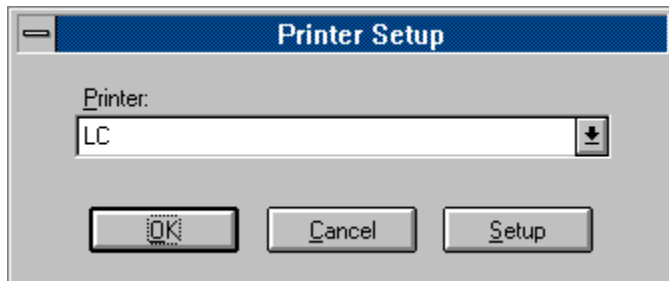
Syntax:

PRINT [[drive:][path]filename] [/tabstop] [/S] [/N] [/L]

filename	Specifies the file to print.
/tabstop	Number between 1 to 8 inclusive. Specifies the number of space characters to use to expand tabstop characters.
/S	Small font size.
/N	Normal font size. This is the default font size used to print a file.
/L	Large font size.

Print Setup:

Type PRINT with no parameters to display the Printer Setup window :-



Select the printer to configure and press the Setup button to change that printers configuration. Changes to a printers configuration is reflected system wide.

Note:

Wildcards are not allowed for parameter filename.

Command SAVEBOX

Function:

Display a file and directory selection pop up window suitable for retrieving a file name that does not exist. The full path of the file is stored in an environment variable called SAVEBOX.

Syntax:

SAVEBOX "title" spec1;spec2;spec3;...

title A string which specifies the window caption.

spec1;spec2;spec3;... Specifies the file names which will be listed.

Note:

Parameters spec1;spec2;spec3;... can contain wildcard characters.

Also see command OPENBOX.

Command SENDKEYS

Function:

Send a number of key strokes to the specified window.

Syntax:

SENDKEYS "who" "keys" [/C]

who	A <u>string</u> that specifies the window to send the keys to. This can be either a window caption or a window class. By default who is a window caption, use the /C switch to specify a window class.
keys	A <u>string</u> containing a sequence of characters which specifies the key strokes to be sent.
[/C]	Specifies that parameter who refers to a window class.

Specifying the key strokes

The string containing the key strokes can include any print-able ASCII characters, that is, characters in the ASCII range of 32 to 126 inclusive. However, there are a number of characters that have a special meaning, these include :-

Short Hand Character

~ Press the **ENTER** key.

Change State Characters. These characters change the state of the next key pressed or a Sub Group of key presses.

+ Hold down the **SHIFT** key.

^ Hold down the **CONTROL** key.

% Hold down the **ALT** key.

Group Characters.

{ } Used to specify a Special Key and/or give a key a Repeat Count.

() Used to group a number of keys together, that is, to specify a Sub Group.

Note:

The maximum number of characters that can be sent is 128.

Examples for parameter keys.

Special Keys

There are a number of Special Keys that can not be specified with a single character, instead, they are specified by enclosing the name of the key in braces. For example, the Caps Lock key is specified by "{CAPLOCK}".

Valid Special Key names include :-

CAPSLOCK	Caps Lock
NUMLOCK	Num Lock
SCROLLLOCK	Scroll Lock
ESCAPE or ESC	Escape
ENTER	Enter
PRTSC	Print Screen
TAB	Tab
BREAK	Break
BACKSPACE or BS	Back Space
DELETE or DEL	Delete
INSERT	Insert
LEFT	Left Arrow
RIGHT	Right Arrow
UP	Up Arrow
DOWN	Down Arrow
PGUP	Page Up
PGDN	Page Down
HOME	Home
END	End
F1 to F12	Function keys F1 to F12

Similarly, the characters ~+^%(){ } each have some special purpose or meaning, and to specify the equivalent key, the character must be enclosed in braces. For example, the plus key is specified by "{+}" and not "+".

Repeat Count

Any key except the SHIFT, CONTROL and ALT keys can be repeated or pressed a number of times, by enclosing the key and the repeat count, separated by a space character, in braces. For example, moving the cursor left 10 positions (ie. pressing the left arrow key 10 times), is specified by "{LEFT 10}".

Sub Groups

To change the state of a number of keys, that is, hold down either the SHIFT, CONTROL or ALT key, while pressing a number of other keys, then the other keys must be enclosed in brackets. For example, to move the cursor back three words, that is, hold down the CONTROL key, while pressing the left arrow key three times, is specified by "`^({LEFT}{LEFT}{LEFT})`".

SENDKEYS Examples

Parameter keys	Valid	Output
"abcDEF"	OK	abcDEF
"abcdef{BACKSPACE}"	OK	abcde
"abcdef{BACKSPACE 3}"	OK	abc
"BACKSPACE"	OK	BACKSPACE
"abcghi{LEFT 3}DEF"	OK	abcDEFghi
"{a 5}"	OK	aaaaa
"+{a 5}"	OK	AAAAA
"abc+(def)"	OK	abcDEF
"+abc+def"	OK	AbcDef
"{}"	OK	{
"{+}"	OK	+
"{+ 5}"	OK	++++
"(abc)"	OK	abc
"{({}abc{})}"	OK	(abc)
"+{+a 5}"	Error - '+a' is not a valid key	
"{abc}"	Error - 'abc' is not a valid key	
"{DELETE}"	Error - missing brace	

Command SHELL

Function:

Set the Windows default start up shell or program list.

Syntax:

SHELL [prog1,prog2,...]

prog1,prog2,... Specifies the list of programs to start when Windows is started.

Note:

Must have system administrator privileges to set the start up shell or program list.

Each program can include an optional drive and path. When a program is not located in the PATH environment variable, then the program name must be a fully qualified filename, including the drive and path.

When specifying the list of programs each program must be separated by comma's.

When no parameters are specified, command SHELL will display the current shell setting, that is, the current list of programs Windows will run on start up.

Also see Program Manger Replacement.

Warning

Before setting the default start up shell make a note of the current shell setting.

Should any difficulties arise with the new shell, then restore the previous shell, either by using the SHELL command or by manually setting the System Registry key value (ie. by running REGEDT32.EXE via the Task Manager) :-

```
HKEY_LOCAL_MACHINE
  \SOFTWARE
    \Microsoft
      \Windows NT
        \CurrentVersion
          \Winlogon
            \Shell=pervious shell setting
```

Command TASKS

Function:

Display a list of the current tasks or performs a specified action on a task.

Syntax:

TASKS [[hwnd | "title" | "class"] state] [/V]

hwnd	Specifies the window handle.
title	A <u>string</u> that specifies the window caption or part of the window caption.
class	A <u>string</u> that specifies the window class name or part of the window class name.
state	Specifies the action to perform :- CLOSE end a program MIN display window as icon MAX display maximised window SHOW display a previously hidden window HIDE hide window RESTORE display window in it original size and position ONTOP set a window to be the top most window NOTONTOP reset a window to no longer be the top most window TERMINATE force a program to exit (ie. terminate) IDLE set idle process priority NORM set normal process priority HIGH set high process priority
/V	Verbose mode display

Note:

Type TASKS without an parameters to display a list of window handles and captions.

Verbose mode will also display window class names.

When the parameter hwnd, title or class is not specified then the action is performed on WinOne itself.

Command VIEWICON

Function:

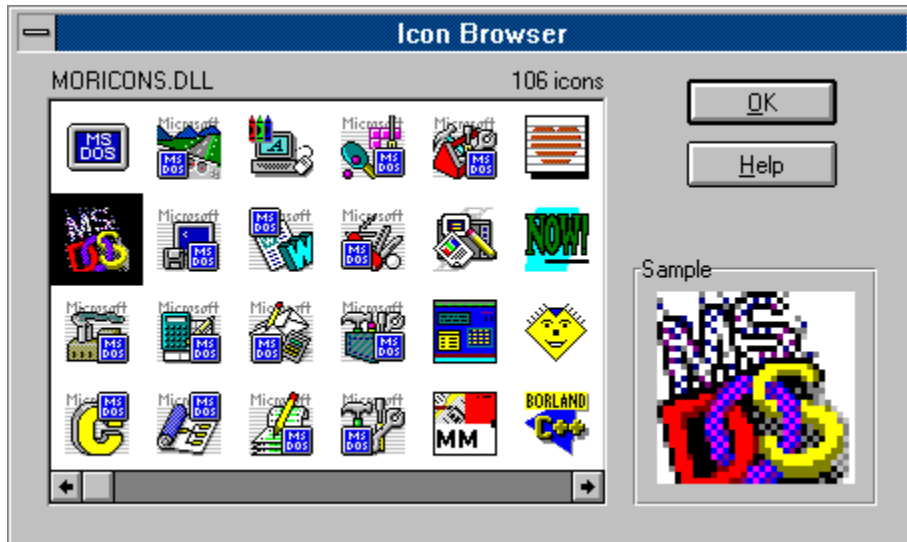
Displays icons in .EXE, .DLL and .ICO files.

Syntax:

VIEWICON [[[drive:]path]filename]

Description:

The VIEWICON command will display a window containing all icons in the specified file(s), as follows :-



To determine the filename for an icon, position the mouse cursor over the icon and press the left mouse button.

Command VOLUME

Function: Set or get the volume level for an audio device.

Syntax: **VOLUME** [level]

level A number specifying the volume level.

Note: The volume level is a value between 0 and 255 inclusive. A value of 0 specifies no sound and a value of 255 specifies the loudest volume.

When parameter level is not specified then the current volume level is displayed.

When no suitable audio device is installed or the installed audio device does not support software volume control then a suitable error message is displayed.

Also see command PLAY and BEEP.

Command WALLPAPER

Function:

Set or remove the Desktop Wallpaper.

Syntax:

WALLPAPER [[drive:][path]]filename [/C] [/T]

filename	Specifies the bitmap file name.
/C	Centre the bitmap.
/T	Title the bitmap.

Note:

Type WALLPAPER without any parameters to remove the current Wallpaper from the Desktop.

External Commands

External Commands are programs that use the main WinOne window for their screen input and output.

Command ARGV [\(External\)](#)

For more information on command ARGV, see the WOIO.HLP help file.

Command **DOS2UNIX** (External)

Function:

Convert DOS text files to Unix text files.

Syntax:

DOS2UNIX [drive:][path]filename [/V]

filename Specifies the file(s) to convert.
/V Show version information.

Note:

This command will not alter a Unix text file.

Also see command [UNIX2DOS](#).

Command **DUPLICAT** (External)

Function:

Locate all duplicate files on the specified drive(s).

Syntax:

DUPLICAT [drive1 drive2 ...] [/N] [/S] [/C]

drive1 drive2 ... Specifies the disk drive(s) to search.

/N By filename only. This will locate files that have the same name. The files found can differ in size or content.

/S By size and filename. This is the default, when no switches are specified. This will locate files the have the same name and size. The files found can differ in content.

/C By size, filename and content. This will locate all files that are exactly the same.

Note:

When no drives are specified DUPLICAT will search all non-removable hard disks.

The /N switch is the fastest and the /C switch is the slowest.

Command FUNC (External)

Function:

Call a Win32 API function at run time.

Syntax:

FUNC module function [params ...] [/V]

module Specifies the module name that contains the function to be called. Can be either a file name (eg. SHELL32.DLL) or an already loaded module name (eg. USER32).
function Specifies the name of the function to be called. Function names are case sensitive.
params Specifies the parameters to be past to the function. Valid parameters can include the following types :-

1. 32 bit integer numbers.
2. strings, where the first character in the string cannot be a digit.
3. NULL, which specifies a NULL pointer (ie. a value of 0).
4. BUF, which represents a pointer to an internal buffer, which can be used to store information that some functions need.

/V Show version information.

Note:

Command FUNC can only call PASCAL type Win32 functions and can not call CDEL type functions. CDEL functions have their parameters pushed onto the stack before the function is called and popped off the stack after the function returns. Similarly, PASCAL functions have their parameters pushed onto the stack before the function is called, however, the function that is been called is then responsible for popping the parameters off the stack before the function returns.

All Windows Dynamic Link Libraries contain functions (ie. exports) that can be called at run time. There are a number of modules that are loaded automatically when Windows starts, including KERNEL32, USER32 and GDI32.

The return value is stored in the environment variable FUNCA as a sequence of digits.

When the parameter BUF is used, then the contents of the internal buffer is stored in the environment variable FUNCB.

Examples on using FUNC.

Also see Batch Programs.

FUNC Examples

Example 1:

To get the window handle (ie. hwnd) for the WinOne window enter at the WinOne prompt :-

```
FUNC user32 FindWindow NULL "WinOne"
```

The environment variable FUNCA will contain the hwnd value.

Example 2:

Consider the following batch program :-

```
@ECHO off  
  
FUNC user32 GetDesktopWindow  
FUNC user32 ArrangeIconicWindows %FUNCA%  
  
IF (%FUNCA%==0) {  
  COLOUR error  
  ECHO Unable to arrange icons.  
  ECHO.  
}
```

The above batch program arranges all the iconic windows on the desktop. This batch program performs the same function as clicking on the Arrange Icons button in the Task Manager.

Command LOGO (External)

Function:

Change the Windows startup logo.

Syntax:

LOGO [drive:][path][filename] [/V]

filename Specifies the file containing the new logo. This file MUST be a bitmap format file.

/V Show version information.

Note:

Type LOGO without any parameters to restore the default startup logo.

When WinOne is running on Windows 95 then parameter filename, which specifies the file containing the new logo, MUST be a 320x400x256 colour bitmap, otherwise an appropriate error message is displayed and the logo is NOT changed.

When WinOne is running on Win32s then parameter filename MUST be a Run Length Encoded 16 colour bitmap, which is no larger than 48K in size, otherwise an appropriate error message is displayed and the logo is NOT changed. The first time command LOGO is used to change the Windows logo under Win32s, a backup copy of WIN.COM is made to the file WINBAK.COM. Should anything go wrong, then you will be able to get back into Windows by running WINBAK.COM at the DOS prompt.

Run Length Encoded (RLE) Bitmaps

Run Length Encoding refers to the compression method that is used to compress a bitmap. Typically, a bitmap compressed using this method will have a file extension of .RLE, but this is not generally the case, so command LOGO checks the bitmap internally to verify that it does meet the specified requirements before changing the start up logo in Win32s.

Due to the nature of RLE bitmaps, that is, it replaces long runs of the same colour with the number of times the colour occurs, it is possible to compress a bitmap and end up with a file that is larger than the un-compressed bitmap. Typically, a bitmap that has relatively large areas of the same colour, for example a black background colour, should compress to no greater than 48K.

There are many graphics programs available that can be used to convert a standard bitmap to a Run Length Encoded bitmap.

Command **MERGE** (External)

Function:

Merge two files together.

Syntax:

MERGE first second destination [/V]

first	Specifies the first file.
second	Specifies the second file.
destination	Specifies the destination file.
/V	Show version information.

Note:

The file first is copied to the file destination, then the file second is appended to the file destination.

Also see command [SPLIT](#).

Command SBANNER (External)

Function: Display small horizontal characters.

Syntax: **SBANNER** [msg] [/V]

msg Sequence of characters.
/V Show version information.

Sample:

```
##      #####      #####      ###
#####  ##  ##  ##  ##
##  ##  ##  ##  ##      #####  #####  #####
##  ##  #####  ##      ##  ##  ##  ##  ##  ##
#####  ##  ##  ##      #####  ##  ##  ##
##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##  ##
##  ##  #####  #####  ### ##  ##  ##  ##  ##
```

Note: Also see command GOTHIC.

Command **SPLIT** (External)

Function:

Split a file into two files.

Syntax:

SPLIT [drive:][path]filename size [/V]

filename	Specifies the file to split.
size	Specifies the size in bytes for the first file. Size bytes from the file filename will be placed into the first file and the remaining bytes will be placed into the second file.
/V	Show version information.

Note:

The resulting two files have an extension of .1 for the first file and .2 for the second file.

Wildcards can not be used for parameter filename.

Also see command MERGE.

Command TOUCH (External)

Function: Touch the date and time for a file(s).

Syntax: **TOUCH** [drive:][path]filename [date] [time] [/V]

filename	Specifies the file(s) to touch.
date	Specified according to the international format
time	hh:mm:ss
/V	Show version information.

Note: The system date and/or time is used when parameters date and/or time are not specified.

Parameter date is specified according to the international format set in the Control Panel. For example, the USA date format is typically mm-dd-yyyy and the Australian date format is typically dd-mm-yyyy.

When parameter filename specifies a file that does not exist, then that file is created and then touched.

Command TOUCH can not be used to change the date or time for a directory.

Command UNCOMP (External)

Function: Uncompress a UNIX compressed file (ie. .Z file).

Syntax: **UNCOMP** source destination [/V]

source	Specifies the file to uncompress.
destination	Specifies the file name of the uncompressed file.
/V	Show version information.

Note: Parameters source and destination can contain an optional path. When the destination file exists, then UNCOMP will fail and display an error message.

Wildcards can not be used for parameters source or destination.

Also see command UNTAR.

Special note: There should be no need to ever tar or compress a file under Windows. There are a number of archive formats that are supported by both UNIX and Windows (eg. ZIP archive format), which should be used instead.

Command **UNIX2DOS** (External)

Function:

Convert Unix text files to DOS text files.

Syntax:

UNIX2DOS [drive:][path]filename [/V]

filename Specifies the file(s) to convert.
/V Show version information.

Note:

This command will not alter a DOS text file.

Also see command [DOS2UNIX](#).

Command **UNTAR** (External)

Function:

Untar a UNIX tape archive (ie. .TAR file).

Syntax:

UNTAR [drive:][path]filename[.TAR] [/D] [/V]

filename	Specifies the file to untar.
/D	Display a directory listing of the tar archive.
/V	Show version information.

Note:

A tar file must have a file extension of .TAR, since this is the only way to determine whether or not a file is a tar file.

The directory structure inside the tar file is recreated and the file names inside a tar file are mapped to long file names.

When a file already exists, then the user will be prompted whether to over write the existing file and if not a new file name will be requested. It is quite possible for a tar file to contain more than one file name that will map to a single long file name, since under UNIX filenames are case sensitive. For example, Read.Me and read.me will both be mapped to READ.ME under Windows.

Wildcards can not be used for parameter filename.

Also see command UNCOMP.

Special note:

There should be no need to ever tar or compress a file under Windows. There are a number of archive formats that are supported by both UNIX and Windows (eg. ZIP archive format), which should be used instead.

Command UNZIP (External)

Function:

Extract files contained in a .ZIP archive.

Syntax:

UNZIP source.zip [destination] [/E] [/X] [/T]

source.zip	Specifies the archive file name.
destination	Specifies the directory to place the extracted file(s) in. When a destination sub-directory is not specified then the current directory is used.
/E	Extract file(s), ignoring paths.
/X	Extract file(s), recreating the directory structure. This is the default.
/T	Test contents of archive.

Note:

Command UNZIP extracts all files from the specified archive.

When the destination sub-directory does not exist, it is created.

Also see command [ARCH](#).

Command **UUDECODE** (External)

Function: Convert a uuencode text file back to a binary file.

Syntax: **UUDECODE** *textfile*
textfile Specifies a previously uuencoded file to convert.

Note: Wildcards can not be used for parameter *textfile*.
By convention uuencoded files generally have a .UU or .UUE file extension.
Also see command [UUENCODE](#).

Command **UUENCODE** (External)

Function:

Convert a binary file to a text file.

Syntax:

UUENCODE *binaryfile* *textfile*

binaryfile Specifies the binary file to convert.

textfile Specifies the text file to create.

Note:

Wildcards can not be used for parameters *binaryfile* or *textfile*.

By convention uuencoded files generally have a .UU or .UUE file extension.

Also see command [UUDECODE](#).

Standard Batch Commands

Command CALL

Function:

Run a second batch program, then returns to the first batch program.

Syntax:

CALL [drive:][path]filename [batch-parameters]

[drive:][path]filename Batch program name and location.

batch-parameters Any command line information that is used by the batch program.

Note:

When there is no need to return to the original batch program, then use the following syntax, without the CALL command :-

[drive:][path]filename [batch-parameters]

When the CALL command is used at the WinOne prompt to run a batch program, the batch program will be executed in a separate window.

Also see command STOP and Batch Programs.

Command ECHO

Function:

Display a message or turns echo on or off.

Syntax:

ECHO [message]

or

ECHO [ON | OFF]

message	Sequence of characters to display.
ON	Display commands as they are carried out.
OFF	Do not display commands as they are carried out.

Note:

Type ECHO with no parameters to display whether echo is on or off.

To display a blank line type ECHO followed by a dot character :-

ECHO.

Also see command [SAY](#) and [Batch Programs](#).

Tip:

To prevent a single command in a batch program from being displayed, put an at sign in front of the command, for example :-

@ECHO OFF

Command **ENDLOCAL**

Function: End the localisation of environment variables.

Syntax: **ENDLOCAL**

Note: Also see command SETLOCAL and Batch Programs.

Command FOR

Function:

Perform a command for each file in the specified set of files or perform a command a given number of times.

Syntax:

FOR %variable **IN** (set) **DO** command

or

FOR %variable **IS** num1 **TO** num2 [**STEP** num3] **DO** command

%variable	Specifies a replaceable parameter
(set)	Specifies a set of files. Wildcards can be used and file names can be separated by comma's.
num1	Number specifying the start of the range.
num2	Number specifying the end of the range.
num3	Number specifying the increment value. The default value is 1 when no STEP value is specified.
command	Specifies the command to be performed for each file found in the set.

Note:

When using the FOR command inside a batch program, use %%variable and not %variable.

The parameters num1, num2 and num3 may either be specified as integer or real numbers.

Also see Command Grouping and Batch Programs.

Command GOTO

Function:

Switch to another part of the batch program, and continue executing the program from that point.

Syntax:

GOTO label

label Sequence of characters.

Note:

The label must also appear on its own line elsewhere in the batch program, preceded by a colon, for example :-

```
GOTO end  
...  
...  
...  
:end
```

Also see command [GOSUB](#) and [Batch Programs](#).

Command IF

Function:

Perform conditional processing in a batch program.

Syntax:

IF [NOT] condition command

or

IF [NOT] condition (command(s)) **else** command

or

IF [NOT] condition { command(s) } **else** command

condition	Is one of the following :- (expr)	Evaluate the parameter expr. A zero result is false, otherwise the condition is true. Parameter expr follows the same rules as parameter expr for command <u>CALC</u> . The brackets around parameter expr are mandatory and must be specified.
	ERRORLEVEL number	Specifies a true condition if the last program returned an exit code greater than or equal to number.
	EXIST filename	Specifies a true condition if the filename exists.
	EXISTCLASS "text"	Specifies a true condition if the window class "text" exists.
	EXISTWINDOW "text"	Specifies a true condition if the window caption "text" exists.
	string1==string2	Specifies a true condition if string1 and string2 are the same.
command	Specifies the command to carry out if the condition is met. Parameter command can also be another IF command.	
NOT	Carry out the command only if the condition is false.	

Note:

The parameters string1 and string2 do not need to be enclosed in quote characters.

When using the IF ELSE syntax, the brackets or braces around parameter command(s) are mandatory and must be specified. See Command Grouping for more information.

Examples on using IF.

Also see command CALC and Batch Programs.

IF Examples

Example 1:

Consider the following batch program :-

```
:loop  
GETKEY Option:  
IF "%GETKEY%"="" {  
    GOTO loop  
} ELSE IF "%GETKEY%"="A" {  
    ECHO Option A selected  
} ELSE IF "%GETKEY%"="B" {  
    ECHO Option B selected  
} ELSE {  
    ECHO Unknown option!  
    GOTO loop  
}
```

This batch program displays the "Option:" prompt and waits for the user to either press the A or B key. It will continue to prompt the user, until either the A or B key is pressed.

Example 2:

Consider the following batch program :-

```
SET VAR=1  
IF ( %VAR% == 1 ) {  
    SET VAR=2  
    ECHO %VAR%  
}  
ECHO %VAR%
```

This program displays 1 on the first line and 2 on the second line. See [Command Grouping](#) for more information on using braces instead of brackets to group commands.

Example 3:

Consider the following erroneous batch program :-

```
SET VAR=0  
IF ( %VAR% != 0 ) {  
:loop  
    ECHO %VAR%  
} ELSE {  
    GOTO loop  
}  
}
```

This program will result in a **No such label** error, since jumping into an IF statement is not permitted, however, jumping out of an IF statement is permitted.

Command PAUSE

Function:

Suspend processing of a batch program and displays a message.

Syntax:

PAUSE [message]

message Sequence of characters.

Note:

Type PAUSE without an parameters to display the message :-

 Press any key to continue . . .

Also see command [ASK](#) and [Batch Programs](#).

Command REM

Function: Allows comments inside a batch file.

Syntax: **REM** anything
anything Sequence of characters.

Note: REM commands are just ignored by WinOne.

Also see [Batch Programs](#).

Command SETLOCAL

Function: Set the localisation of environment variables.

Syntax: **SETLOCAL**

Note: Changes to the current environment variables are un-done with the ENDLOCAL command.

Command SETLOCAL is cumulative, that is. when command SETLOCAL is used 5 times then 5 ENDLOCAL commands are required to return to the original environment variables.

The original environment variables are not automatically restored when a batch program ends, the ENDLOCAL command is required.

Also see command [ENDLOCAL](#) and [Batch Programs](#).

Command SHIFT

Function:

Change the position of replaceable parameters in a batch program.

Syntax:

SHIFT

Note:

The SHIFT command changes the values of replaceable parameters %0 through to %9, by copying each parameter into the previous one.

When there are more than 10 command line parameters, each will be shifted one at a time into %9.

Also see [Batch Programs](#).

Enhanced Batch Commands

Command ASK

Function: Ask a yes/no question and set the errorlevel respectively.

Syntax: **ASK** [message]

message Sequence of characters.

Note: Type ASK without any parameters to display the message :-

Continue (Y/N)?

The errorlevel is set to 0 for a yes response, 1 for a no response.

Example on using ASK.

Also see command PAUSE and Batch Programs.

ASK Example

Consider the following batch program, which will display a yes/no message, and depending on the user response the program will either continue or end :-

```
ASK Are you sure (Y/N)?  
IF ERRORLEVEL 1 GOTO no  
...  
...  
...  
:no
```

Similarly, the following batch program will display a yes/no message and either loop or end :-

```
:loop  
...  
...  
...  
ASK Again (Y/N)?  
IF ERRORLEVEL 1 GOTO no  
GOTO loop  
:no
```

Command BEEP

Function:

Send a beep to the system speaker.

Syntax:

BEEP [ON | OFF]

or

BEEP frequency duration [...]

ON Enable the system sound.

OFF Disable the system sound.

frequency Specifies the sound frequency, in hertz. Must be in the range 37 through 32767.

duration Specifies the sound duration, in milliseconds. 1000 milliseconds is equal to 1 second.

Note:

Type BEEP without any parameters to send a beep to the system speaker.

When the system sound is disabled then no sound will be played or heard for the system beep. The system beep is used by Windows as a audio warning signal and is generally sounded when a message window is displayed. Changes to the system sound settings are reflected system wide.

Also see command [PLAY](#), [VOLUME](#), [Event Sounds](#) and [Batch Programs](#).

Command Box

Function:

Display a box in one of four pre-defined formats.

Syntax:

BOX left top right bottom [type]

left top right bottom Screen co-ordinates of box.

type Specifies a number 1 to 4, which represents one of the following box formats :-



Note:

Co-ordinates start from 0. For example, the first character on the screen is at co-ordinate 0, 0.

When parameter type is not specified then format 1 is used.

Also see [Batch Programs](#).

Command CALC

Function:

Perform basic arithmetic calculations in a batch program. The result is either displayed or stored in an environment variable called CALC.

Syntax:

CALC [?] *expr*

? Display the result only. Does not set the environment variable.
expr Sequence of tokens, that can include the following :-

+	Addition
-	Subtraction
*	Multiplication
/ or \	Division
%	Remainder or modulus
~	Bitwise NOT
^	Bitwise XOR
&	Bitwise AND
	Bitwise OR
<<	Left bit shift
>>	Right bit shift
&&	Logical AND
	Logical OR
!	Logical NOT
<	Boolean less than
>	Boolean greater than
<=	Boolean less than or equal to
>=	Boolean greater than or equal to
==	Boolean equal to
!=	Boolean not equal to
(Left bracket
)	right bracket
<u>numbers</u>	Either integer or real numbers
<u>strings</u>	Sequence of characters
<u>functions</u>	Scientific functions

Warning

When using tokens that include characters that have a special meaning (eg. <, >, & etc) then enclose the complete *expr* in brackets, as follows :-

CALC [?] (*expr*)

When in doubt, use the above syntax.

Note:

The normal operator precedence is assumed, unless the left and right brackets are used to over-ride the default precedence.

Examples on using CALC.

Also see command LET and Batch Programs.

Scientific functions

The following table includes all the functions that may be included in an expression :-

Name	Calculates	Syntax
ACOS	Arc cosine	ACOS(x)
ASIN	Arc sine	ASIN(x)
ATAN	Arc tangent	ATAN(x)
COS	Cosine	COS(x)
COSH	Hyperbolic cosine	COSH(x)
EXP	Exponential eto the x	EXP(x)
LOG	Logarithm (base 10)	LOG(x)
LN	Natural Logarithm (base e)	LN(x)
POW	x to the power of y	POW(x, y)
SIN	Sine	SIN(x)
SINH	Hyperbolic sine	SINH(x)
SQR	Square root	SQR(x)
TAN	Tangent	TAN(x)
TANH	Hyperbolic Tangent	TANH(x)
ABS	Absolute value	ABS(x)
SGN	Sign of x	SGN(x)
REAL	Convert an intger to a real	REAL(x)
INT	Convert a real to an integer	INT(x)

Note:

All the above functions accept both integer or real numbers.

Both x and y may be specified as further expressions to evaluate. Similarly, the above functions may be repeatedly nested inside other functions.

CALC Examples

Example 1:

Consider the following command line :-

```
CALC ? 1 + 1 > 1
```

The result of the above arithmetic expression $1 + 1$ is 2. Since the output has been redirected, the result will be written to the file "1". However, if brackets were used around the expression, as follows :-

```
CALC ? ( 1 + 1 > 1 )
```

then the result of the above boolean expression $(1 + 1 > 1)$ is the value 1 (ie. true), which will be display on the screen.

Example 2:

Consider the following command line :-

```
CALC ? ( "a string" == "a string" && "hello" != "world" )
```

The result of the above boolean expression is the value 1 (ie. true), which is displayed on the screen.

Example 3:

Calculate the cosine of 2.5 degrees :-

```
CALC ? COS( (2.5-INT(2.5/360)*360) * 3.14159265359 / 180.0 )
```

will display the value 0.999048221582.

Example 4:

Calculate the distance between the two points (1, 1) and (3, 4) :-

```
CALC ? SQR( POW(3-1, 2) + POW(4-1, 2) )
```

will display the value 3.60555127546

Example 5:

Consider the following batch program :-

```
@ECHO off  
CLS  
SET text=hello, world  
STRLEN text  
CALC ( 80 - %strlen% ) / 2  
LOCATE %calc% 0  
ECHO %text%
```

This program will clear the screen and display the text 'hello, world', centred on the first line.

Command COLOUR

Function:

Sets foreground and background colours inside a batch program.

Syntax:

COLOUR foreground [background]

foreground Text foreground colour .

background Text background colour.

Valid foreground and background colours include :-

1. System Colours. These colours are predefined and the actual colour displayed depends on which colour is associated to each of the following :-

FILENAME	- file and path names
NUMBER	- numbers
TEXT	- plain text
HIGHTEXT	- highlighted text
BOLDTEXT	- bold text
ENVNAME	- environment variable names
ENVSTR	- environment variable strings
ERROR	- error messages
LHS	- left hand side of equal signs
HIGHLHS	- highlighted left hand side of equal signs
RHS	- right hand side of equal signs
FILEDATE	- file dates
FILETIME	- file times
FILEATTRIB	- file attributes
FILEDESC	- file descriptions
BACKGRD	- background

2. Fix colours. These colours are fixed. :-

WHITE
GRAY
BLACK
RED
GREEN
BLUE
CYAN
MAGENTA
YELLOW

Note:

When the background parameter is not specified the BACKGRD system colour is assumed.

Also see [Custom Colours and Colour Schemes](#) and [Batch Programs](#).

Command COMMA

Function:

Insert comma's into a number. The comma delimited number is stored in an environment variable called COMMA.

Syntax:

COMMA [number]

number Specifies the number to insert comma's into.

Note:

Also see [Batch Programs](#).

Command DATA

Function:

Clear or add items to a global list.

Syntax:

DATA [item, ...]

item Sequence of characters.

Note:

When no parameters are specified for command DATA then the global list is cleared.

Each item is separated by comma's and multiple DATA commands simply append items to the global list.

Each batch program receives its own global list. A batch program can not affect the global list of another batch program. Also, the global list is automatically discarded when a batch program ends.

An item is retrieved from the global list by using command READ.

Example on using DATA.

Also see command READ and Batch Programs.

DATA Example

Consider the following batch program :-

```
@ECHO OFF  
SETLOCAL  
  
REM Random Desktop Wallpaper Changer  
REM for Windows NT, 95 and Win32s  
  
REM The first DATA statement is the number  
REM of bitmap files and the remaining DATA  
REM statements contain the full path for each  
REM bitmap file that will be used.  
DATA 3  
DATA d:\pics\file1.bmp  
DATA d:\pics\file2.bmp  
DATA d:\pics\file3.bmp  
  
REM Get the number of bitmap files.  
READ  
  
REM Randomly pick one of the files.  
LET selected=(%random_number% %% %read%)  
READ %selected%  
READ  
  
REM Set the selected file as the next wallpaper.  
SETINI Desktop "Wallpaper=%read%" win.ini  
  
ENDLOCAL
```

This batch program will set the next wallpaper Windows will display the next time Windows is started. The blue lines (ie. the DATA statements) need to be set so that the first DATA statement specifies the number of bitmap files and the remaining data statements specify the full path of each individual bitmap file. For example, in the above batch program there are 3 bitmap files :-

1. d:\pics\file1.bmp
2. d:\pics\file2.bmp
3. d:\pics\file3.bmp

To have WinOne change the wallpaper each time WinOne is started, then simply add the above batch program to the STARTUP.BAT file after the DATA statements have been set.

Command DATE2SER

Function:

Convert a date to a serial date value. The serial date value is stored in an environment variable called DATE2SER.

Syntax:

DATE2SER date

date Specifies the date to convert.

Note:

Parameter date is specified according to the international format set in the Control Panel. For example, the USA date format is typically mm-dd-yyyy and the Australian date format is typically dd-mm-yyyy.

[Example](#) on using DATE2SER.

Also see command [SER2DATE](#) and [Batch Programs](#).

DATE2SER Example

Consider the following batch program :-

```
@ECHO OFF  
GETSTR Enter your date of birth ?  
DATE2SER %getstr%  
SET dob=%date2ser%  
  
DATE2SER %date_today%  
CALC %date2ser% - %dob%  
  
ECHO.  
ECHO You have been alive for %calc% days.
```

This batch program demonstrates the use of command DATE2SER to perform arithmetic calculations on dates and simply displays the number of days you have been alive.

Command DIRS

Function: Display the directory stack.

Syntax: **DIRS**

Note: The directory stack can hold at most 10 directories. When more than 10 directories are pushed on to the stack then the oldest directories are removed to make room for the new directories.

Use the commands PUSHD and POPD to manipulate the stack.

Also see Batch Programs.

Command DISKFREE

Function:

Get the amount of free disk space for the specified drive. The free disk space (in bytes) is stored in an environment variable called DISKFREE.

Syntax:

DISKFREE [drive:]

drive: Specifies the drive to use.

Note:

When the parameter drive: is not specified then the current drive is used.

Also see command [DISKUSED](#) and [Batch Programs](#).

Command DISKUSED

Function:

Get the amount of used disk space for the specified drive. The used disk space (in bytes) is stored in an environment variable called DISKUSED.

Syntax:

DISKUSED [drive:]

drive: Specifies the drive to use.

Note:

When the parameter drive: is not specified then the current drive is used.

Also see command [DISKFREE](#) and [Batch Programs](#).

Command **END**

Function: End a batch program.

Syntax: **END**

Note: Also see command STOP and Batch Programs.

Command FILEDATE

Function:

Get the last modified date of a file. The file date is stored in an environment variable called FILEDATE.

Syntax:

FILEDATE [drive:][path]filename

[drive:][path]filename Specifies the file name.

Note:

Relative paths are allowed for parameter path.

Wildcards are not allowed for parameter filename.

The date retrieved is formatted according to the international format set in the Control Panel. For example, the USA date format is typically mm-dd-yyyy and the Australian date format is typically dd-mm-yyyy.

Also see command [FILETIME](#) and [Batch Programs](#).

Command FILESIZE

Function:

Get the size (in bytes) of a file. The file size is stored in an environment variable called FILESIZE.

Syntax:

FILESIZE [drive:][path]filename

[drive:][path]filename Specifies the file name.

Note:

Relative paths are allowed for parameter path.

Wildcards are not allowed for parameter filename.

Also see [Batch Programs](#).

Command FILETIME

Function:

Get the last modified time of a file. The file time is stored in an environment variable called FILETIME.

Syntax:

FILETIME [drive:][path]filename

[drive:][path]filename Specifies the file name.

Note:

Relative paths are allowed for parameter path.

Wildcards are not allowed for parameter filename.

Also see command [FILEDATE](#) and [Batch Programs](#).

Command FILETYPE

Function:

Determine whether a file contains text or binary data. The file type is stored in an environment variable called FILETYPE.

Syntax:

FILETYPE [drive:][path]filename

[drive:][path]filename Specifies the file name.

Note:

When the specified file contains text then the environment variable will be set to TEXT. Similarly, when the file does not contain text then the environment variable will be set to BINARY.

Also see [Batch Programs](#).

Command GETINI

Function:

Get an initialisation file key value or enumerate all the key names in a section. The value is stored in an environment variable called GETINI.

Syntax:

GETINI section key [filename]

section	Specifies the section that contains the desired key.
key	Specifies the key whose value is to be retrieved.
filename	Specifies the initialisation file name.

Note:

Initialisation files generally have a .INI file extension.

The file extension must be included as part of the file name when parameter filename is specified (eg. WIN.INI).

When parameter filename is not specified then WIN_ONE.INI is used. Since WIN_ONE.INI is automatically mapped to the System Registry, then retrieving a key value from WIN_ONE.INI will retrieve the key value from the System Registry.

When parameter key is specified as an empty string (ie. "") then command GETINI will enumerate all the key names contained in the specified section and create a list which contains all the key names separated by comma's.

Example on using GETINI.

Also see command SETINI, GETREG, SETREG and Batch Programs.

GETINI Example

Consider the following batch program :-

```
@ECHO OFF
IF "%1"==" " {
    COLOUR error
    ECHO Section name not specified
    STOP
}

GETINI %1 "" %2
IF "%getini%"==" " {
    COLOUR error
    ECHO Section not found
    STOP
}

FOR %%i IN (%getini%) DO {
    SAY %%i=
    GETINI %1 %%i %2

    REM Must use SAY to print %getini% since we can not
    REM be sure that %getini% will not contain the word
    REM 'off', which has a special meaning for command
    REM ECHO.

    SAY %getini%
    ECHO.
}
```

This batch program demonstrates the use of command GETINI to display all the key names along with their values for the specified section and initialisation file. The section and initialisation file are specified as the first and second parameters, respectively, when running the batch program.

Command GETKEY

Function:

Wait for a single keypress from the user. The character entered is stored in an environment variable called GETKEY.

Syntax:

GETKEY [message]

message Sequence of characters.

Note:

Also see command GETSTR, GETNUM and Batch Programs.

Command GETNUM

Function:

Wait for a sequence of keypresses from the user. The sequence of characters is stored in an environment variable called GETNUM.

Syntax:

GETNUM [message]

message Sequence of characters.

Note:

When the sequence of characters entered by the user is not either an integer or a real number then the environment variable is cleared.

Example on using GETNUM.

Also see command GETKEY, GETSTR and Batch Programs.

GETNUM Example

Consider the following batch program :-

```
@ECHO off

GETSTR Instructions (Y/N) ?
UPPER %getstr%
IF ("%upper%" == "Y") {
    ECHO.
    ECHO In this game, the players, you and the
    ECHO computer, start with a common pile of
    ECHO many stones, usually between 15 and 35.
    ECHO.
    ECHO While alternating turns, you take away
    ECHO one or more stones from the pile.
    ECHO You may not take more than some agreed
    ECHO upon number of stones, usually between
    ECHO three and six.
    ECHO.
    ECHO Whoever is forced to take the last
    ECHO stone loses.
    ECHO.
    ECHO You get to go first.
    ECHO.
}

ECHO.
LET remaining=(%random_number% %% 21) + 15
LET max=(%random_number% %% 3) + 3
LET magic=%max% + 1

ECHO Starting with %remaining% Stones.

:next
ECHO.
GETNUM %remaining% Stones left. Take 1 to %max% stones ?
IF ("%getnum%" == "") {
    ECHO You fool! You must enter the number of stones to take.
    GOTO next
}

IF (int(%getnum%) != %getnum%) {
    ECHO You fool! There are no broken stones in the pile.
    GOTO next
}

IF (%getnum% < 1 || %getnum% > %max%) {
    ECHO You cheat! You may only take 1 to %max% stones.
    GOTO next
}

ECHO Human takes %getnum% stones
IF (%getnum% < %remaining%) {
    LET remaining=%remaining% - %getnum%
    IF (%remaining% == 1) {
```

```

ECHO Hmmm I lost. You must be using a defective computer.
END
}

LET mymove=(%remaining% - 1) %% %magic%
IF (%mymove% == 0) {
    LET mymove=(%random_number% %% %max%) + 1
}

LET left=%remaining% - %mymove%
SAY Hmm %remaining% stones left.
ECHO I take %mymove% stones, leaving %left%.

LET remaining=%left%
IF (%remaining% != 1) GOTO next
}

ECHO Human stuck with final stone.
ECHO Nyah Nyah Nyah. You Lose.

```

The above batch program is an implementation of the classic Stones puzzle game. Each player takes turns at removing stones and the player that is left with the last stone loses. This program demonstrates the use of command GETNUM to retrieve an integer number from the player.

Command GETREG

Function:

Get a System Registry value or enumerate all the names for the specified hkey. The value is stored in an environment variable called GETREG.

Syntax:

GETREG hkey [name]

hkey Specifies the path of the complete key.
name Specifies the name of the value to retrieve.

Note:

Parameter hkey must begin with one of the following System Registry names :-

HKEY_USERS
HKEY_CURRENT_USER
HKEY_LOCAL_MACHINE
HKEY_CLASSES_ROOT

and all the sub-keys must be separated by backslashes (ie. '\ '), For example :-

HKEY_CURRENT_USER\Software\WinOne\WinOne

The following types of values may be retrieved using command GETREG :-

REG_SZ	A null terminated string.
REG_DWORD	A 32 bit number.
REG_DWORD_LITTLE_ENDIAN	Same as REG_DWORD.
REG_DWORD_BIG_ENDIAN	A 32 bit number which is automatically converted to REG_DWORD format.
REG_MULTI_SZ	A list of null terminated strings. Each entry is separated with comma's.

When parameter name is not specified then command GETREG will enumerate all the names contained in the specified hkey and create a list which contains all the names separated by comma's.

Command GETREG is not available under Win32s, since Win32s does not support a System Registry.

Also see command [SETREG](#), [GETINI](#), [SETINI](#) and [Batch Programs](#).

Command GETSTR

Function:

Wait for a sequence of keypresses from the user. The sequence of characters is stored in an environment variable called GETSTR.

Syntax:

GETSTR [message]

message Sequence of characters.

Note:

Also see command GETKEY, GETNUM and Batch Programs.

Command GOSUB

Function:

Jump to another part of a batch program and continue executing the batch program from that point onwards until a RETURN command is found. Execution of the batch program will then return to the next command following the original GOSUB command.

Syntax:

GOSUB label

label Sequence of characters.

Note:

The label must also appear on its own line elsewhere in the batch program, preceded by a colon, for example :-

```
GOSUB subproc  
...  
...  
:subproc  
...  
...  
RETURN
```

GOSUB's can be nested a maximum of 8 levels.

Also see command [GOTO](#) and [Batch Programs](#).

Command LOCATE

Function: Position the cursor anywhere on the screen.

Syntax: **LOCATE** x y

x, y Specifies the co-ordinates.

Note: Co-ordinates start from 0. For example, the first character on the screen is at co-ordinate 0, 0.

Also see [Batch Programs](#).

Command LOWER

Function:

Convert a text string to lower case. The converted text string is stored in an environment variable called LOWER.

Syntax:

LOWER [text]

text Sequence of characters.

Note:

Also see command UPPER and Batch Programs.

Command PARSE

Function:

Allows a sentence to be broken into pieces. The pieces are stored in environment variables. PARSEA contains the text that was extracted, and PARSEB contains the remainder of the text.

Syntax:

PARSE [text]

text Sequence of characters.

Note:

The PARSE command automatically breaks on comma's, quotation marks, dots, exclamation marks, colons, backslashes and frontslashes.

[Example](#) on using PARSE.

Also see [Batch Programs](#).

PARSE Example

Consider the following batch program :-

```
@ECHO off  
SET text=a,b,c,d,e,f  
PARSE %text%  
  
:loop  
IF "%PARSEB%"==" " GOTO stop  
SAY %PARSEA%  
PARSE %PARSEB%  
GOTO loop  
  
:stop
```

This batch program will display the letters ' abcdef ' on a single line.

Command **PLAY**

Function:

Play a wave sound (ie. .WAV) file.

Syntax:

PLAY [drive:][path]filename

[drive:][path]filename Specifies the sound file to play.

Note:

Command PLAY will only play a wave sound file when a suitable audio device is installed. When no audio device is installed, then no sound will be played or heard.

Wildcards are not allowed for parameter filename.

Also see command [BEEP](#), [VOLUME](#) and [Batch Programs](#).

Command **POPD**

Function:

Pop a directory from the directory stack and make this directory the current directory.

Syntax:

POPD [number]

number Number of directories to pop of the stack.

Note:

Also see command [PUSHD](#), [DIRS](#) and [Batch Programs](#).

Command PUSHD

Function:

Push the current directory onto the directory stack and change to the specified directory.

Syntax:

PUSHD [[drive:]directory]

directory Specifies the directory to change to.

Note:

When no parameters are specified then the current directory is pushed onto the stack only.

Also see command POPD, DIRS and Batch Programs.

Command READ

Function:

Read the next item from the global list. The item is stored in an environment variable called READ.

Syntax:

READ [index]

index Specifies the index of the next item to be returned.

Note:

When no parameters are specified for command READ then the next item in the list is returned in the READ environment variable.

The index into the global list start from 0.

The global list is created using command DATA.

Also see command DATA and Batch Programs.

Command READLN

Function:

Read a line of text from a file. The line of text is stored in an environment variable called READLN.

Syntax:

READLN [drive:][path]filename [linenum]

filename Specifies the text file to read.
linenum Specifies the line number to retrieve.

Note:

Line numbers start from 1.

When parameter linenum is not specified then the environment variable will be set to the total number of lines in the text file.

Example on using READLN.

Also see Batch Programs.

READLN Example

Consider the following batch program :-

```
@ECHO OFF  
  
SET calc=0  
READLN %1  
FOR %%i is 1 TO %readIn% DO {  
    READLN %1 %%i  
    IF NOT "%readIn%"="" CALC %calc%+%readIn%  
}  
ECHO Total is %calc%
```

This batch program will sum a list of numbers contained in a text file, that has been specified on the command line. Blank lines are ignored and one number per line is assumed.

Command RETURN

Function: Return to the next command following the original GOSUB command.

Syntax: **RETURN**

Note: Also see Batch Programs.

Command SAY

Function:

Display a message. This command with NOT add a carriage return - line feed at the end of the message.

Syntax:

SAY [message]

message Sequence of characters.

Note:

Example on using SAY.

Also see command ECHO and Batch Programs.

SAY Example

Consider the following batch program :-

```
@ECHO off  
SAY hello  
SAY world
```

This example will display 'hello world' on the same line.

Command SER2DATE

Function:

Convert a serial date value to a date. The date is stored in an environment variable called SER2DATE.

Syntax:

SER2DATE number

number Specifies the serial date value to convert to a date.

Note:

When the serial date value is converted to a date, then the environment variable contains the date in the international format set in the Control Panel. For example, the USA date format is typically mm-dd-yyyy and the Australian date format is typically dd-mm-yyyy.

Also see command [DATE2SER](#) and [Batch Programs](#).

Command SETINI

Function:

Set or delete an initialisation file key value.

Syntax:

SETINI section key=[value] [filename]

section	Specifies the section name.
key	Specifies the name of the key in the section to use.
value	Specifies the key value to set.
filename	Specifies the initialisation file name.

Note:

Initialisation files generally have a .INI file extension.

The file extension must be included as part of the file name when parameter filename is specified (eg. WIN.INI).

When parameter filename is not specified then WIN_ONE.INI is used. Since WIN_ONE.INI is automatically mapped to the System Registry, then setting or deleting a key value from WIN_ONE.INI will set or delete the key value from the System Registry.

When parameter value is not specified then the key will be deleted from the specified section in the initialisation file.

When parameter key is specified as an empty string (ie. "") then the complete section will be deleted from the initialisation file. The syntax is as follows :-

SETINI section "" [filename]

Also see command [GETINI](#), [GETREG](#), [SETREG](#) and [Batch Programs](#).

Command SETREG

Function:

Set or delete a value for the specified hkey in the System Registry.

Syntax:

SETREG hkey [name=[value type]]

hkey	Specifies the path of the complete key.
name	Specifies the name of the value to retrieve.
value	Sequence of characters make up the value.
type	Specifies the type of value to set.

Note:

Parameter hkey must begin with one of the following System Registry names :-

HKEY_USERS
HKEY_CURRENT_USER
HKEY_LOCAL_MACHINE
HKEY_CLASSES_ROOT

and all the sub-keys must be separated by backslashes (ie. '\ '), For example :-

HKEY_CURRENT_USER\Software\WinOne\WinOne

Parameter type may be any one of the following :-

REG_SZ	A null terminated string.
REG_DWORD	A 32 bit number.
REG_DWORD_LITTLE_ENDIAN	Same as REG_DWORD.
REG_DWORD_BIG_ENDIAN	A 32 bit number which is automatically converted to REG_DWORD format.

When only parameter hkey is specified then the hkey will be deleted from the System Registry. Similarly, when parameters value and type are not specified then the specified name will be deleted from the hkey in the System Registry.

Command SETREG is not available under Win32s, since Win32s does not support a System Registry.

Also see command [GETREG](#), [GETINI](#), [SETINI](#) and [Batch Programs](#).

Command SLEEP

Function:

Do nothing for some time.

Syntax:

SLEEP [seconds]

seconds Specifies the number of seconds to wait.

Note:

When no parameter is specified then zero seconds is assumed. Similarly, zero seconds is assumed for invalid values of parameter seconds.

Also see [Batch Programs](#).

Command STOP

Function:

Stop processing a batch program and continue processing the batch program that called this batch program (ie. via the CALL batch command).

Syntax:

STOP

Note:

When there is no batch program to continue processing, then the batch program simply ends.

Also see command END and Batch Programs.

Command STRFIND

Function:

Find the first occurrence of a string within the specified text. The index where the string occurs is stored in an environment variable called STRFIND.

Syntax:

STRFIND "string" [text]

string	A <u>string</u> that specifies the string to find.
text	A sequence of characters.

Note:

When the string does not occur within the text then the environment variable STRFIND will be set to 0.

The first character in the text is at position 1, the second character is at position 2 and so on.

Also see command [STRRFIND](#), [SUBSTR](#) and [Batch Programs](#).

Command STRPAD

Function:

Pad a text string with space characters. The padded text string is stored in an environment variable called STRPAD.

Syntax:

STRPAD LEFT | RIGHT | CENTRE width [text]

LEFT RIGHT CENTRE	Specifies how to justify the text string :-
LEFT	- insert spaces at the end of the string.
RIGHT	- insert spaces at the front of the string.
CENTRE	- insert spaces at both the front and the end of the string so that the string is centred.
width	Specifies the minimum width of the text string. When the length of the string is less than the width then space characters are inserted to pad the text string.
text	Sequence of characters.

Note:

When the text is larger than the specified width then the text is copied to the environment variable unmodified.

Also see command [STRTRIM](#) and [Batch Programs](#).

Command STRREP

Function:

Replace all occurrences of a string within the specified text with another string . The resulting string is stored in an environment variable called STRREP.

Syntax:

STRREP "old" "new" [text]

old	A <u>string</u> that specifies the string to find.
new	A <u>string</u> that specifies the replacement string.
text	A sequence of characters.

Note:

When the string (ie. parameter old) does not occur within the text then the environment variable STRREP will contain a copy of parameter text.

Also see command [STRFIND](#), [STRRFIND](#) and [Batch Programs](#).

Command STRREV

Function:

Reverse all the characters in a text string. The reversed string is stored in an environment variable called STRREV.

Syntax:

STRREV [text]

text A sequence of characters.

Note:

Also see [Batch Programs](#).

Command STRRFIND

Function:

Find the last occurrence of a string within the specified text. The index where the string occurs is stored in an environment variable called STRRFIND.

Syntax:

STRRFIND "string" [text]

string	A <u>string</u> that specifies the string to find.
text	A sequence of characters.

Note:

When the string does not occur within the text then the environment variable STRRFIND will be set to 0.

The first character in the text is at position 1, the second character is at position 2 and so on.

Also see command [STRFIND](#), [SUBSTR](#) and [Batch Programs](#).

Command STRSIZE

Function:

Determine the length of a string. The length is stored in an environment variable called STRSIZE.

Syntax:

STRSIZE [text]

text Sequence of characters.

Note:

Also see [Batch Programs](#).

Command STRTRIM

Function:

Remove leading and trailing space characters from a text string. The resulting string is stored in an environment variable called STRTRIM.

Syntax:

STRTRIM [text]

text A sequence of characters.

Note:

Also see command STRPAD and Batch Programs.

Command SUBSTR

Function:

Extract a section of text from a text string. The extracted text string is stored in an environment variable called SUBSTR.

Syntax:

SUBSTR pos size [text]

pos	Position to start extracting text from, where the first character is at position 1, the second character is at position 2 and so on.
size	Number of characters to extract.
text	Sequence of characters.

Note:

Parameter pos can also be negative. The last character is at position -1, the second last character is -2 and so on.

[Examples](#) on using SUBSTR.

Also see command [STRFIND](#), [STRRFIND](#) and [Batch Programs](#).

SUBSTR Example

Example 1:

To extract the first 5 characters from the text string 'abcdefgh', enter at the WinOne prompt :-

SUBSTR 1 5 abcdefgh

The environment variable SUBSTR will be set to 'abcde'.

Example 2:

To extract 5 characters from the text string 'abcdefgh', start from the third character, enter at the WinOne prompt :-

SUBSTR 3 5 abcdefgh

The environment variable SUBSTR will be set to 'cdefg'.

Example 3:

To extract 5 characters from the end of the text string 'abcdefgh' , enter at the WinOne prompt :-.

SUBSTR -1 5 abcdefgh

The environment variable SUBSTR will be set to 'defgh'.

Example 4:

To extract 5 character from the text string 'abcdefg', starting at the third last character in the string, enter at the WinOne prompt :-

SUBSTR -3 5 abcdefgh

The environment variable SUBSTR will be set to 'bcdef'.

Command UPPER

Function:

Convert a text string to upper case. The converted text string is stored in an environment variable called UPPER.

Syntax:

UPPER [text]

text Sequence of characters.

Note:

Also see command [LOWER](#) and [Batch Programs](#).

